

Proceedings of the Seventh IN5550 Workshop on Neural Natural Language Processing (WNNLP 2025)

Andrey Kutuzov, David Samuel, Vladislav Mikhailov,
Roxana Pop and Sondre Wold (editors)

June 02, 2025

University of Oslo, Norway

Published by

Language Technology Group

Department of Informatics

University of Oslo

Preface

We are delighted to present the proceedings of the Seventh IN5550 Teaching Workshop on Neural Natural Language Processing (WNNLP 2025). Spurred by great advancements in deep learning and language modeling approaches to NLP, this is the seventh in a series of successful annual workshops, each showcasing some of the best efforts made by MSc and PhD students completing the IN5550/IN9550 class — all tackling modern research tasks related to deep learning in NLP.

The workshop received 22 submissions, of which 19 (by 35 authors) have been accepted for publication as part of the WNNLP 2025 proceedings (this volume). The number of submissions in on paper with WNNLP 2025.

The Programme Committee has selected the paper *Balancing Accuracy, Efficiency, and Training Cost in Runtime-Constrained Food Hazard Classification* by **Aliaan Nadeem Azam, Soban Ali and Sander Harald Sævik** for the WNNLP 2025 Best Paper award. In this paper, the authors study the trade-off between predictive performance and inference speeds for classification models. Their main contribution is a new metric that balances these two concerns.

The IN5550 course offered three obligatory assignments throughout the semester, with the maximum number of points gradually increasing for each of them (6, 12 and 14 correspondingly). This year, the Outstanding Coursework award goes to the team of **Nora Winger Heggelund and Eivind Morris Bakke**, who earned maximum points for all three assignments (32 total). The submission by Nora and Eivind also had the best performance in the obligatory 2 leaderboard.

Congratulations to all the award recipients (and runners-up)!

This workshop would not have been possible without the assistance of the reviewers in our Program Committee, whose careful and constructive feedback has been an important element in improving the submitted papers. All the members of the Program Committee are listed on the next page.

And of course, we extend warmest thanks to all the participants of this workshop, who spent many a sleepless nights working on the projects that made WNNLP 2025 an exciting and stimulating event!

WNNLP 2025 Committee
Oslo; June 02, 2025

Programme Committee

A. Rudi Bråthen Bakir
Abdirahman Ali Abdulle
Aliaan Nadeem Azam
Andre Maharaj Gregussen
Andrey Kutuzov
Brage Eilertsen
Carl-Henrik Lien
Caroline Vannebo
David Samuel
Dongzhuoran Zhou
Eivind Morris Bakke
Eivind Nordahl Andersen
Elisabeth Othamar
Emil Poiesz
Filip Flyen Andresen
Greg taube
Gyula Szenté
Hamed Hussaini
Hannah Clausen
Hannah Scheufele
Henrik Bergli
Henrik Johansen
Henrik Olafsen
Håkon Liltved Hyrve
Iris S. Bore
Katarzyna Segiet
Kjetil Indrehus
Le Heng Chieu
Lucas Georges Gabriel Charpentier
Mariia Fedorova
Marius Aas Eidsaunet
Michele Chini
Mina Sheikhi
Nora Winger Heggelund
Roxana Pop
Said Hemaz
Sander Harald Sævik
Sander Segbø Løvaas
Sigurd Myhre Vittersø
Soban Ali

Sondre Wold
Tharusan Julianv Vegard Broen
Victor Moene
Vladislav Mikhailov
Yannic Hinrichs
Yuanyuan Jiang
Zulfeqar Shirzadeh

Food hazard detection track chair

Sondre Wold

Parameter-efficient definition modeling track chair

Mariia Fedorova

Pretraining Tiny Language Models track chair

Lucas Georges Gabriel Charpentier

Preference prediction and explanation track chair

Vladislav Mikhailov

Unlearning sensitive content from LLMs track chair

Andrey Kutuzov

Full dependency parsing of Norwegian track chair

David Samuel

Fact verification track chair

Roxana Pop

General Chair

Andrey Kutuzov

Contents

1	Food hazard detection track	1
	Large Language Models for Imbalanced Multi-Label Classification (<i>Michele Chini</i>)	1
	Food Hazard Detection: Experimenting with Balancing Strategies (<i>Yannic Hinrichs</i>)	12
	Assessing the Impact of Data Augmentation on Food Hazard Detection (<i>Mina Sheikhi, Iris S. Bore, Elisabeth Othamar</i>)	19
	Enhancing Food Hazard Detection with BERT-Based Text Classification and Synthetic Data Augmentation (<i>Gyula Szente, Carl-Henrik Lien, Katarzyna Segiet</i>)	29
	Balancing Accuracy, Efficiency, and Training Cost in Runtime-Constrained Food Hazard Classification (<i>Aliaan Nadeem Azam, Soban Ali, Sander Harald Sævik</i>)	39
2	Parameter-efficient definition modeling track	49
	Parameter-efficient Definition Modeling in Norwegian with LoRA and IA3 (<i>Le Heng Chieu, Yuanyuan Jiang</i>)	49
3	Pretraining Tiny Language Models track	59
	1 hour to train, time constrained pre-training for LLMs (<i>Greg taube</i>)	59
	The Impact of Masking Rate and Strategy on Tiny Language Models (<i>Henrik Johansen, Tharusan Julian, Hannah Scheufele</i>)	68
4	Preference prediction and explanation track	78
	Preference Prediction and Explanation using a Multi-head BERT Classifier (<i>Håkon Liltved Hyrve, Filip Flyen Andresen, Sander Segbø Løvaas</i>) . . .	78
5	Unlearning sensitive content from LLMs track	87
	Parameter-Efficient Unlearning for Large Language Models – Leveraging NPO, Rényi Divergence, and Sequential Data Sampling (<i>Hannah Clausen</i>)	87
	Combining Winning Strategies. Building on Submissions to SemEval 2025 Shared Task 4: Unlearning Sensitive Content from Large Language Models (<i>Emil Poiesz</i>)	99
	Unlearning Sensitive Content from Large Language Models through Synthetic Data Generation (<i>Henrik Olafsen, Said Hemaz, Henrik Bergli</i>)	106
	Triangular Trade-offs in Machine Unlearning: A Gradient-Balanced Approach for Large Language Models (<i>Brage Eilertsen</i>)	114
	Anti-clustering is probably not all you need: Unlearning sensitive content from Large Language Models with Anti-Clustering (<i>Victor Moene, Marius Aas Eidsaunet</i>)	124
6	Full dependency parsing of Norwegian track	132

Asymmetric Transfer Between Morphosyntactic Variants of Norwegian: The Case of Bokmål and Nynorsk (<i>Eivind Nordahl Andersen</i>)	132
Norwegian Parsers and their Patterns Across the Scandinavian Language Space (<i>A. Rudi Bråthen Bakir</i>)	141
7 Fact verification track	150
GQC: LLM-Based Grouped QA Consolidation for Open-Domain Fact Verification at AVeriTeC (<i>Dongzhuoran Zhou</i>)	150
(Fact) Check Your Bias (<i>Nora Winger Heggelund, Eivind Morris Bakke</i>)	162
SemQA: Evaluating Evidence with Question Embeddings and Answer Entailment for Fact Verification (<i>Kjetil Indrehus, Caroline Vannebo</i>)	176

1 Food hazard detection track

Large Language Models for Imbalanced Multi-Label Classification

Michele Chini

Correspondence: michechi@uio.no

Abstract

This study investigates strategies to improve multi-class text classification for food hazard detection with unbalanced data. We tested traditional models, such as Logistic Regression and XGBoost, against LLM models like RoBERTa. To manage class imbalances and long texts, we tried summarization and various data augmentation strategies, such as oversampling, back-translation, and generative few-shot augmentation with the LLaMA-2 7B model. Transformer-based models significantly outperformed traditional methods, particularly when augmented via balanced few-shot generation. However, aggressive augmentation negatively affected performance. We acknowledge limitations related to synthetic data quality and augmentation biases. Future work should refine augmentation techniques and evaluate their generalizability.

1 Introduction

Large language models (LLMs) like BERT (Devlin et al., 2019) (and its variants like RoBERTa (Liu et al., 2019), DeBERTa (He et al., 2021), etc.) have achieved great performance in classification tasks, including multi-class and multi-label tasks. Often, using fine-tuning techniques on pretrained LLMs on a specific dataset yields very good performance, comparable to that obtained using traditional classifiers (e.g., logistic regression or XGBoost using bag-of-words features, as shown by Galke et al. (2025)).

Recent comparative studies note that LLMs do not always perform well: with careful tuning, simpler models can sometimes perform on par with complex LLMs on specific datasets (Randl et al., 2024; Reusens et al., 2024). These results suggest that while BERT-like models generally set a high performance bar in classification tasks, baseline methods (e.g., logistic regression, SVMs, XGBoost) remain competitive under specific conditions – especially on smaller datasets or on datasets

that have extremely unbalanced distributions of classes to be predicted. Overall, fine-tuned transformer encoders tend to perform very well in language understanding and remain a valid choice for text classification, but evaluating against classical baselines is still recommended, especially in such extreme cases. In practice, many recent works choose a BERT-based model as the main classifier, since models intended for classification tasks usually only employ an encoder followed by several task-specific layers (Devlin et al., 2019; Liu et al., 2019; Conneau et al., 2020), and include simpler classifiers as baselines for comparison.

In this paper, we tried to test different data augmentation solutions to see which was the most effective. We started from standard oversampling techniques to increasingly elaborate/complex methodologies (i.e., back-translation and generation) involving the use of LM as well.

2 Data

In order to evaluate the performance of LLMs in a multiclass classification task involving highly imbalanced classes, we utilized a subset of the dataset described in Randl et al. (2024). The dataset comprises short texts, each representing a title from food-recall announcements collected by Agroknow¹ (Papakonstantinou et al., 2022) across 24 governmental and NGO sources. We worked with only titles written in English.

Three datasets were provided: training (containing 5,082 observations), validation (with 565 observations), and test (with 997 observations). Our analysis focused on two distinct tasks. The first task (ST1) involved predicting both the hazard category and the product category given a food recall title and/or text. The second task (ST2) required predicting the specific hazard and product classes from the same input. Task ST1 is inherently broader

¹<https://agroknow.com>

and thus simpler, as it encompasses fewer classes (10 unique *hazard-category* and 22 unique *product-category*). By comparison, ST2 presents a more challenging scenario due to a significant increase in complexity, involving 128 unique *hazard* classes and 1,022 *product* classes.

The task was made even more difficult by the highly imbalanced nature of the dataset, with many underrepresented classes. Starting from subtask ST1, we can notice a highly imbalanced situation, which gets worse as we move to subtask ST2. As illustrated in Figure 1, the distribution of *hazard-category* and *product-category* is heavily skewed, with *allergens* and *biological* hazards representing the vast majority of the training data, while categories such as *migration*, *organoleptic aspects*, and *food additives and flavourings* have very poor representation. Moreover, Figure 4 and Figure 5 in Appendix A highlight the complexity of interactions between hazard categories and product categories, showing substantial variation across different food types. For example, *meat*, *egg*, and *dairy products* are predominantly associated with *biological* hazards, whereas *nuts*, *nut products*, and *seeds* mainly involve allergenic hazards. This uneven class distribution significantly increases the difficulty of accurately classifying less-represented categories.

3 Methodology

In this study, we aimed to investigate possible strategies to address multiclass classification tasks characterized by severely imbalanced datasets. To this purpose, we initially explored the performance of standard machine learning classifiers (ML), specifically **Logistic Regression** and **XG-Boost**, as baseline models, using Bag-of-Words (BOW) and Term Frequency–Inverse Document Frequency (TF-IDF) encodings. After that, we explored how encoder-only language models, specifically **RoBERTa** (Liu et al., 2019), could help in this type of problem.

We also investigated whether incorporating additional textual features (other than just the *title* feature) would enhance predictive performance. In particular, we examined the effect of including longer textual descriptions (i.e., the *text* field), which have high character counts (on average 1,978 characters, with a maximum of 48,318 characters). Given this massive length, we also explored the necessity of applying text summarization as a pre-processing step to reduce dimensionality and re-

tain only relevant information. For this task, we utilized **T5-base**, a sequence-to-sequence model widely employed in summarization tasks (Raffel et al., 2023).

Finally, we addressed the issue of class imbalance by exploring methods to synthetically augment the training dataset using LLMs. To this end, we experimented extensively with various prompting techniques and different few-shot learning strategies, systematically identifying and reporting only the most successful configurations that led to improved class distribution and overall model performance.

3.1 Classifiers

In this section, we introduce the classifiers evaluated in our study. Please note that each model was trained separately for each subtask, ST1 and ST2. To ensure robustness in our evaluation, validation samples containing classes unseen during training were excluded.

The performance of each classifier was measured using macro-averaged and micro-averaged F1 scores. Moreover, to assess the classifier’s performance considering both labels simultaneously, we computed a conditional F1 score. This score is calculated as the arithmetic mean of the macro-F1 score on the *hazard* (or *hazard-category*, depending on the subtask) classification and the macro-F1 score on the *product* (or *product-category*, again depending on the subtask) classification, the latter computed only on instances where the predicted *hazard* labels match the true labels². In Section 4 (see Table 1) we report only this metric.

3.1.1 Logistic Regression

As our baseline model, we used a simple **Logistic Regression** classifier implemented with a Bag-of-Words (BoW) representation using character-level n-grams (ranging from bigrams to 5-grams), weighted with TF-IDF.

Implementation details of the baseline classifier are provided below:

- Input text was vectorized using scikit-learn’s `TfidfVectorizer` (character-level, n-grams = 2–5, `max_df` = 0.5, `min_df` = 5).
- Labels were encoded numerically via scikit-learn’s `LabelEncoder`.

²See <https://food-hazard-detection-semeval-2025.github.io>

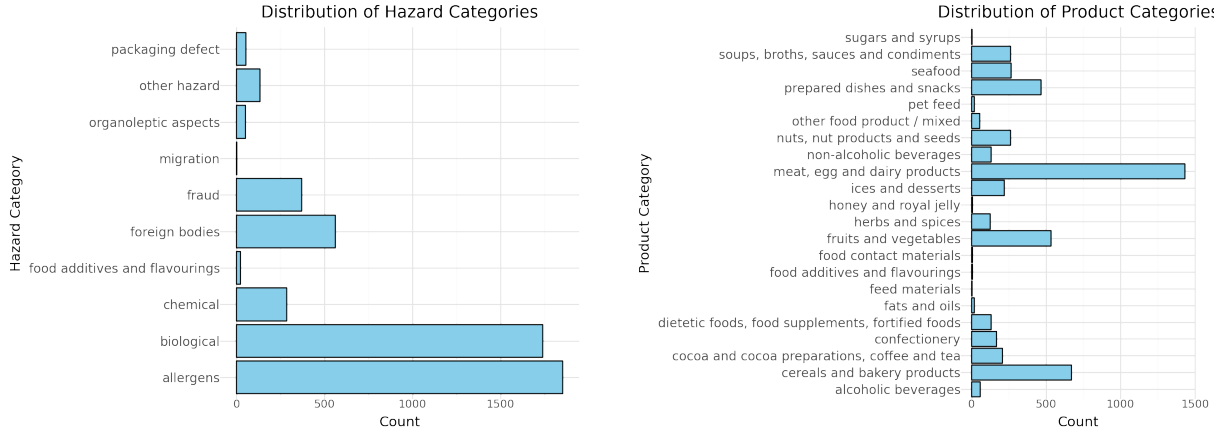


Figure 1: Number of examples for each Hazard-Category and Product-Category class in the Training dataset.

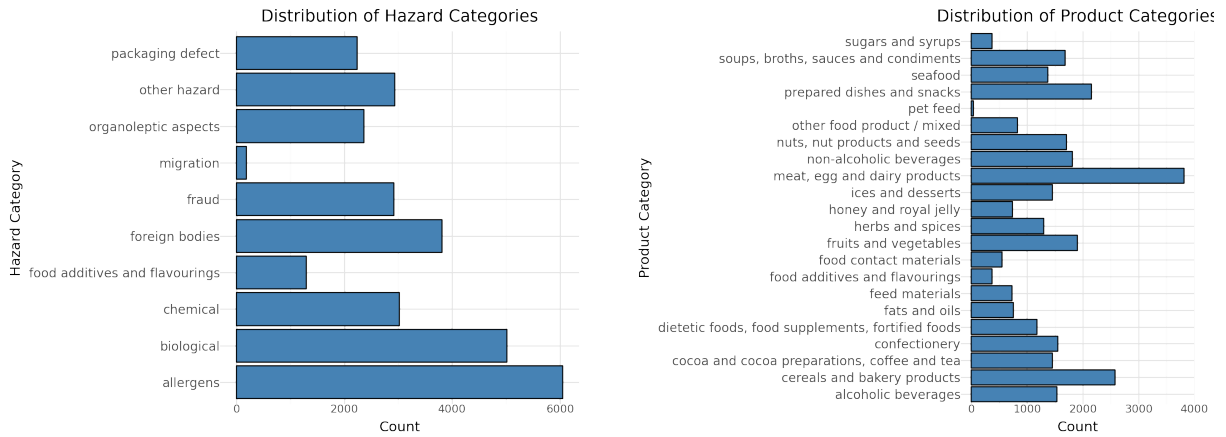


Figure 2: Number of examples for each Hazard-Category and Product-Category class in the Training dataset after augmenting the data with Llama-2 7B model ($\alpha = 0.3$, max_length = 100) and removing the generated invalid data.

- Logistic Regression classifiers were trained with a maximum of 1,000 iterations.

3.1.2 XGBoost Classifier

As an alternative to our logistic regression baseline, we explored the performance of an **XGBoost** model, as it is commonly used as a baseline in many comparative studies between LLMs and traditional ML models.

In this approach, we maintained the same structure as the logistic baseline, employing a TF-IDF-based character-level n-gram representation. To mitigate class imbalance, we implemented class weighting by assigning higher importance to less frequent classes, inversely proportional to their frequency.

We ran the XGBoost model under the following configuration: number of estimators: 100, maximum tree depth: 6, evaluation metric: Multiclass log loss (mlogloss).

3.1.3 Encoder-only Transformers

Transformer-based models have boosted the natural language processing field thanks to the work of Vaswani et al. (2017). Subsequently, models like BERT (Devlin et al., 2019) introduced encoder-only architectures that proved highly effective in numerous tasks, especially in classification.

For our purpose, following the work of Randl et al. (2024), we chose and fine-tuned the base variant of **RoBERTa** (125M params) (Liu et al., 2019), which uses the structure introduced by BERT_{base}.

For our multi-class classification task, we adapted the standard RobertaModel³. Given our dual-label setting, we extended the model architecture by adding two distinct classification heads, each implemented as a single fully connected (nn.Linear) layer, and optimized the model using separate cross-entropy losses for each head. Both classification heads take the output from

³https://huggingface.co/docs/transformers/model_doc/roberta

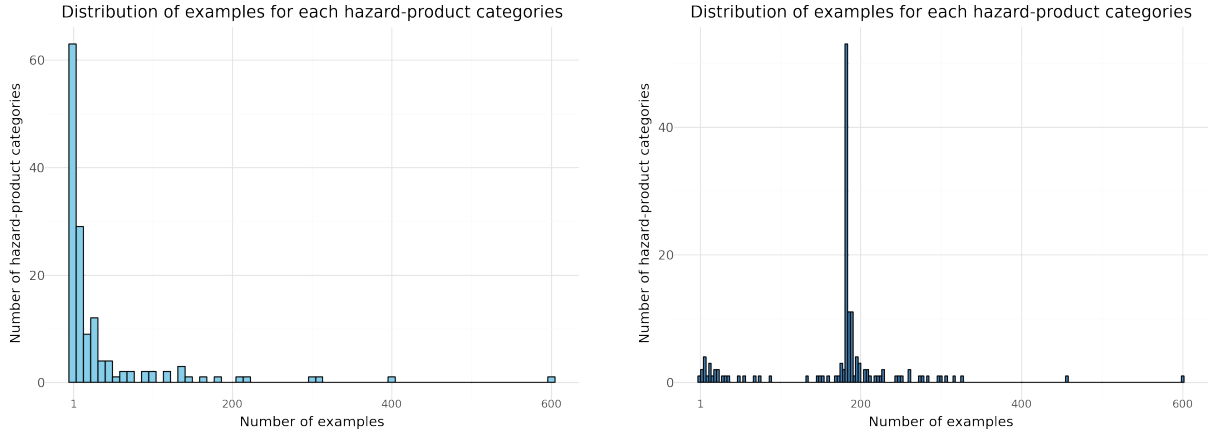


Figure 3: Left: Distribution of the number of observations for each hazard-category/product-category unique pair in the training dataset. Right: the same distribution after augmenting the data with Llama-2 7B model ($\alpha = 0.3$, $\text{max_length} = 100$) removing the generated invalid data. As can be seen there are still classes with few observations.

RoBERTa’s encoder, apply dropout for regularization, and then predict the labels for the *hazard* and *product* classes independently (depending on the subtask, we consider either *hazard-category/product-category* or *hazard/product*). Implementation details include the concatenation of *title* and *text* features when required by the task configuration. Training optimized via AdamW with a learning rate of $2e-5$. Early stopping based on validation loss with a patience of 5 epochs to avoid overfitting. Maximum of 50 epochs. Batch size: 36.

To address issues caused by long texts (possibly exceeding RoBERTa’s context window limit of 512 tokens (Sun et al., 2020)), we attempted to preprocess our data by shortening the *text* field. Specifically, we utilized the T5-base model (Raffel et al., 2023) to generate concise summaries of the original *text* field. However, this preprocessing step did not yield noticeable improvements in predictive performance compared to fine-tuning directly on the original (though truncated) *text* feature. It is possible that summarization excessively reduced general information, potentially removing valuable context necessary for accurate predictions. See Sections 4 and 5 for a more in-depth discussion.

3.2 Data Augmentation

In many classification problems, special attention is always placed on situations where the dataset has extremely unbalanced data. In this case, training a model on the original data is usually not enough to make the model learn different things apart the most represented class in the dataset.

To address this issue, studies have explored dif-

ferent techniques ranging from data-level interventions (resampling and augmentation) to algorithmic modifications (cost-sensitive losses and specialized training procedures), see Fernández et al. (2018) and (Papakonstantinou et al., 2022). Below we tested some of these approaches, with an emphasis on methods using LMs.

3.3 Baseline

To address the curse of class imbalance, we applied a very basic oversampling approach as a baseline method. This technique aims to equalize class representation by increasing artificially (using random sampling) the number of examples for minority classes.

To achieve so, we combined both *hazard-category* and *product-category* into a single class, defined explicitly as the concatenation of the two. Subsequently, we determined the most frequent class combination and computed the necessary samples to generate for each minority class according to what we called IFR method (Inverse Frequency Ratio).

This method determines the number of synthetic samples to generate for each combination of *hazard-category* and *product-category* calculating the number of new examples proportionally to an α control parameter. Specifically:

$$\text{Samples to generate} = \alpha \times (n_{\text{max}} - n_i)$$

where n_{max} represents the count of samples in the most frequent class combination (in this case, each class is each unique pair *hazard-category/product-category*), n_i the current count

for a given class combination and $\alpha \in [0, 1]$ controls the intensity of the sample strategy. We experimented with $\alpha = (0.3, 0.4, 0.5)$.

3.4 Back-translation

Back-translation is a widely adopted data augmentation technique used in natural language processing tasks, involving translating text from its original language into an intermediate language and subsequently translating it back into the original language. This approach aims to introduce linguistic variability into the training dataset enhancing the model robustness and generalization capabilities by actually augmenting the data in a relatively straightforward way (Sugiyama and Yoshinaga, 2019).

For our backtranslation-based augmentation, we selected the LM **NLLB-200-distilled-600M** model developed by Facebook (Team et al., 2022). We chose this model for its multilingual capabilities, including high-quality translation performance across diverse languages, and for its computational efficiency relative to its substantial representation power (600 million parameters).

In our setup, the model was explicitly configured for translations between English (source language) and French (pivot language). During backtranslation, each text entry (title and main text fields) was first translated from English to French and subsequently retranslated back to English, thus generating textual variations.

We augmented the data using again the IFR method. To control the augmentation intensity, we tested with $\alpha = (0.4, 0.3)$.

3.5 Few-shot generation

Finally we tested a data augmentation strategy based on few-shot learning utilizing the IFR method described above.

We selected the **LLaMA-2 7B-chat-hf** model, a decoder-only transformer architecture, suitable for generative tasks, proving extremely competitive even in comparison with proprietary LLMs (Touvron et al., 2023). We run the model to generate new examples using the following key configuration settings: Temperature: 0.3, Top-p: 0.9 and Top-k: 40.

Given the large dimensionality of our data and the numerous possible subcategories, we decided to sample up to three examples from each *hazard-category* and *product-category* pair to create effective few-shot prompts (so we did not choose them manually for each sub-class). If a class contained

three or fewer examples, we utilized all existing instances. The final prompt used, shown below, is the result of extensive experimentation and was chosen on the basis of its demonstrated flexibility, brevity and effectiveness:

Rephrase the examples above to generate a food
 $\hookrightarrow$ recall statement for <Product Category>
 $\hookrightarrow$ product due to presence of <Hazard Category>
 $\hookrightarrow$ then stop.

Example 1:
 Title: <Sample Title 1>.
 Text: <Sample Text 1>.

Example 2:
 Title: <Sample Title 2>.
 Text: <Sample Text 2>.

Example 3:
 Title: <Sample Title 3>.
 Text: <Sample Text 3>.

Example 4:"

For classes with less than three examples, of course, we presented just the existing ones counting up to that number.

We implemented as well a basic data quality control step for the generated data. We identified and removed invalid samples from the augmented dataset (i.e., all examples without Title: and Text: words). This ensured that only correct synthetic data were used. Although it may seem banal, this step had significant repercussions on the rebalancing of the dataset. Let us focus on Figure 1, Figure 2 and Figure 3. Comparing Figure 1 to Figure 2, it can be seen that we have greatly increased the minority classes; however, after controlling for data quality, the distribution of the number of samples for each unique category of *hazard-category*\product-category becomes the one on the right in Figure 3, where we can see the mass of classes with few observations still remains high (although we can still see an effect compared to such original distribution on the right).

4 Results

The results of all our tests can be found in Table 1.

As we can notice, both baseline models (i.e., Logistic Regression and XGBoost) employing only the *title* field without any additional preprocessing or data augmentation did not perform well, highlighting the complexity of dealing with unbalanced data. The introduction of RoBERTa significantly enhanced performance, suggesting that using LLMs for text classification task can help

considerably, especially in such situations with multiclass classification tasks and imbalanced datasets.

Adding textual information (*Title+Text* features) without summarizing them and without data augmentation resulted in improvement in terms of performance (conditional F1 of 0.7520 for ST1 and 0.4863 for ST2, the latter the highest scores among all the experiments), confirming the importance of using all possible information that could help in classifying. However, applying summarization to reduce input text length slightly decreased the performance, indicating that the summarization process might excessively reduce critical context required for accurate classification.

Increasing data has proven to be particularly useful, regardless of the strategy used. This seems to confirm what was expected: increasing data in situations of extreme imbalance always has a positive effect. Standard oversampling provided moderate performance gains (conditional F1 of 0.7398 for ST1 and 0.4736 for ST2). More sophisticated techniques, such as back-translation, yielded incremental improvements, with an optimal setting of $\alpha = 0.4$ achieving the best conditional F1 of 0.7611 for ST1.

The data augmentation strategy that proved to be most efficient was by few-shot using LLaMA-2 7B. Specifically, using $\alpha = 0.3$, maximum sequence length of 100 tokens resulted in the highest conditional F1 score among all methods for ST1 (0.7847). Note how increasing the maximum length to 200 tokens or using a more aggressive data augmentation strategy ($\alpha = 0.4, 0.5$) did not further improve performance.

5 Discussion

The empirical results presented in the previous section provide several key insights.

First, we can see how using LM as predictive models in this context has greatly improved performance over traditional ML models. This could be explained both by an LM’s better ability to predict in such an unbalanced context but also by the fact that we are predicting textual data, thus, not with intrinsic numeric meaning for a classical ML model.

As a second thing (also quite expected) is how using as much information as possible could further increase the performance of all the models tested in this condition (*title* column only or *title* column + *text*). It is also interesting to see how summarizing

the text field did not help to improve the predictive capabilities of the models, but worsened them, suggesting that such summarization removed useful information for the model. This is certainly something to keep in mind when considering methods to succeed in exploiting the entire context window of the model.

Regarding data augmentation, we can say that we achieved exactly the expected results. In general, augmenting data in such a pronounced unbalanced situation is always a good idea in order to be able to make the model learn something and consequently get better results. Specifically, the less gain we obtained was with a basic oversampling technique such as pure oversampling. In this setting, we just duplicated data, which is clearly not optimal. Moving instead to data augmentation methods involving LMs, we saw that both backtranslation and few-shot generation techniques performed well. Again, the more elaborate technique proved to be the most effective, suggesting the generation of more diverse examples than the ones generated using simple backtranslation.

The few-shot approach using LLaMA-2 proved to be the best strategy, improving significantly the conditional F1 scores for ST1, demonstrating the potential of LMs for data augmentation of textual data. We should note as well that the best performances were achieved with $\alpha = 0.3$ (the less-aggressive augmentation we tested), which suggests that there is an equilibrium beyond which further augmentation does not contribute positively. This could also be due to the fact that our generated examples were not sufficiently realistic. Consequently, this current threshold could also indicate that a generation of data not of extreme quality like this one should never exceed a certain threshold but that, with generated data of higher quality, this threshold could significantly increase.

6 Limitations and Future Directions

Despite the promising results obtained in our experiments, several limitations need to be carefully considered. First of all, on the data augmentation strategy we saw that methods such as back-translation and generative few-shot augmentation have proven to be effective, but still, the quality of generated data heavily depends on the selected generative model and the prompt engineering process. Therefore, further evaluation on this should be made to ensure better results.

Model	Features	Text Summ.	Data Aug.	(α)	(Max Length)	ST1 - F1	ST2 - F1
Logistic Reg.	Title	N	-	-	-	0.4979 (0.4830)*	0.2013 (0.1407)*
XGBoost	Title	N	-	-	-	0.6278 (0.5693)*	0.0374 (0.0291)*
RoBERTa	Title	N	-	-	-	0.6902 (0.6489)*	0.3755 (0.3661)*
RoBERTa	Title	Y	-	-	-	0.6997 (0.5982)*	0.3696 (0.3562)*
RoBERTa	Title+Text	N	-	-	-	0.7520 (0.7462)*	0.4863 (0.4647)*
RoBERTa	Title+Text	Y	-	-	-	0.7163 (0.7044)*	0.4177 (0.3970)*
RoBERTa	Title+Text	N	Oversampling	0.3	-	0.7398 (0.7609)*	0.4736 (0.4504)*
RoBERTa	Title+Text	N	Back-translation	0.3	100	0.7580 (0.7692)*	0.4854 (0.4524)*
RoBERTa	Title+Text	N	Back-translation	0.4	100	0.7611 (0.7709)*	0.4514 (0.4653)*
RoBERTa	Title+Text	N	Llama-2 7B	0.3	100	0.7847 (0.7663)*	0.4637 (0.4292)*
RoBERTa	Title+Text	Y	Llama-2 7B	0.3	100	0.6536 (0.6389)*	0.3868 (0.3758)*
RoBERTa	Title+Text	N	Llama-2 7B	0.3	200	0.7318 (0.7348)*	0.4453 (0.4376)*
RoBERTa	Title+Text	N	Llama-2 7B	0.4	100	0.7701 (0.7454)*	0.4578 (0.4484)*
RoBERTa	Title+Text	N	Llama-2 7B	0.5	100	0.7688 (0.7449)*	0.4566 (0.4430)*

Table 1: Experimental results comparing baseline models and Transformer-based approaches with various preprocessing techniques. *F1 computed on Test data, **just for exploratory insight**.

A limitation in the generative augmentation process is the exclusive use of a three-shot prompting strategy with Llama 7B-parameter LLM. Future research should explore alternative strategies such as chain-of-thought, or even fine-tuning smaller LLMs on task-specific examples before augmenting the data. These could lead to more likely and higher-quality augmented data.

Second, the optimal balance between augmentation intensity and model performance identified in our experiments suggests that overly aggressive augmentation could lead to worst performances. The specific thresholds found (e.g., optimal α values) might vary significantly across different tasks and domains. Hence, more effort should be considered in considering different datasets and scenarios to generalize these results.

Moreover, our experiments used primarily a single transformer model variant (i.e., RoBERTa-base) to perform text-classification with LMs. Given the rapidly evolving landscape of transformer architectures and their variants, future research should explore other advanced models (e.g., larger models, multilingual transformers, or domain-specific adaptations) to assess potential performance improvements.

In terms of summarization, our findings indicated a trade-off between computational efficiency and predictive accuracy. Future research could explore adaptive summarization techniques that dynamically preserve critical contextual information while minimizing input length.

Additionally, the generative few-shot augmentation approach employed lacked explicit quality control measures regarding the initial selection of

few-shot examples. The absence of thorough quality assessments for these selected examples could have introduced suboptimal or misleading prompts, thus affecting the quality of the synthetic data generated. Future research should incorporate systematic quality checks or criteria to ensure the representativeness and adequacy of selected few-shot prompts.

Another point to consider is that our data augmentation process primarily focused on the *hazard-category* and *product-category* labels relevant to Subtask 1 (ST1). This approach likely affected Subtask 2 (ST2), which requires augmentation at a more granular *hazard* and *product* level. Consequently, ST2 performance might have been systematically penalized due to this bias toward ST1-related categories (indeed, with no surprise the best F1 score for ST2 was obtained just finetuning RoBERTa without augmenting the data). Addressing this limitation by exploring augmentation strategies for both subtasks represents an important future direction.

Finally, it is important to enlighten a limitation in the training pipeline adopted across all tested classifiers. Hyperparameter configurations were not fully tuned; we just relied on commonly used values reported in the literature. This might have limited the full potential of each model. Future work should incorporate systematic hyperparameter tuning to enable a fairer and more robust comparison, in line with best practices in the machine learning community.

References

- Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. [Unsupervised cross-lingual representation learning at scale](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of deep bidirectional transformers for language understanding](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Alberto Fernández, Salvador García, Mikel Galar, Ronaldo C Prati, Bartosz Krawczyk, and Francisco Herrera. 2018. *Learning from imbalanced data sets*, volume 10. Springer.
- Lukas Galke, Ansgar Scherp, Andor Diera, Fabian Karl, Bao Xin Lin, Bhakti Khera, Tim Meuser, and Tushar Singhal. 2025. [Are we really making much progress in text classification? a comparative review](#). *Preprint*, arXiv:2204.03954.
- Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. [Deberta: Decoding-enhanced bert with disentangled attention](#). *Preprint*, arXiv:2006.03654.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. [Roberta: A robustly optimized bert pretraining approach](#). *Preprint*, arXiv:1907.11692.
- Mihalis Papakonstantinou, Manos Karvounis, Giannis Stoitsis, and Nikos Manouselis. 2022. [Deploying a Scalable Big Data Platform to Enable a Food Safety Data Space](#), pages 227–248. Springer International Publishing, Cham.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. [Exploring the limits of transfer learning with a unified text-to-text transformer](#). *Preprint*, arXiv:1910.10683.
- Korbinian Randl, John Pavlopoulos, Aron Henriksson, and Tony Lindgren. 2024. [CICLE: Conformal in-context learning for largescale multi-class food risk classification](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7695–7715, Bangkok, Thailand. Association for Computational Linguistics.
- Manon Reusens, Alexander Stevens, Jonathan Tonglet, Johannes De Smedt, Wouter Verbeke, Seppe vanden Broucke, and Bart Baesens. 2024. [Evaluating text classification: A benchmark study](#). *Expert Systems with Applications*, 254:124302.
- Amane Sugiyama and Naoki Yoshinaga. 2019. [Data augmentation using back-translation for context-aware neural machine translation](#). In *Proceedings of the Fourth Workshop on Discourse in Machine Translation (DiscoMT 2019)*, pages 35–44, Hong Kong, China. Association for Computational Linguistics.
- Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2020. [How to fine-tune bert for text classification?](#) *Preprint*, arXiv:1905.05583.
- NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Hefernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, and 20 others. 2022. [No language left behind: Scaling human-centered machine translation](#). *Preprint*, arXiv:2207.04672.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023. [Llama 2: Open foundation and fine-tuned chat models](#). *Preprint*, arXiv:2307.09288.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). *CoRR*, abs/1706.03762.

A Appendix

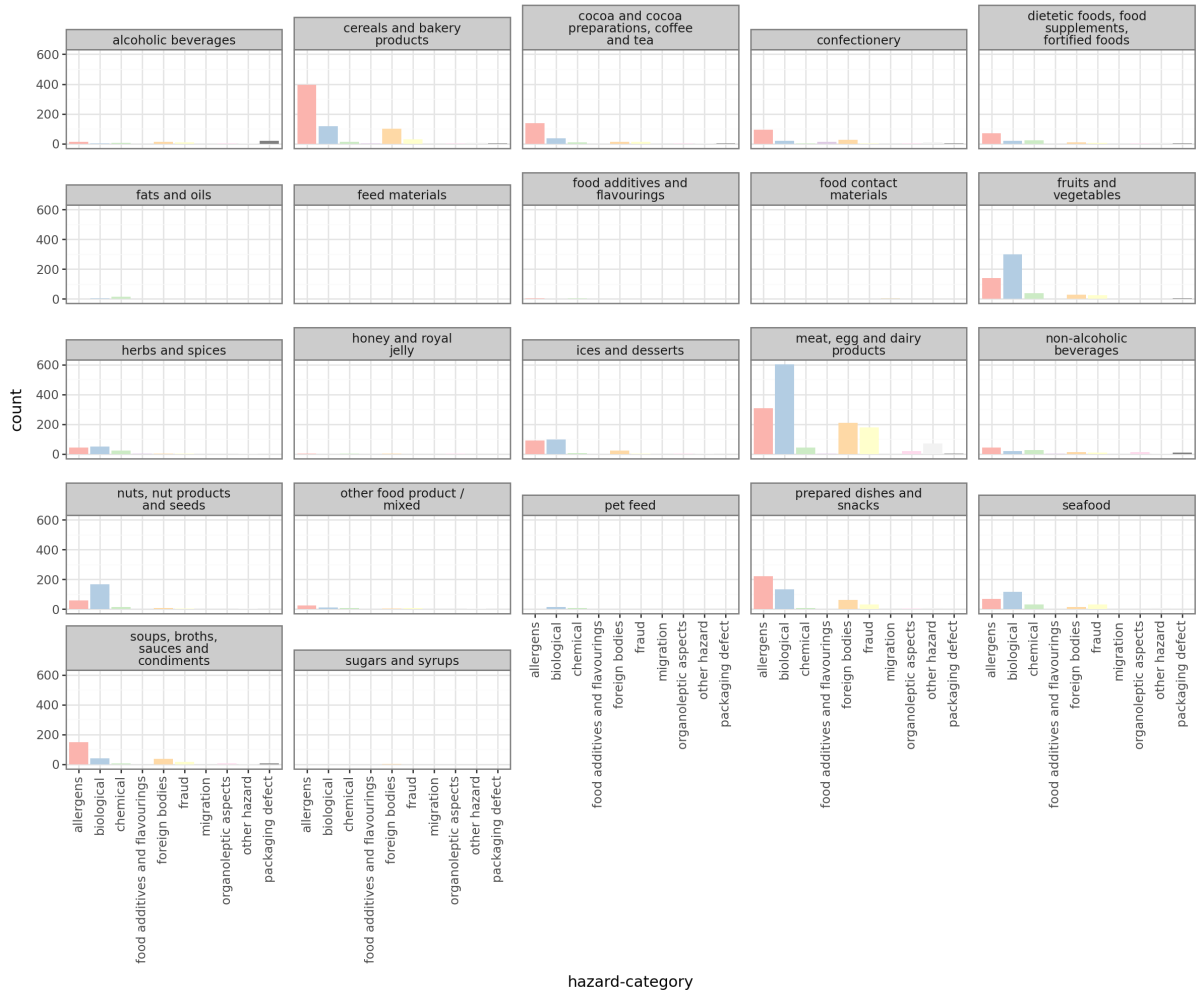


Figure 4: Distribution of *hazard-category* across different *product-category* in the dataset. Each subplot represents a specific product category, illustrating the frequency of associated hazard categories and highlighting significant class imbalances present within the data.

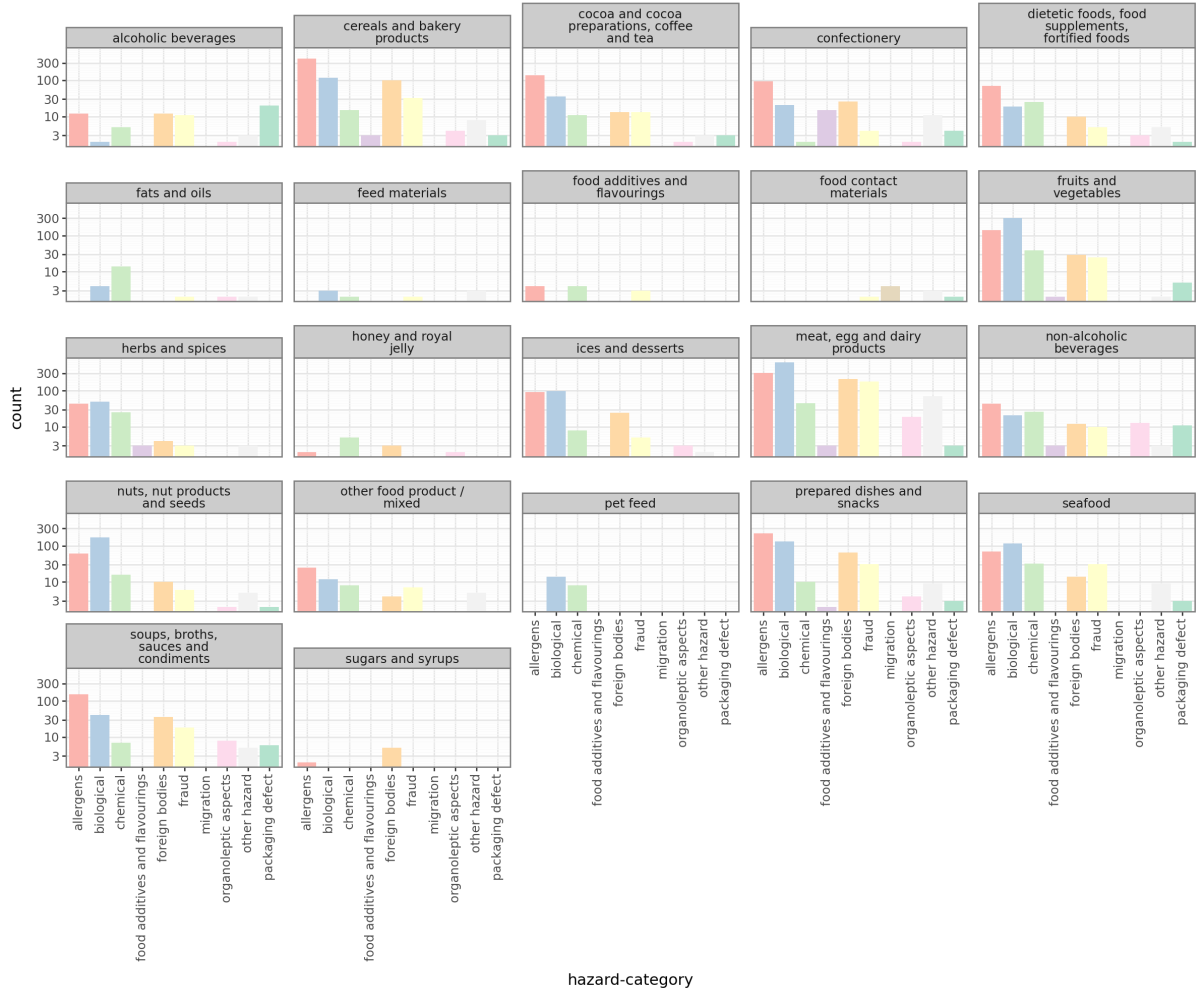


Figure 5: Distribution of *hazard-category* across different *product-category* in the dataset. Each subplot represents a specific product category, illustrating the frequency of associated hazard categories and highlighting significant class imbalances present within the data. $\log_{10}$ scale.

Examining Strategies to deal with imbalanced data using the example of SemEval 2025 Food Hazard Detection Challenge

Yannic Hinrichs

Department of Psychology, University of Oslo

Abstract

This paper investigates strategies for addressing imbalanced data in the context of the SemEval 2025 Food Hazard Detection Challenge, which focuses on classifying food recall statements. We evaluate three methods: downsampling, oversampling, and focal loss. Our findings indicate that reducing training data size for the sake of balance impoverishes performance of the classifier for both subtasks. Inflating the dataset with duplicates of minority samples to increase balance improves performance on ST2 but not ST1 compared to the baseline. Lastly, the implementation of focal loss improves performance on both metrics for every configuration of its parameters alpha and gamma. However, the best configurations of alpha and gamma for the two subtasks are different.

1 Introduction

Contaminated food poses a significant risk to human health (de Noordhout et al., 2014; Majowicz et al., 2014). Food recalls, however, are not published centrally, but on different websites by the respective food companies. A combination of crawling these websites and natural language processing (NLP) on the crawled texts offers a viable approach to gather and efficiently analyse these texts. One efficient method of analysis is text classification whereby the food recall statement is classified into product or hazard categories. This can help distinguishing more serious hazards from less serious ones, among others. SemEval 2025 Task 9: The Food Hazard Detection Challenge (Website: <https://food-hazard-detection-semeval-2025.github.io/>) posed such a task. Based on a dataset from Randl et al. (2024) that features recall

titles from official food agency websites which were manually labelled for their hazard and product category by experts in the field, the challenge is to build an NLP classifier that classifies a recall title and/or text into a product category, hazard category, product, and hazard.

However, what makes this task challenging is that it is highly skewed, meaning that some product and hazard categories occur more than thousand times while others appear only three to five times. Training natural language classifier on imbalanced data creates significant challenges which can result in poor performance (Shaikh et al., 2021). For that reasons, techniques have been developed to deal with imbalanced training data (Padurariu & Breaban, 2019).

This paper aims to experimentally test different strategies of dealing with imbalanced training data and see how they affect the performance of the classifier. In the following paragraph, I will outline the problems that come with imbalanced training data as well as techniques to remedy them. Next, I will state my research question and subquestions before describing the method and the results.

2 Background and Related Work

2.1 Natural language classifiers

Natural language classifiers are machine learning (ML) models specifically designed to categorize text based on its content (Goldberg, 2017). These classifiers can be used for various tasks, such as sentiment analysis, topic classification, spam detection, and more. They analyze textual data and assign predefined labels to it. Modern classifiers employ neural network architectures (Goldberg, 2017). The most advanced family of them are transformer models, such as BERT (Devlin et al., 2019). The classifier models encode text

documents as dense vectors to represent their meaning, feed them into a neural network, whose architecture vary across models, and linear heads for the final classification (Goldberg, 2017). During the training phase, the model learns to associate input features with the corresponding labels by minimizing a loss function, which quantifies the difference between the model's predicted outputs and the actual labels. Since training LLMs is cost and time consuming, most natural text classification tasks use a pretrained LLM and finetune it on a specific data set.

2.2 Finetuning on specific data

Finetuning is a transfer learning technique that involves taking a pre-trained model that has already learned useful features from a large dataset and further training it on a smaller, task-specific dataset to adapt the model to a particular problem (Wei et al., 2021). As a result, the finetuned model performs better on the task for which it was finetuned than an LLM without further training. Technically, what happens during fine-tuning is that the model's weights are adjusted slightly rather than trained from scratch. This way, the model retains useful representations learned in the pre-training phase while allowing the model to specialize in the new, smaller dataset (Wei et al., 2021). For the SemEval 2025 Task 9: The Food Hazard Detection Challenge, an existing LLM is fine-tuned on a subset (training set) of Randl et al.'s (2024) dataset to obtain a model specialised on food hazard classification. However, the imbalance of this dataset can pose problems to the fine-tuning as I will discuss in the next section.

2.3 Problems of imbalanced training data

Imbalanced data refers to datasets where the distribution of classes is not uniform, meaning that some classes are represented by a significantly larger number of instances than others. When LLMs are trained or fine-tuned on imbalanced data, the classifier usually becomes biased toward the majority class, often failing to accurately identify or predict the minority class (Choi et al., 2021). This leads to poor overall performance, especially for the underrepresented categories. The problem worsens when multiple classes should be predicted: This is because the different classes to be predicted might be differently imbalanced. As a consequence, traditional balancing techniques become less effective and negative side effects may appear,

such as further reduced accuracy for minority classes (García et al., 2018).

2.4 Strategies for dealing with imbalanced data

Strategies to deal with imbalanced can be categorised into data-level approaches where the training data itself is modified, and algorithm-level approaches where the algorithms used for the training (finetuning) are tweaked.

2.4.1 Resampling

Resampling is one of the data-level methods which address the imbalanced directly at the data-level. Two resampling strategies can be differentiated which result in a more balanced distribution of the training data: Downsampling, where the majority categories are reduced by removing a random selection of instances. While this increases the balance of the data, the training data is reduced which can also lower the performance. Thus, a trade-off has to be found between training data size and imbalance of the data. The second strategy is oversampling, where the minority categories are filled up – either by duplicating instances or by synthetically generate new instances.

2.4.2 Focal Loss

A second strategy examined in this paper is implementing focal loss (Lin et al., 2017). It is an extension of cross-entropy with class weights which allows for a more refined approach to hard-to-handle cases and class imbalance. It has two key parameters – gamma and alpha. Alpha controls the class weights, which is similar to what the class weight matrices do when using cross-entropy with class weights. A higher alpha gives more importance to minority classes and should therefore be used for highly skewed datasets. Gamma sets the ‘focus’ on hard-to-classify samples. If gamma is zero, focal loss reduces to the usual cross-entropy loss. A higher gamma means that difficult-to-classify samples weigh stronger on the calculated loss.

3 Research Question

My research question thus is: *How do different strategies of dealing with imbalanced data increase the performance of the food hazard classifier?*

4 Experiments

I tested two data-level methods, downsampling and oversampling, as well as one algorithm-level method, i.e. focal loss.

4.1 Model architecture and baseline

As a base for my model, I chose ‘bert-base-uncased’, on top of which I added four linear layers as classifier heads, one for each category to be predicted. In addition, I appended the output of the product category classifier to the input of the hazard category classifier and product classifier. Similarly, I added the output of the hazard category classifier to the hazard classifier. In the baseline setup, the classifier is trained on the original provided training dataset ($N = 5082$), and the loss is calculated using an averaged cross-entropy loss across the four classes. As evaluation method, I calculated ST1 and ST2 scores for the two subtasks, which are averaged F1_macro_scores – averaged across hazard category and product category for subtask 1, and averaged across hazard and product for subtask 2.

4.2 Downsampling

The dataset was downsampled for the ‘hazard-category’ and ‘product-category’. However, the sampling was done row-wise, therefore, the values for ‘hazard’ and ‘product’ were included in the subset, even though they were not explicitly balanced.

Three strategies were tested (see Table 1). Each strategy features a maximum size for the majority classes of ‘hazard-category’ and ‘product-category’. From classes with more instances, the defined maximum size was randomly sampled from that category. The maximum sizes for the three tested strategies were set based on a graphic exploration of the distribution. The lower the maximum category size, the more equal the distribution of the subset is, but the smaller its size. The final dataset sizes can be found in Table 1.

Strategy	Maximum Size ‘hazard-category’	Maximum Size ‘product-category’	Resulting dataset size
Down_1	100	100	624
Down_2	300	150	1247
Down_3	600	270	2071

Table 1: Downsampling Strategies

4.2.1 Results

The results can be found in Table 3 and Figure 1. Down_1 with the most balanced but smallest dataset performed by far worse than the other two downsampling strategies and baseline on both metrics. Down_2 with double the training data size than Down_1 performed much better than Down_1 on both metrics. However, further doubling the training data size only results in a small improvement in ST1 and ST2 compared to the performance increase from Down_1 to Down_2. Compared to the baseline, however, all three downsampling strategies performed worse than on both metrics.

4.3 Oversampling

Same as for the downsampling, the oversampling was done for the ‘hazard-category’ and ‘product-category’ and the sampling was done row-wise. Inversing the strategy used for the downsampling, now a minimum class size was set for the minority classes of ‘hazard-category’ and ‘product-category’. From classes with less instances, the available instances were duplicated until they reached the minimum size.

Since oversampling for both categories proved challenging, only the ‘product-category’ was oversampled. The ‘hazard-category’ was ‘indidentally’ increased due to the row-wise oversampling; however, it did not reach the specified minimum size. This limitation is discussed in the discussions section. Three strategies were tested (Table 2). Each strategy features a minimum size for the minority classes of ‘product-category’.

Strategy	Maximum Size ‘product-category’	Resulting dataset size
Over_1	100	5795
Over_2	200	6935
Over_3	300	8509

Table 2: Oversampling Strategies

	Subtask	F1 Hazard Type (ST1)	F1 Product Type (ST1)	F1 Averaged
		F1 Hazard (ST2)	F1 Product (ST2)	
Baseline	ST1	0.5860	0.5232	0.5546
	ST2	0.1454	0.0241	0.0847
Down_1	ST1	0.1659	0.0935	0.1297
	ST2	0.0209	0.0011	0.0110
Down_2	ST1	0.5151	0.2814	0.3983
	ST2	0.0640	0.0077	0.0358
Down_3	ST1	0.5242	0.3293	0.4267
	ST2	0.0876	0.0053	0.0465

*highest values (compared to the same subtask) in bold

Table 3: Results for the downsampling strategies

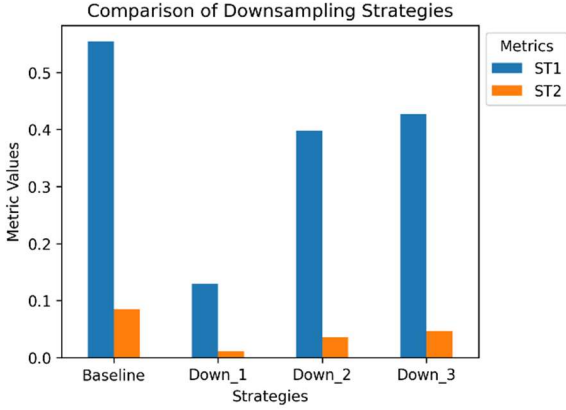


Fig. 1: Comparison of downsampling strategies

4.3.1 Results

The results for the experiments with oversampling strategies can be found in table 4 and figure 2. Although the performance on the F1_macro for hazard type decreased with increasing minimum size for ‘product-category’ and thus, overall training data size, the performance on all over partial F1_macros increased with it, as did the F1 Averaged Macro for both subtasks. Compared to the baseline, however, all three oversampling strategies perform considerably worse on ST1, but better on ST2. Over_1 performs as good as the baseline on ST2, Over_2 and Over_3 better in this order.

4.4 Focal Loss

To experiment with focal loss, I used the original training data again. I varied the alpha and gamma

inductively, i.e. by reviewing the results and testing in both directions whether the performance increases. The range of the values, and recommendations for the starting values, I gained from Cai et al. (2025). The tested values can be found in table 5.

4.4.1 Results

The results can be found in table 5 and figure 3. Compared to the resampling strategies, the performance differences caused by tweaking the focal loss are smaller. Focal_1 performed best on ST1, followed by Focal_5, -2, -3, -4, and the baseline. Focal_3 performed best on ST2, followed by Focal_5, -2, -4, -1, and the baseline. There is no linear association between parameter alpha and performance on ST1 and ST2 because if we compare Focal_2, Focal_4, and Focal_5, where gamma is constant, we see that lowering alpha from 0.75 to 0.65 marginally increases performance on both metrics, but further lowering alpha from 0.65 to 0.5 impoverishes performance on both metrics. However, there is a linear association between gamma and performance on ST2. Increasing gamma from 2 to 3 and from 3 to 4 increases performance on ST2. Thus, it seems that increasing gamma, and thus, putting a stronger focus on hard-to-classify examples, benefits performance on tasks with many categories, such as hazard (number of categories = 128) and product (number of categories = 1022).

4.5 Overall comparison of balancing strategies

Lastly, we can compare the best solutions from each balancing strategy, that is Focal_1, Over_3 and Down_3, and the baseline. Over_3, which has

	Subtask	F1 Hazard Type (ST1)	F1 Product Type (ST1)	F1 Averaged
		F1 Hazard (ST2)	F1 Product (ST2)	
Baseline	ST1	0.5860	0.5232	0.5546
	ST2	0.1454	0.0241	0.0847
Over_1	ST1	0.0903	0.5348	0.3125
	ST2	0.1423	0.0333	0.0878
Over_2	ST1	0.0834	0.6640	0.3737
	ST2	0.1950	0.0403	0.1177
Over_3	ST1	0.0761	0.7374	0.4068
	ST2	0.2178	0.0559	0.1369

*highest values (compared to the same subtask) in bold

Table 4: Results for Oversampling Strategies

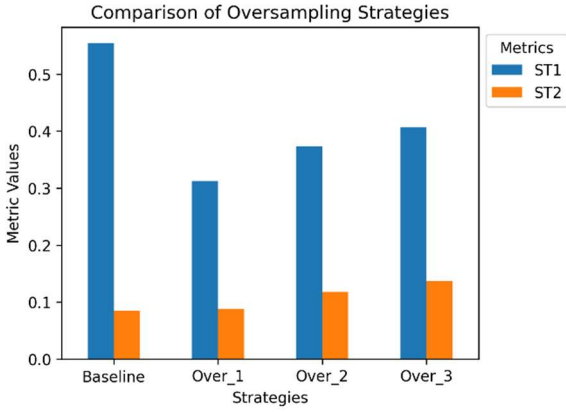


Fig. 2: Comparison of Oversampling Strategies

the largest training dataset ($N = 8509$), performs best on ST2 but worst on ST1. Focal_1 performs best on ST1, and second best on ST2. The baseline is better on ST1 than both resampling strategies, but worse than Focal_1. Furthermore, the baseline is better on ST2 than Down_3, but worse on ST2 than Focal_1 and Over_3.

5 Discussion

Three different strategies were tested to deal with imbalanced data using the example of SemEval 2025 Food Hazard Detection Challenge. The strategies were downsampling, oversampling, and focal loss. I found that downsampling, i.e. reducing the number of majority classes, leads to more balanced, but smaller datasets which yield a poorer performance than the baseline. Moving from the smallest and most balanced dataset of Down_1 up in training data size and imbalance of the data, i.e. Down_2, Down_3 and the baseline, the

performance monotonously increases on both ST1 and ST2. This shows that, at least in this range of training data size, training data size matters more to the performance of the classifier than the balance within the training data. What is more, training data size seems to matter most in the low training data range because moving from $N = 624$ to $N = 1247$ improved the performance on both metrics drastically, while moving from $N = 1247$ to $N = 2071$ only increased the performance a little suggesting a plateau effect.

Oversampling increased both training data size and balance of the data, however, at the cost of informational value since samples from minority classes were simply duplicated. In the most extreme case, a class with only 3 cases was duplicated 100 times. Despite the little added information, the different oversampling strategies yielded better results on ST2 indicating that for this subtask training data size and balance matter more than informational value.

However, all oversampling strategies led to worse performance on ST1. This is surprising because during all the oversampling strategies, the classifier learns as much from the training data as in the baseline condition, only additionally a series of duplicates. However, this seems to dampen the performance on subtask 1. Hence, informational value appears to be more important to subtask 1 than subtask 2. One limitation of these results is that only the ‘product-category’ was oversampled and not both. Judging from the increased performance due to oversampling one category, further performance increases are likely when oversampling for hazard-category too. Future studies need to re-assess the

	Alpha	Gamma	Subtask	F1 Hazard Type (ST1) F1 Hazard (ST2)	F1 Product Type (ST1) F1 Product (ST2)	F1 Averaged
Baseline	-	-	ST1	0.5860	0.5232	0.5546
			ST2	0.1454	0.0241	0.0847
Focal_1	1	2	ST1	0.7449	0.5953	0.6701
			ST2	0.1709	0.0291	0.1000
Focal_2	0.75	3	ST1	0.7086	0.5601	0.6343
			ST2	0.1739	0.0342	0.1040
Focal_3	0.75	4	ST1	0.6488	0.5804	0.6146
			ST2	0.1838	0.0340	0.1089
Focal_4	0.5	3	ST1	0.5975	0.5801	0.5888
			ST2	0.1696	0.0345	0.1021
Focal_5	0.65	3	ST1	0.7482	0.5429	0.6455
			ST2	0.1758	0.0332	0.1045

*highest values (compared to the same subtask) in bold

Table 5: Results for Focal Loss

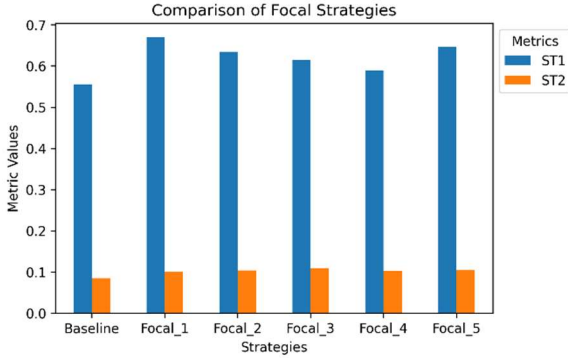


Fig. 3: Comparison of Focal Strategies

effectiveness of oversampling when done for both categories.

Contrary to downsampling and oversampling, implementing focal loss yielded better performance on both metrics in every configuration. However, adjusting the parameters alpha and gamma seems tricky for our dataset since the best performances on ST1 and ST2 were achieved by two different configurations respectively: Alpha = 1 and Gamma = 2 yielded the best result for ST1 while Alpha = 0.75 and Gamma = 4 the best result for ST2. A higher gamma, i.e. stronger focus on hard-to-classify examples, improves performance on ST2 but not ST1. For alpha, no consistent pattern with regard to ST1 or ST2 performance emerges..

6 Conclusion

This paper aimed to experimentally test different strategies of dealing with imbalanced data using both data-level methods which modify the data the classifier is trained on, and algorithm-level methods which tweak calculations within the training and evaluation phase.

After experimenting with downsampling, oversampling, and focal loss, I can conclude that training data size trumps informational value and skewness of the data, at least in this range of training data size. Future studies can try combining over and undersampling as well as more sophisticated data augmentation, e.g. by LLM generating new instances and classifying them.

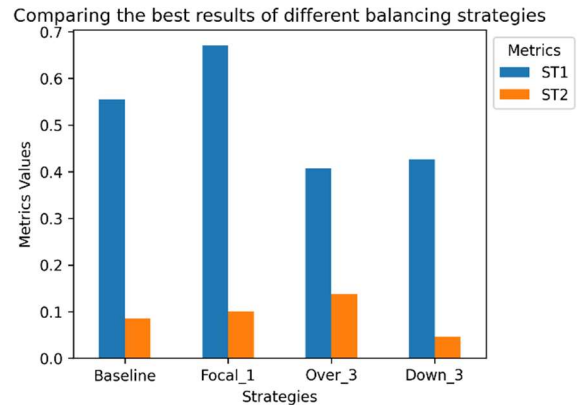


Fig. 4: Comparison of best solutions from the different balancing strategies

References

- Cai, Z., Li, Z., Liu, Y., Guo, L., & Song, Y. (2025). Ustnlp16 at SemEval-2025 Task 9: Improving Model Performance through Imbalance Handling and Focal Loss. arXiv preprint arXiv:2505.00021.
- Choi, H.-S., Jung, D., Kim, S., & Yoon, S. (2021). Imbalanced data classification via cooperative interaction between classifier and generator. *IEEE Transactions on Neural Networks and Learning Systems*, 33(8), 3343-3356.
- de Noordhout, C. M., Devleeschauwer, B., Angulo, F. J., Verbeke, G., Haagsma, J., Kirk, M., Havelaar, A., & Speybroeck, N. (2014). The global burden of listeriosis: a systematic review and meta-analysis. *The Lancet Infectious Diseases*, 14(11), 1073-1082.
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*,
- García, S., Zhang, Z.-L., Altalhi, A., Alshomrani, S., & Herrera, F. (2018). Dynamic ensemble selection for multi-class imbalanced datasets. *Information Sciences*, 445, 22-37.
- Goldberg, Y. (2017). *Neural network methods in natural language processing*. Morgan & Claypool Publishers.
- Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. *Proceedings of the IEEE international conference on computer vision*,
- Majowicz, S. E., Scallan, E., Jones-Bitton, A., Sargeant, J. M., Stapleton, J., Angulo, F. J., Yeung, D. H., & Kirk, M. D. (2014). Global incidence of human Shiga toxin-producing *Escherichia coli* infections and deaths: a systematic review and knowledge synthesis. *Foodborne pathogens and disease*, 11(6), 447-455.
- Padurariu, C., & Breaban, M. E. (2019). Dealing with data imbalance in text classification. *Procedia Computer Science*, 159, 736-745.
- Randl, K., Pavlopoulos, J., Henriksson, A., & Lindgren, T. (2024). CICLE: Conformal In-Context Learning for Largescale Multi-Class Food Risk Classification. arXiv preprint arXiv:2403.11904.
- Shaikh, S., Daudpota, S. M., Imran, A. S., & Kastrati, Z. (2021). Towards improved classification accuracy on highly imbalanced text dataset using deep neural language models. *Applied Sciences*, 11(2), 869.
- Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., & Le, Q. V. (2021). Finetuned language models are zero-shot learners. arXiv preprint arXiv:2109.01652.

Assessing the Impact of Data Augmentation on Food Hazard Detection

Iris Sørbye Bore
irissb@uio.no

Elisabeth Othamar
elisaot@uio.no

Mina Sheikhi
minashe@uio.no

Abstract

Contaminated food poses a substantial risk to human health. With access to a labeled data set of food recalls, we apply Machine Learning and Natural Language Processing methods to automatically detect such risks. By using language models in both classifier for enhanced performance and for data augmentations, we significantly improve the performance of this task. Finally, we discuss different prompting strategies and sampling criteria for the augmentation.

1 Introduction

1.1 Background and motivation

Food safety is a critical global concern, contaminated food pose a serious threat to human health, with millions of people affected by foodborne illnesses each year (Uyttendaele et al., 2016) (Majowicz et al., 2014). Natural Language Processing (NLP) solutions based on Machine learning (ML) and Large Language Models (LLMs) enable fast responses to new threats by generating warnings from publicly available texts on the internet (Maharana et al., 2019). However, these texts can be noisy and characterized by thousands of classes. Categorization of incidents into these classes is important to address the urgency of food risks. Automatic classification of the hazards involved in food recalls by leveraging NLP and LLMs can help estimating the severity of these food risks and lead to faster and more targeted measures. Publicly available official food recall announcements, and specifically their shorter titles, contain detailed information on food risks and the specific products involved and may be used as a basis for data-driven ML approaches (Randl et al., 2024). In this paper, we will focus on employing a machine learning technique called data augmentation (Bayer et al., 2022) to enhance the performance of a BERT-based language model (LM) (Devlin et al., 2018) to classify hazard and

product categories. As the data set is extremely unbalanced, we aim to explore methods to increase the sample number of the minority-classes within each categories to address the class imbalance. This will involve performing an in-context learning technique called few-shot prompting (Brown et al., 2020) on existing decoder-only language models called *SmolLM2-360M* (Allal et al., 2025) and *Qwen2/Qwen2.5-3B-Instruct* (Yang et al., 2024) to generate titles for the samples. We also explore the effect of using the BERT-based LM with four classification heads on the data by comparing the results with a simple baseline structure. We will work with the Food Hazard Detection Challenge from SemEval 2025 (Randl et al., 2025).

1.2 Related work

Several studies have addressed an automatic food hazard detection system, focusing on aspects such as developing an optimized classification model and data augmentation. There is not enough research specifically in the text-based food hazard detection area, thus related areas such as automatic food hazard detection on image data have also been addressed. In this section, we review the most relevant work related to these topics.

The paper by Jimenez-Moreno and Baquero (2023) presents the design of a convolutional neural network architecture oriented to the detection of food waste using machine vision systems, to generate a low, medium, or critical level alarm. An architecture based on four convolution layers is used, for which a database of 100 samples is prepared. The two classes, food and residue, are used as output and a 100 images from the database as input to the classification model. Different hyperparameters make up the final architecture, after the training process. By means of confusion matrix analysis, a 100% performance of the network is obtained, which delivers its output to a fuzzy system that,

depending on the duration of the detection time, generates the different alarm levels associated with the risk.

The paper by [Rasheed et al. \(2025\)](#) has addressed a problem more closely aligned with ours. The primary objective of this study is to demonstrate the impact of data augmentation using ChatGPT-4o-mini on food hazard and product analysis. The augmented data is generated using ChatGPT-4o-mini where the focus was on increasing the number of samples for classes with a small sample number. This data was subsequently used to train two large language models: RoBERTa-base and FlanT5-base. The models are evaluated on test sets. The results indicated that using augmented data helped improve model performance across all key metrics, including recall, F1 score, precision, and accuracy, compared to using only the provided dataset. Additionally, FlanT5 demonstrates superior performance both with and without augmentation compared to RoBERTa. Training time was also measured, revealing that RoBERTa required less computational time compared to FlanT5, indicating higher computational efficiency.

2 Method

2.1 Data

The dataset includes recall announcements, split to title and text, from official food agency websites and is manually labeled by food professionals ([Randl et al., 2024](#)). Each announcement is labeled in four groups of classes, describing hazards and products with two levels of granularity, as seen in Table 1.

Class	Description
Hazard	Fine-grained description of the hazards mentioned in the texts; 261 classes.
Hazard-category	Categorized version of the hazard classification task; 10 classes.
Product	Fine-grained description of the products mentioned in the texts; 1,256 classes.
Product-category	Categorized version of the product classification task; 22 classes.

Table 1: Description of classification tasks and their corresponding label spaces.

The training objective is to predict certain labels of the dataset samples given the recall title. The

training and validation set contains 55902 and 6215 samples respectively and the target labels are highly-skewed. According to the research introducing the data set, the fine-grained tasks, hazard and product, have very low support for many classes ([Randl et al., 2024](#)).

2.2 Baseline

Logistic regression over a Term Frequency - Inverse Document Frequency encoding is the best performing traditional classifier ([Randl et al., 2024](#)). Thus, a BoW classifier is utilized as our baseline model. First we transform the input to vectors using Term Frequency - Inverse Document Frequency (TF-IDF), and then send the vectors to a logistic regression (LR) classifier. This baseline is used for comparison with the BERT-based classifier.

The baseline for data augmentation is the performance of the BERT-based classifier on the original data set.

2.3 BERT-Based Classification Model

The classification is usually a two-step process with the first step mapping the text into a machine-readable embedding vector, and the second one involving the learning process. The LM classifier tokenizes the input and transforms them into an embedding by passing them to a pretrained BERT model, which in addition to the embeddings outputs a class token embedding.

The labels and the predictions have to be integer scalars. However, our dataset includes string target classes and to obtain the desired result, we map them to integer scalars.

To approach the problem using a language model (LM), we have developed a BERT-based model with two linear heads that each predicts the category type of hazard and product, sub-task 1 (ST1), and the exact hazard and product, sub-task 2 (ST2). The BERT-based model is a pre-trained language model from Huggingface *bert-base-uncased* ([Devlin et al., 2018](#)), and the second head, which predicts ST2, uses the predicted class of ST1 along with BERT’s special classification token for further prediction. In this way, the classifier head leverages the information available to predict additional specifications about the food related issues for ST2, which would be more efficient than predicting each subtask separately.

2.4 Data augmentation

Drawing inspiration from the current best performing classifier of the SemEval 2025 leaderboard (Randl et al., 2025), we decided to try to expand our dataset by creating synthetic data. We chose this approach not only because it showed promising performance, but also because we believe this is a method relevant to many different areas, not just food hazard detection, as machine learning projects often lack enough data or cannot use the real data due to privacy reasons. To augment the data set, we first balanced the data set in algorithm 1, and then generated titles as shown in algorithm 2.

Algorithm 1 This algorithm balances dataset groups by upsampling under-represented classes and optionally downsampling over-represented classes.

```

1: Let  $G \leftarrow$  groups of training data aggregated
   by hazard category and product category
2: Let  $T \leftarrow$  median group size in  $G$ 
3: Initialize  $S \leftarrow \emptyset$   $\triangleright$  final list of sampled data
4: for all group  $g \in G$  do
5:   if  $|g| < T$  then
6:      $g' \leftarrow$  Upsample( $g$ ) to size  $T$ 
7:   else
8:     if downsample is True then
9:       if  $|g| > 5T$  then
10:         $g' \leftarrow$  Sample( $g$ , size =  $5T$ )
11:      else
12:         $g' \leftarrow g$ 
13:      end if
14:    else
15:       $g' \leftarrow g$ 
16:    end if
17:  end if
18:  Append  $g'$  to  $S$ 
19: end for

```

2.4.1 Sampling with replacement

We chose to sample with replacement to keep the hierarchical relationship between categories and specific classes, and their distribution. This way we preserve some information of the relative frequency of the original distribution.

2.4.2 Sampling criterion

We defined the combination of product category and hazard category to be a minority if the number of instances were less than the median number of instances for each of the combinations. We later tested

if our classifier benefited from manually increasing the minority threshold (and thus the number of samples generated from those minorities).

2.4.3 Experimenting with down-sampling

As the data set is so highly skewed that a few groups dominate the data set, it is interesting to see if the classifier could benefit from not only upsampling the minority groups, but also down-sampling the majority groups. We defined the majority threshold to have a 5:1 relationship to the minority (threshold), so if a group were larger than five times the minority threshold, we only kept five times the minority threshold of those instances, as seen in algorithm 1, line 8. We chose that approach to minimize the loss of data, while at the same time making a cut in the largest groups.

2.4.4 Few-shot prompting

Algorithm 2 This algorithm prompts a pretrained language model to create new titles for duplicate examples sampled in 1, using the context of the duplicate sample.

```

1: Split sampled examples into:
    $U \leftarrow$  unique examples (originals)
    $D \leftarrow$  duplicated examples
2: for all example  $x$  in  $D$  do
3:    $x$ .prompt  $\leftarrow$  FormatPrompt( $x$ , context
   =  $U$ .hazard-category,  $U$ .product-category,
    $U$ .hazard,  $U$ .product, shots)
4:    $x$ .title  $\leftarrow$  CompletePrompt( $x$ .prompt)
5: end for

```

We used few-shot prompting to generate the titles. Previous research emphasized to select "good examples" for prompting (Randl et al., 2024), however, what is a good example? For our research, we decided to not manually select examples from our data set, but instead draw the demonstration examples randomly from the original data set to reflect the variation and noise in the data, and thus hope to create more realistic synthetic training examples for our classifier. Additionally, we experimented with drawing the demonstration examples randomly from a set with the same product category and hazard category as the synthetic row. This means that product and hazard may still vary, as one can see in figure 1. For a prompt example with a diverse demonstration example, see figure 4 in Appendix A. This way, we could test if the language model responded best to diverse demonstration examples

or examples belonging to the same categories as the ones the model should generate the titles from. It should be noted that the prompts were later pre-processed by built in methods to match the desired format of the corresponding pretrained language model.

```
Write a title for product recall.
Demonstration example:
Hazard category: biological
Product category: feed materials
Product: herbal premix
Hazard: listeria monocytogenes
Title: Dunnes Stores recalls Chicken, Bacon
& Sweetcorn Deli Filler due to possible
contamination with Listeria monocytogenes

Instruction:
Hazard category: biological
Product category: feed materials
Product: feed materials
Hazard: salmonella
Title:
```

Figure 1: Prompt example for one-shot, where the demonstration example is drawn from the same hazard category/product category group. For several shots, the template is the same, but with appended demonstration examples.

2.4.5 Pre-trained language models

We tested the models *SmolLM2-360M* (Allal et al., 2025) and *Qwen2/Qwen2.5-3B-Instruct* (Yang et al., 2024). The models were chosen to experiment with performance and computational cost, against the model’s complexity, with the first being a small language model and the other a large language model.

2.5 Evaluation

The evaluation metric we use is F1-macro for all the classes, averaged over the two label types in each sub-task. That is: the final ST1 score is the F1-macro average of the hazard category and the product category. For ST2 scores the procedure is the same, but for specific hazards and products.

This kind of metric is not the norm: Normally accuracy, precision, recall and F1-macro are reported separately for each classification task. However, in this experiment the different scores of hazards and product categories vs. their specific hazard and product classes are highly dependent. Our approach is done to make our predictions more explainable. In other words, a high ST2 score supports the reliability of the ST1 score, and vice versa, while a low ST2 score calls the ST1 score into question.

This was also done in the Food Hazard Detection Challenge from SemEval 2025 (Randl et al., 2025).

Additionally, we evaluate whether the generated titles are reasonable with respect to the instruction, by visually inspecting random examples in the synthetic data set.

3 Results and discussion

3.1 Baseline

The results of our baseline classifier are illustrated in Table 2, with a macro-F1 average of 0.50 for ST1 and 0.19 for ST2. Considering that the categories are highly skewed, with the two majority classes being almost 70% of the instances for hazard category and 40% for product category (see Figure 2, and Figure 5 in Appendix A), we suspect this score is not far from what you would get if you only predicted the majority classes for ST1. For ST2 the score is even worse, but one should keep in mind that there are a high number of classes to predict from, and that many of the classes only contains one instance. We will use these scores as a reference going forward.

ST1	ST2
0.50	0.19

Table 2: Macro-F1 scores on the validation set for the baseline classifier trained on the original data.

3.2 BERT-Based Classification Model

For optimization of the LM we finetune it on different configurations of the hyperparameters batch size and learning rate, and choose the best combination based on performance. Finetuning with the original data set yielded the results seen in table 3 on the validation data. The ST1 scores are similar for all hyperparameters, while the ST2 scores are not. There is a clear trend: With a small learning rate the model is not generalizing well for the diverse classes hazards and products. A higher learning rate yields the opposite effect. Additionally, a larger batch size leads to fewer updates of the model parameters, and in our experiment, poorer generalization. Thus, our best configuration of hyperparameters is batch size 80 and learning rate 1e-4.

Based on the scores from the baseline classifier and the BERT-Based classifier, we clearly see the BERT-Based model outperform the baseline. The model more than doubles the ST2 score and improves the ST1 score. This implies that the use of a

Batch size	Learning rate	ST1	ST2
80	3e-4	0.72	0.40
80	1e-4	0.73	0.41
80	3e-5	0.72	0.26
100	3e-4	0.71	0.41
100	1e-4	0.72	0.39
100	3e-5	0.72	0.24
120	3e-4	0.71	0.40
120	1e-4	0.71	0.37
120	3e-5	0.72	0.21

Table 3: Hyperparameters and their corresponding scores on the validation set for classifier task ST1 and ST2, while finetuning with original data. The optimal hyperparameter configuration is shown in bold.

language model is beneficial for these classification tasks.

3.3 Data augmentation

3.3.1 Initial sampling results

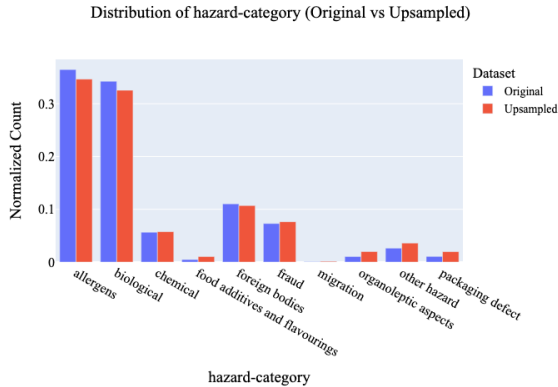


Figure 2: Distribution of hazard category

Our initial minority threshold, computed by taking the median of each combination of hazard and product categories, was seven. The resulting distribution is seen in Figure 2. We see that our method of upsampling the minority classes keeps the relationship between the hazard categories, while also increasing the number of examples in the smallest categories and thus slightly smoothing out the towering domination of the largest classes. However, the figure also shows that our augmented data is still skewed and may benefit from increasing the minority threshold manually or downsampling the majority classes, which we experiment with later.

3.3.2 Pre-trained language models

When we tested the models *SmolLM2-360M* (Al-lal et al., 2025) and *Qwen2/Qwen2.5-3B-Instruct* (Yang et al., 2024), it was clear by visual inspection that only the latter generated sufficient titles. Thus, moving forward only *Qwen2/Qwen2.5-3B-Instruct* was used to create synthetic data sets.

3.3.3 Different prompt templates

By visual inspection, we observe that the titles generated with demonstration examples belonging to the same categories as the ones the titles should be based on, are less creative and borrows more from the demonstration examples than the titles with more diverse demonstration examples. For instance, instead of generating brand names, the generated titles tend to borrow brand names from the demonstration examples, as seen in table 4.

Demonstrating Example

Ginco recalls its Chocolate Coated Honeycomb 100g due to undeclared allergens.

2015 - UPC Correction: David Trail Mix Sweet and Salty Voluntarily Recalled Due To Undeclared Dairy Allergen.

Generated Title

Ginco Recalls Honeycomb Products Due to Undeclared Milk Allergens.

2023 - Voluntary Recall of Certain Flavors Due to Potential Milk Allergen Contamination.

Table 4: Examples of titles generated with a one-shot demonstrating example of the same hazard and product categories

From table 5, we observe that the titles generated from the demonstration examples with random categories are more creative. A such example: Creating brand names inspired by its creator company Alibaba Cloud.

Product and Hazard Category in the demonstration examples had a surprising effect on the BERT-Based classifier. As mentioned, the titles generated with similar demonstration examples were less creative. Thus, when we evaluated the effect of similar combinations of product and hazard categories in a 1-shot setting, the BERT-Based model did not learn more from the augmented data, with ST1 score 0.72 and ST2 score 0.39, as shown in 6. This suggests that the increase in training data

Demonstrating Example
VICS Acquisition, LLC Recalls Chicken Products due to Misbranding and Undeclared Allergens.
Pinnacle Foods, Inc. Recalls Chili Products Due To Misbranding.
Generated Title
Alibaba Cloud Recall: Gluten-Containing Cakes Recalled Due to Allergen Hazards.
ABC Beverage Co. Recalls Certain Varieties of Wine Due to Presence of Yeasts.

Table 5: Examples of titles generated with diverse categories in demonstrating example in a one-shot setting.

is negated by the fact that it is too similar to the original data. Additionally, the classifier seemed to learn less regarding the categories judging by the decrease of ST1 by 0.01 from the baseline BERT model.

This decrease is also present in the classifier with the augmented data made with diverse demonstrating examples, as this model also yields a ST1 score of 0.72. However, the BERT-Based model generalizes slightly better than with original data for the specific products and hazards, see the ST2 score of 0.42 from table 6. This indicates that the augmented data is diverse enough for the BERT-Based model to learn more than with the original training data for product and hazard, and still generalize well for the validation data. Therefore, moving forward, the demonstration examples for generating more training data will be diverse.

Demonstration Example	ST1	ST2
With similar example	0.72	0.39
With diverse example	0.72	0.42

Table 6: Macro-F1 averages for the BERT-Based classifier with augmented data using 1-shot prompts and different types of demonstration examples.

Testing different number of shots yielded generated titles like those in table 7. The examples are quite similar in content, however, the two-shot setting seems to generate longer texts. Longer texts might not be beneficial in our classifier, as the validation set and production setting will classify with normal length titles.

The number of shots in the prompts can greatly impact the quality of the generated titles and consec-

One-shot
Alibaba Cloud Recall: Gluten-Containing Cakes Recalled Due to Allergen Hazards.
Herr Foods Inc. Announces Voluntary Recall of Cookies Due to Undeclared Eggs: Safety Alert.
Two-shot
Alibaba Cloud Recall: XYZ Bakery Inc. Recalls Cakes Due to Gluten Contamination.
Herr Foods Inc. Announces Voluntary Recall of Sweet Treats Brand of 3 oz. Chocolate Chip Cookies Due to Undeclared Eggs: Allergy Concern for Those With Egg Allergy.

Table 7: Examples of titles generated in a one- and two-shot setting.

Number of shots	ST1	ST2
1	0.72	0.42
2	0.71	0.39
3	0.71	0.39

Table 8: Scores for the BERT-Based classifier with augmented data using different numbers of shots in the prompts. The best scores are shown in bold.

utively the performance of the BERT-Based model. From table 8 we see that the impact of the generated titles is best in the one-shot setting. This is in line with what the examples seen in table 7 indicated, and supports our interpretation that longer texts may not benefit our classifier. In this experiment we clearly see this trend, more complicated input leads to poorer predictions. Thus, a one-shot setting is most beneficial for our classification task.

3.3.4 Adding Downsampling

The result of adding downsampling to our classifier can be seen in Figure 3. The figure shows that downsampling gives a smoother distribution. However, this comes at the cost of losing more than half of our training examples, effectively reducing the training set from 5082 samples to 2274 samples.

Downsampling of the majority classes can have different effects during training. On the one hand, it could yield the model more room to generalize better for the minority classes, on the other hand, less data could lead to the model underfitting and not generalize well for any classes. The results of downsampling are seen in table 9. We see that the

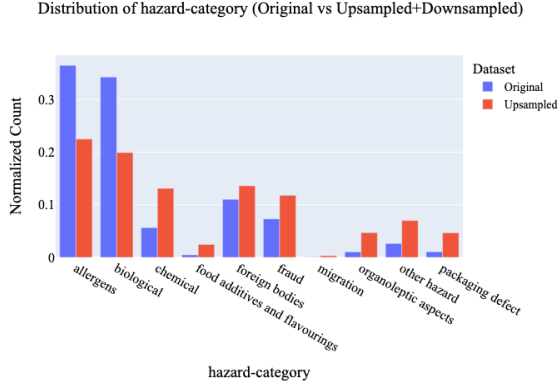


Figure 3: Comparison of distribution of hazard category for original data set and augmented data set with upsampling and downsampling

decrease in training data is too large and leads to underfitting.

Downsampling	ST1	ST2
With downsampling	0.68	0.28
Without downsampling	0.72	0.42

Table 9: Scores for the BERT-Based classifier with augmented data using 1-shot prompts and downsampling.

3.3.5 Experimenting with sampling criterion

We experimented with our threshold and the results are in table 10. The best number of generated examples for the classification is 10, and compared to the scores with the original data both ST1 and ST2 increase by 0.01. The minority threshold

Minority threshold	ST1	ST2
7	0.72	0.42
9	0.74	0.40
10	0.74	0.42
11	0.71	0.41
12	0.73	0.41

Table 10: Scores for the BERT-Based classifier using augmented data with different number of samples per category combination. The best scores are shown in bold.

and samples drawn from the minority needs to be balanced with what is actually beneficial for the classification. With our implementation, we could also increase the amount of drawn samples for the minority classes, without adjusting the minority threshold. However, at some point the augmented data no longer yields any more useful information

for the BERT-Based classifier, and might even lead to confusion and poorer performance.

3.4 Final Results

From our experiments, the best classifier was the BERT-based classifier trained on the augmented data set where we drew 10 samples from the minority combinations of hazard and product category, and generated titles for the rows not in the original data set using a one-shot template with diverse examples.

Testing this classifier with our independent test data set gave us the final results seen in table 11. We also tested the BERT-classifier trained on the original data and our baseline on the same dataset. The results show that these classification tasks benefited from a BERT-based model such as ours, and that the model benefited from augmented data.

Model	ST1	ST2
Baseline	0.48	0.14
BERT-Original	0.67	0.36
BERT-Augmented	0.71	0.36

Table 11: Test results of our classifiers.

The table 12 shows the results of [Rasheed et al. \(2025\)](#) against ours. They reported F1-macro scores for only product and hazard categories, which we averaged. Their results were better than ours, and could be due to their augmentation process leading to better generated data, to their classifier or that they use both the title and the text from the recall statements as input for the models.

Model	ST1	ST2
BERT (our)	0.71	0.36
RoBERTa	0.77	-
Flan-T5	0.78	-

Table 12: Scores of Classifiers on the test data, with models trained on augmented data. Ours and ([Rasheed et al., 2025](#))’s.

A selection of results for this challenge in the SemEval 2025 ([Randl et al., 2025](#)) and our results can be found in table 13. This indicates that our solution has room for improvement regarding performance. Additionally, we seem to have a disproportionately poor ST2 score given our ST1 score, which could be due to our synthetic titles leading to poorer context for predicting the product and hazard classes.

Lastly, we check the credibility of our final results

Group	title	text	ST1	ST2
SRCB	x	x	0.80	0.55
HU	x	x	0.79	0.51
VerbaNexAI	x	-	0.52	0.32
BrightCookies	-	x	0.76	0.45
Our Solution	x	-	0.71	0.36

Table 13: Scores from different solutions to SemEval 2025 (Randl et al., 2025) and our solution.

by computing test results, using our best hyperparameter configuration and training data set, with three different initializations of the BERT-based model. Table 14 shows the results.

Runs	ST1	ST2
1	0.71	0.36
2	0.71	0.37
3	0.70	0.37
Average	0.707	0.367

Table 14: Scores, and the average, of the best BERT-based classifier with different initializations, trained with the same augmented data.

4 Conclusion and future work

By introducing a language model for the task of detecting food related issues from food recall statements, we were able to improve the results from our baseline. Further, we showed that the classifier may benefit from data augmentation, but that this is not straightforward for multi-label and multi-class textual data. While the results are promising with the method we developed, which increased the classifier performance slightly, this field needs further work on establishing more balanced methods suitable for our use case. However, we conclude that language models were a helpful tool in generating the food recall titles for augmenting the data set. Finally, we showed that the way we prompted the language model to generate titles impacted the classifier performance. This supported our impression that the language models of today are sensitive to both the input examples and formatting of them, and that one as practitioners always benefits from experimenting with this.

For future work we would like to put more effort into experimenting with the sampling criterion and number of samples drawn. The downsampling implementation loses too much data, and the up-sampling has potential to be more aggressive in

balancing the distribution. Additionally, we choose not to manually select top-notch demonstrating examples, however, doing this might improve the augmented data.

In the initial phase of our research, we researched other methods for data augmentation. SMOTE (Chawla et al., 2002) is such a procedure and we investigated if there were previous works that implemented the SMOTE sampling method for a use case similar to ours. As that was not the case, we implemented a simplified implementation inspired by SMOTE, but it would be interesting to explore the potential of interpolating example titles from samples in the data.

Another alternative could be to back-translate the titles to generate similar versions of them. This approach may benefit from the BERT-classifier’s possible internal knowledge of the connection between brand names and product type, which we lose when we generate the brand names. Further, it would be interesting to apply supervised finetuning of our language model, like LoRA, to see if this gives more representative generated titles.

Lastly, we have done no experimentation regarding the choice of input. We decided on only using *title*, on the basis of reducing computational cost. However, experiments using also *text*, and comparisons of inference time would be beneficial for making a more informed decision.

Limitations

We have used the best hyperparameters from the classifier trained on the original set data when running the model with augmented data. It might be beneficial with new hyperparameter tuning experiments specifically for the augmented data.

As the focus of this research paper was to experiment with the effect of data augmentation on this task, we only compared one classifier against the TF-IDF-LR baseline model. However, it would be interesting to set the best performing model, an SVM model (Randl et al., 2024), as the baseline classifier, and further experiment with different model setups or other language models. This would increase the possibility of developing an optimal classifier.

Ethical considerations for this experiment is that the use of a language model to generate more text might lead to biases and inaccuracies in the titles, which is considered as harmful. This is due to the learned biases and inaccuracy in the language model

from its training corpora. We have not assessed the model used for text generation with regards to fairness. The titles generated are only developed for internal use and will not be reflected in the final production set up.

Our improvement in classification by augmenting the data set was only increased by 0.01 F1 macro scores on the validation set. While augmenting the data set by generating titles slightly improves performance, the environmental and computational costs raises concerns about whether it is worth the environmental impact of using a large language model for this task.

Acknowledgments

We would like to thank Sondre Wold for helpful guidance with this research paper.

References

- Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, and 3 others. 2025. *Smolm2: When smol goes big – data-centric training of a small language model*. *Preprint*, arXiv:2502.02737.
- Markus Bayer, Marc-André Kaufhold, and Christian Reuter. 2022. A survey on data augmentation for text classification. *ACM Computing Surveys*, 55(7):1–39.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. *Language models are few-shot learners*. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.
- N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. 2002. *Smote: Synthetic minority over-sampling technique*. *Journal of Artificial Intelligence Research*, 16:321–357.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. *BERT: pre-training of deep bidirectional transformers for language understanding*. *CoRR*, abs/1810.04805.
- Robinson Jimenez-Moreno and Javier Eduardo Martínez Baquero. 2023. Automatic food bio-hazard detection system. *International Journal of Electrical and Computer Engineering (IJECE)*, 13(3):2652–2659.
- Adyasha Maharana, Kunlin Cai, Joseph Hellerstein, Yulin Hswen, Michael Munsell, Valentina Staneva, Miki Verma, Cynthia Vint, Derry Wijaya, and Elaine O Nsoesie. 2019. Detecting reports of unsafe foods in consumer product reviews. *JAMIA open*, 2(3):330–338.
- Shannon E Majowicz, Elaine Scallan, Andria Jones-Bitton, Jan M Sargeant, Jackie Stapleton, Frederick J Angulo, Derrick H Yeung, and Martyn D Kirk. 2014. Global incidence of human shiga toxin-producing escherichia coli infections and deaths: a systematic review and knowledge synthesis. *Foodborne pathogens and disease*, 11(6):447–455.
- Korbinian Randl, John Pavlopoulos, Aron Henriksson, and Tony Lindgren. 2024. *CICLe: Conformal in-context learning for largescale multi-class food risk classification*. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7695–7715, Bangkok, Thailand. Association for Computational Linguistics.
- Korbinian Randl, John Pavlopoulos, Aron Henriksson, Tony Lindgren, Juli Bakagianni, and Giannis Stoitsis. 2025. Food hazard detection semeval 2025. <https://github.com/food-hazard-detection-semeval-2025/food-hazard-detection-semeval-2025.github.io>. Accessed: 2025-05-19.
- Areeg Fahad Rasheed, M. Zarkoosh, Shimam Amer Chasib, and Safa F. Abbas. 2025. *Data augmentation to improve large language models in food hazard and product detection*. *Preprint*, arXiv:2502.08687.
- Mieke Uyttendaele, Eelco Franz, and Oliver Schlüter. 2016. Food safety, a global challenge.
- An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 40 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

A Appendix

Write a title for product recall.
 Demonstration example:
 Hazard category: biological
 Product category: ices and desserts
 Product: ice cream
 Hazard: salmonella
 Title: Certain Nestle brand Chocolate Peanut Butter Ice Cream may contain Salmonella

Instruction:
 Hazard category: allergens
 Product category: cereals and bakery products
 Product: cakes
 Hazard: cereals containing gluten and products thereof
 Title:

Figure 4: Prompt example from diverse hazard/product group

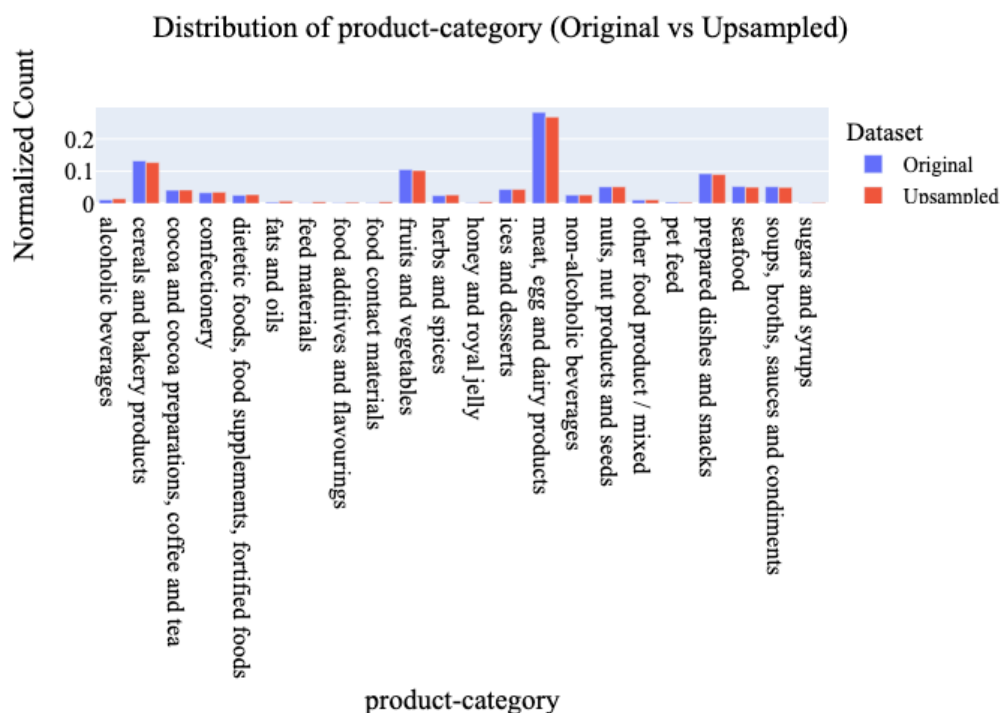


Figure 5: Comparison of distribution of product category for original data set and augmented data set with upsampling

Enhancing Food Hazard Detection with BERT-Based Text Classification and Synthetic Data Augmentation

Carl-Henrik Lien
University of Oslo
carlhehl@uio.no

Gyula Szente
University of Oslo
gyulas@uio.no

Katarzyna Segiet
University of Oslo
katarzse@uio.no

Abstract

This paper explores the use of synthetic training data generated by a large language model (GPT-4.1-mini) to improve text classification performance in *SemEval 2025 Task 9: The Food Hazard Detection Challenge*. The task involves classifying food recall statements across four subtasks: identifying the specific hazard, hazard category, product, and product category. We adopt a BERT-based classifier as our baseline and address data imbalance by augmenting the training set with synthetic examples. Performance is evaluated using two aggregate F1 metrics: ST1 (covering hazard and product categories) and ST2 (covering specific hazard and product labels). Augmenting the original data with LLM-generated samples improves the average F1-score from 0.76 to 0.80 for ST1 and from 0.41 to 0.46 for ST2, demonstrating the effectiveness of synthetic data in this domain-specific classification task.

1 Introduction

Food safety is an important public health issue, in 2010, food-borne illnesses lead to 420,000 deaths globally (World Health Organization, 2015). Regulatory agencies publish recall notifications to alert the public about contaminated, misbranded, or mislabeled products. These texts typically include structured identifiers (e.g., product names, hazard types) and semi-structured text that describes the problem in detail. Automatically classifying such notices into predefined categories can enhance surveillance, response time, and public accessibility.

In this paper, we make the classifier for food-hazard detection using data from the SemEval2025 Task: The Food Hazard Detection Challenge (Randl et al., 2024). The classifier is based on a BERT-model to automate the labeling of food hazard notifications. We focus on two tasks:

- **ST1:** Predicting general product and hazard categories.
- **ST2:** Predicting specific hazard and product labels, evaluated with conditional scoring.

The main challenge with the available dataset is the significant class imbalance among the target labels. To mitigate this, we use GPT-4.1-mini to generate entirely new training examples, specifically targeting combinations of target labels that are underrepresented in the original training set. To address this, we explore the following research question: *Can synthetic data generated by large language models improve classification performance and alleviate class imbalance in food hazard detection?* We then evaluate whether incorporating this synthetic data improves the overall performance of the model.

Using the synthetic data improves the performance our model. We find that the average F1-score increases from 0.76 to 0.8 for ST1 classification and from 0.41 to 0.46 for ST2 classification.

2 Related Work

2.1 Food Hazard Detection with NLP

Various NLP methods have been applied to the Food Hazard Detection Challenge, with top-performing systems achieving F1-scores of 0.82 for ST1 and 0.55 for ST2 (SemEval, 2025). Some approaches have demonstrated improved performance through data augmentation to address class imbalance. Papadopoulou et al. (2025) experimented with several non-LLM augmentation techniques and observed slight but inconsistent improvements on a BERT-based classifier; these improvements were statistically significant only for minority classes. In contrast, Rasheed et al. (2025) used ChatGPT-4o-mini to generate synthetic training examples and reported more substantial F1-score gains for ST1, hazard-category and product-

category prediction. However, their work did not evaluate performance on the more granular ST2 subtask involving specific hazard and product labels.

2.2 BERT based classifiers

Since its release, BERT (Devlin et al., 2019) has significantly influenced the field of NLP, particularly in classification tasks. Its ability to model bidirectional context has made it especially effective for understanding sentence-level semantics. Researchers have applied BERT to a wide range of classification problems, including sentiment analysis, spam detection, and topic classification. Variants such as RoBERTa (Liu et al., 2019) have further refined the architecture, offering trade-offs between performance and computational efficiency.

2.3 Synthetic Data Augmentation in Text Classification

Low-resource text classification has long benefited from data augmentation techniques like back-translation (Shleifer, 2019), paraphrasing (Marivate and Sefara, 2020), and token-level noising (Wei and Zou, 2019) to expand training corpora. When combined with machine learning classifiers, such methods help combat data scarcity and improve model performance.

Recently, the emergence of powerful large language models (LLMs) has enabled a new paradigm of LLM-generated synthetic data. These models can produce high-quality, human-like text on demand (OpenAI et al., 2024), making them attractive “data generators” for NLP research. LLM-based augmentation is often found to outperform standard augmentation methods in boosting classifier performance (Li et al., 2023).

Several studies from 2022–2024 leverage GPT-3+ models to augment training data, reporting improved outcomes across a range of text classification tasks. For example, Ding et al. (2023) trained classifiers on GPT-3 annotated data, where GPT-3 labels unlabeled examples, and found performance comparable to or better than training on human-labeled data under limited-budget settings. Samuel et al. (2024) achieved significant improvements in question-answering performance by augmenting the training set with synthetic data generated by GPT-4. Hartvigsen et al. (2022) developed the ToxiGen dataset, consisting of 274,000 machine-generated toxic and benign statements about minority groups, and demonstrated that the synthetic

data significantly improved classifier performance. Similarly, Han et al. (2024) employed ChatGPT to generate synthetic data for debiasing large language models, showing that targeted augmentation strategies can improve fairness while maintaining classification performance.

These works collectively demonstrate the growing utility of synthetic data from LLMs in enhancing text classification, particularly in domains with limited labeled resources. Our study extends this line of work to the food safety domain.

3 Methodology

3.1 BERT Classifier

Our baseline model for text classification is based on a pretrained transformer encoder. Specifically, we use RoBERTa-base, a variant of BERT with improved training strategies and performance on downstream NLP tasks (Liu et al., 2019). RoBERTa consists of 12 transformer layers with a hidden size of 768 and 12 attention heads, totaling approximately 125 million parameters.

We fine-tune the model by adding a linear classification head on top of the [CLS] token representation, which is standard practice for sentence-level classification (Sun et al., 2020). The classification head outputs logits over the set of target classes, and the entire model is trained end-to-end.

The input to the model is a concatenation of structured text features. Each training example is created by merging the title and text columns from the dataset, and augmenting the resulting text with structured metadata. This metadata includes information about detected product names and hazard types, extracted from the training data using simple string matching. These structured cues are appended as a prefix to each input instance, in the form: FOUND PRODUCTS: ...; FOUND HAZARDS: ..., followed by the concatenated title and text. This additional context helps the model disambiguate relevant semantic signals, particularly when product- or hazard-specific phrases are indicative of certain labels.

We tokenize input sequences using the AutoTokenizer associated with RoBERTa-base, truncating or padding to a maximum length of 512 tokens. The model is trained using cross-entropy loss, optimized with the AdamW optimizer and a linear learning rate scheduler with warmup. Training is performed over multiple epochs, with early stopping based on macro-averaged F1 score on a valida-

tion set. During each training step, batches are tokenized on-the-fly using a custom collate function to handle dynamic padding and attention masking.

The label set is encoded using LabelEncoder, with an additional *unknown* class appended to handle unseen classes during evaluation. The number of output logits in the final classification layer matches the number of encoded labels.

Overall, this setup serves as our baseline classifier for evaluating the impact of synthetic data augmentation on downstream classification performance.

3.2 Real Dataset

The original dataset comprises 5082 food recall records manually labeled with four fields: *hazard*, *product*, *hazard-category*, and *product-category*. Records are drawn from public health agencies (e.g., USDA, FSANZ) and cover a wide geographic and temporal range. Each record includes timestamps and geolocation metadata, enriching the dataset’s contextual relevance.

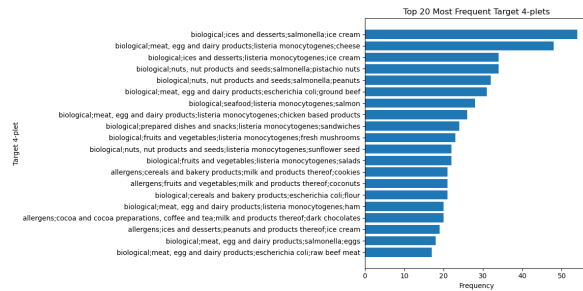


Figure 1: Top 20 most frequent target 4-plets in the real dataset. Frequencies remain low even for common combinations.

3.3 Motivation for Synthetic Data Augmentation

An analysis of the training dataset reveals substantial label sparsity and skew. Among the 5,082 total recall entries, there are 2,762 unique combinations of the four target labels (*hazard-category*, *product-category*, *hazard*, *product*). As visualized in Figure 2, 1991 of these 4-plets appear only once (over 72%), and fewer than 5% appear more than five times. This heavy-tailed distribution indicates a lack of redundancy in the dataset, which is problematic for supervised learning tasks — especially when learning fine-grained or rare hazard-product relations.

To reduce this sparsity, we augmented the training set with synthetic samples generated via few-

shot prompting using OpenAI’s language model. These synthetic entries were designed to mirror the structure and diversity of the real data while introducing additional examples for underrepresented combinations. As shown in Figure 3, after augmentation, the majority of 4-plets appear at least three times, reflecting a significant improvement in class support. The use of synthetic data was motivated not only by the limited dataset size but also by the high variance across target labels, making it a valuable tool to improve model generalization.

3.4 Synthetic Data Generation

To improve the diversity and robustness of our training data, we generated synthetic examples using a decoder-only large language model (LLM). As previously mentioned, decoder-based LLMs have demonstrated superior performance across a wide range of natural language processing tasks, particularly in settings that involve realistic text generation and instruction following. This makes them especially well-suited for creating plausible synthetic examples of food recall statements.

We selected OpenAI’s *gpt-4.1-mini* as our generation engine. The GPT-4.1 series has shown strong performance across multiple benchmarks and tasks, offering an appealing balance between cost-efficiency and model capabilities (OpenAI, 2024). For example, in the MMLU benchmark, designed to assess LLM performance across 57 subject areas, *gpt-4.1-mini* achieves a score of 87.5%, significantly outperforming earlier models such as GPT-3 (43.9%) (Hendrycks et al., 2021). For comparison, human domain experts are estimated to achieve 89.8% on the same benchmark. As of mid-2024, models such as GPT-4o, Claude 3.5 Sonnet, and LLaMA 3.1 405B consistently reach performance levels around 88%, placing *gpt-4.1-mini* competitively close to state-of-the-art. We were able to generate 4,373 new synthetic examples for under \$3 USD.

3.4.1 Synthetic Data Creation Procedure

To guide the LLM in generating high-quality, label-consistent synthetic data, we crafted a prompt template based on best practices in few-shot prompting (Sahoo et al., 2024). Each prompt includes one example that shares the same labels as the target output, as well as one randomly selected example from the training set. These examples serve as in-context demonstrations. Few-shot prompting has been shown to significantly enhance LLM output

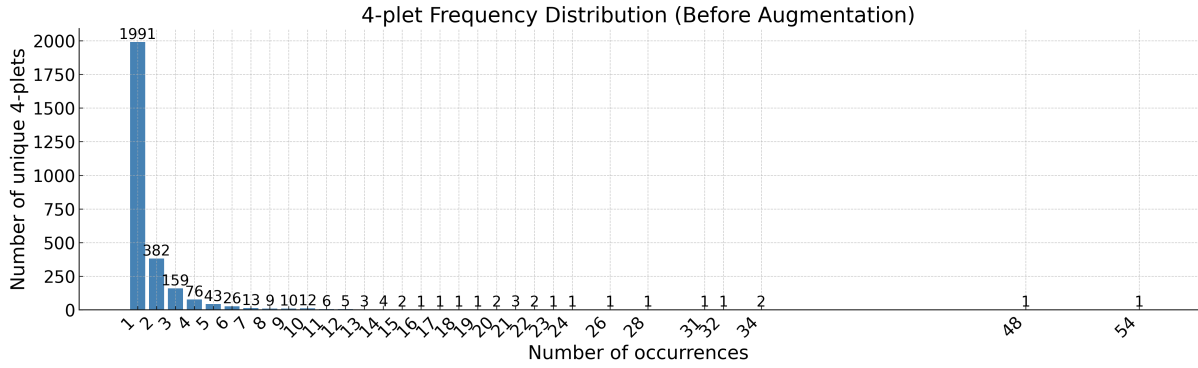


Figure 2: 4-plet Frequency Distribution (Before Augmentation). Most combinations appear only once.

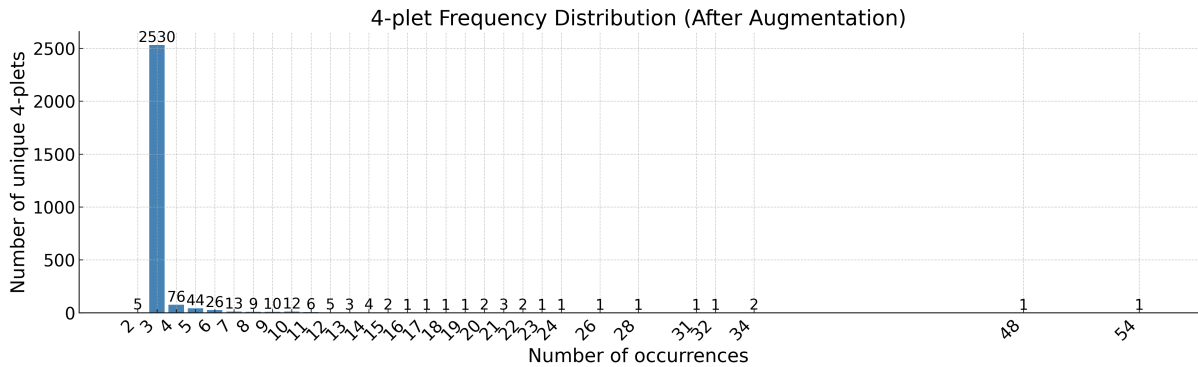


Figure 3: 4-plet Frequency Distribution (After Augmentation). Most underrepresented combinations now appear at least three times.

alignment with task requirements, particularly in text generation tasks (Brown et al., 2020).

The prompt explicitly provides the model with the target values for all relevant classification labels: *hazard-category*, *product-category*, *hazard*, and *product*. The model is then asked to generate a realistic and structured food recall statement, consisting of a title and detailed text, that aligns with those labels. This ensures the resulting data is not only semantically coherent but also label-accurate for supervised training (see Appendix A).

3.5 Analysis of the Dataset after Augmentation

The addition of synthetic data led to a marked improvement in class balance across both hazard and product categories. As shown in Figures 4 and 5, the augmentation strategy did not attempt to flatten the distribution entirely. Instead, it enriched underrepresented categories while maintaining the overall distributional shape of the real data. This targeted growth is particularly evident in the hazard categories such as *fraud*, *chemical*, and *organoleptic aspects*, which saw increases of over 100%.

Similarly, rare product categories like *honey and royal jelly*, *pet feed*, and *sugars and syrups* experienced substantial growth. This is further quantified in Tables 1 and 2, which show the relative percentage increase in entries per hazard and product category, respectively. By addressing class imbalance without artificially equalizing all categories, the augmentation approach improved data diversity and coverage in a way that is both realistic and beneficial for generalization. This is expected to support better classifier performance on previously underrepresented classes, as confirmed by the improved ST2 scores in Section 4.

3.5.1 Motivation for augmentation decisions

To combat the skewed label distribution in the training data, we specifically targeted 4-plet combinations (comprising product, product category, hazard, and hazard category) that appeared only once or twice in the original training set. We generated synthetic examples of these underrepresented 4-plets until each was represented at least three

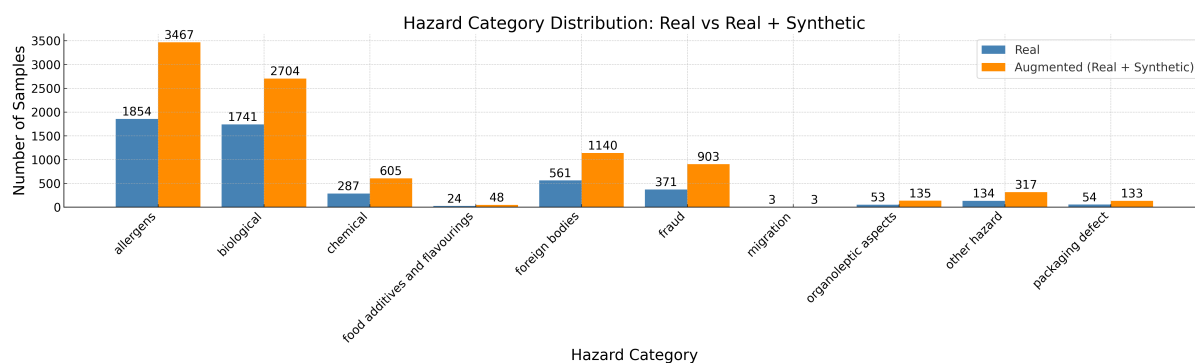


Figure 4: Class distribution comparison of hazard categories before and after augmentation.

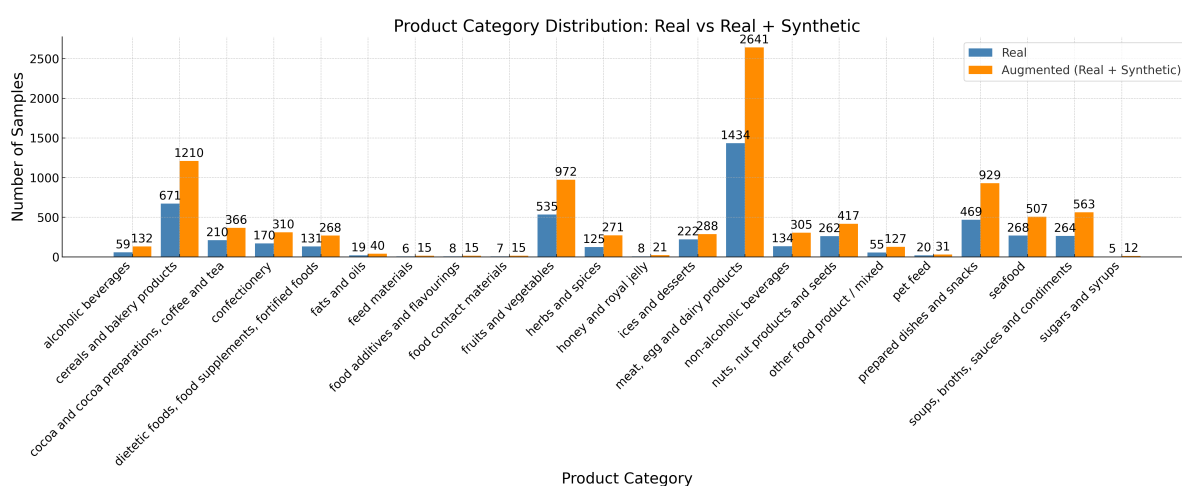


Figure 5: Class distribution comparison of product categories before and after augmentation.

Table 1: Percentage Increase by Hazard Category

	Augmentation %
allergens	87.0
biological	55.3
chemical	110.8
food additives and flavourings	100.0
foreign bodies	103.2
fraud	143.4
migration	0.0
organoleptic aspects	154.7
other hazard	136.6
packaging defect	146.3

Table 2: Percentage Increase by Product Category

	Augmentation %
alcoholic beverages	123.7
cereals and bakery products	80.3
cocoa and cocoa preparations, coffee and tea	74.3
confectionery	82.4
dietetic foods, food supplements, fortified foods	104.6
fats and oils	110.5
feed materials	150.0
food additives and flavourings	87.5
food contact materials	114.3
fruits and vegetables	81.7
herbs and spices	116.8
honey and royal jelly	162.5
ices and desserts	29.7
meat, egg and dairy products	84.2
non-alcoholic beverages	127.6
nuts, nut products and seeds	59.2
other food product / mixed	130.9
pet feed	55.0
prepared dishes and snacks	98.1
seafood	89.2
soups, broths, sauces and condiments	113.3
sugars and syrups	140.0

times¹ in the final dataset. This augmentation increased the size of the training set from 5,082 to 9,455 examples. We chose 4-plets as the basis for generation to ensure semantic consistency and to avoid producing unrealistic combinations, such as mismatched products and product categories. The effect of this augmentation is illustrated in Figure 3, which shows the distribution of 4-plet frequencies after data generation.

3.6 Dataset Variants

To evaluate the effect of synthetic data, we constructed five training configurations:

- **Real Only:** A dataset consisting of all 5082 real samples.
- **50/50 Mix:** A dataset of the same size as the real dataset, composed of 50% real and 50% synthetic samples.
- **All Real + Partial Synthetic:** 5082 real samples combined with 2541 synthetic samples, resulting in a dataset where exactly two-thirds of the entries are real and one-third are synthetic.
- **Full Real + Full Synthetic:** A combined dataset containing all real and all synthetic entries.
- **Synthetic Only:** A dataset composed entirely of synthetic samples, with no real data included.

All configurations share the same validation and test sets derived from real data.

4 Results and Discussion

4.1 Macro-Level Metrics: ST1 and ST2

We evaluated the model’s performance under varying training data conditions, detailed in Table 3. These results show that while the overall trends (e.g., real + synthetic beats synthetic-only) are consistent, the scores can vary across different seeds.

4.2 Fine-Grained Label Scores

Macro F1 scores for individual label types are presented in Table 4. These results indicate improvements in F1 scores for specific hazards and products alongside general categories with the inclusion of synthetic data.

¹Due to formatting errors in some responses generated by GPT-4.1-mini, five 4-plets occur only twice in the final combined training set.

Configuration	Seed	ST1	ST2
100% Real	Seed 1	0.7158	0.4155
100% Real	Seed 2	0.7999	0.4022
50% Real + 50% Synth	Seed 1	0.7309	0.3861
50% Real + 50% Synth	Seed 2	0.7787	0.4326
100% Real + 50% Synth	Seed 1	0.7734	0.4215
100% Real + 50% Synth	Seed 2	0.7756	0.4759
100% Real + 100% Synth	Seed 1	0.7809	0.4314
100% Real + 100% Synth	Seed 2	0.8106	0.4975
100% Synth	Seed 1	0.6522	0.3102
100% Synth	Seed 2	0.6468	0.3448

Table 3: ST1 and ST2 macro F1 scores across different training configurations and random seeds.

4.3 Interpreting Label-Level Scores

The individual F1 scores offer insight into the model’s strengths and weaknesses across different label types. Hazard and product labels, which require fine-grained distinctions (e.g., *salmonella*, *ice cream*), tend to yield lower F1 scores due to greater variability and complexity in naming. In contrast, broader categories like hazard type (e.g., *biological*) and product category (e.g., *meat*, *egg and dairy products*) are easier to classify, resulting in consistently higher scores. While synthetic data helped improve recognition of both hazards and products, gains were especially noticeable in general category prediction, where label consistency provided stronger learning signals.

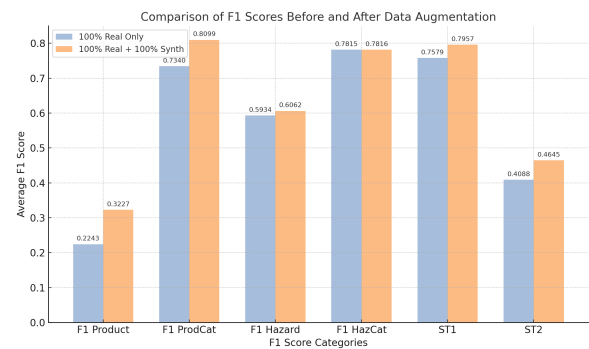


Figure 6: Average F1-Scores across Two Seeds

4.4 Impact of Seed Variation

Seed variation introduced noticeable fluctuations in model performance, highlighting the sensitivity of transformer-based models to initialization. For example, the ST1 score for the 100% real data configuration increased from 0.7158 to 0.7999 between two seeds, while other setups showed smaller gains or slight drops. This underlines the importance of testing across multiple seeds to ensure stable and

Config	Product	ProdCat	Hazard	HazCat
100% Real (Seed 1)	0.2218	0.6622	0.6092	0.7693
100% Real (Seed 2)	0.2268	0.8059	0.5776	0.7938
50% Real + 50% Synth (Seed 1)	0.2021	0.7555	0.5701	0.7064
50% Real + 50% Synth (Seed 2)	0.2508	0.7759	0.6144	0.7816
100% Real + 50% Synth (Seed 1)	0.2479	0.7875	0.5950	0.7593
100% Real + 50% Synth (Seed 2)	0.3208	0.7946	0.6311	0.7567
100% Real + 100% Synth (Seed 1)	0.2842	0.8061	0.5787	0.7558
100% Real + 100% Synth (Seed 2)	0.3612	0.8137	0.6337	0.8074
100% Synth (Seed 1)	0.2129	0.6428	0.4075	0.6616
100% Synth (Seed 2)	0.2368	0.6803	0.4527	0.6134

Table 4: F1 scores for individual label types across training data variants and seed settings.

reliable results.

4.5 Limitations of Synthetic-Only Training

The model trained solely on synthetic data performs considerably worse across all metrics, especially in ST2 (0.3102) and hazard F1 (0.4075). While ST1 remained above 0.65, this setup lacked the precision and domain specificity inherent in authentic records. This finding underscores the limitations of synthetic data when used in isolation, indicating that while beneficial for category generalization, it fails to replicate the intricacies found in real-world supervision.

4.6 Discussion and Implications

Overall, synthetic data acts as a useful complement to real examples. Our experiments show that by augmenting the training set with synthetic entries, we were able to consistently improve model performance across both general and fine-grained classification tasks. In particular, ST2 scores and product-level F1 metrics benefited the most, suggesting that synthetic examples helped the model generalize to more diverse or underrepresented label combinations.

Interestingly, the best performance was achieved

by using the full set of synthetic data alongside the real dataset, indicating that the model benefited from maximum exposure to varied examples. However, relying on generated data also comes with certain risks. Although most synthetic samples matched the intended labels and format, there is no guarantee that every output from the language model is perfectly aligned with real-world label distributions. Some generated examples may contain mislabeled, incoherent, or unrealistic content, which could potentially introduce noise into training.

For this reason, future work may benefit from incorporating a filtering or scoring step to assess the quality of synthetic outputs before adding them to the training set. For example, a small validation model, heuristic rules, or human-in-the-loop review could be used to exclude problematic examples. This additional step could help improve label consistency, reduce noise, and further boost the model’s generalization and robustness.

The conditional ST2 scoring also highlights an important interdependency between hazard and product predictions. Future work may consider joint or hierarchical modeling approaches to better capture such dependencies and improve overall label accuracy.

5 Conclusion

This work demonstrates that large language models can produce high-quality synthetic data that improves food hazard classification, particularly when paired with real examples. The most robust models emerged from combined real and synthetic training, with improvements in both general (ST1) and fine-grained (ST2) prediction.

Notably, the inclusion of synthetic data led to measurable gains in F1 scores across nearly all label types, especially in product classification, which is typically the most difficult. This supports the value of using generated data for low-resource or highly variable domains.

6 Limitations and Future Work

Although our experiments showed good results, there are some important limitations to keep in mind.

First, the scientific novelty of this work is limited by its reliance on a single, proprietary LLM — GPT-4.1-mini — for generating synthetic training data. While this model proved effective in boosting

classification performance, using a closed-source model restricts reproducibility and limits insight into the underlying generation mechanisms. Future work could explore the applicability of other LLMs or settings where commercial APIs are not accessible.

Second, we did not perform a comparative evaluation between proprietary and open-source LLMs (e.g., LLaMA, Mistral, or Falcon). Conducting such comparisons could be valuable for assessing trade-offs in data quality and classifier performance using more accessible, community-supported models. Investigating how open-source LLMs perform for domain-specific data generation remains an important direction for future research.

References

- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [Bert: Pre-training of deep bidirectional transformers for language understanding](#). *Preprint*, arXiv:1810.04805.
- Bosheng Ding, Chengwei Qin, Linlin Liu, Yew Ken Chia, Shafiq Joty, Boyang Li, and Lidong Bing. 2023. [Is gpt-3 a good data annotator?](#) *Preprint*, arXiv:2212.10450.
- Pengrui Han, Rafal Kocielnik, Adhithya Saravanan, Roy Jiang, Or Sharir, and Anima Anandkumar. 2024. [ChatGPT based data augmentation for improved parameter-efficient debiasing of LLMs](#). In *Proceedings of the Fourth Workshop on Language Technology for Equality, Diversity, Inclusion*, pages 73–105, St. Julian’s, Malta. Association for Computational Linguistics.
- Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022. [ToxiGen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3309–3326, Dublin, Ireland. Association for Computational Linguistics.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. [Measuring massive multitask language understanding](#). In *International Conference on Learning Representations (ICLR)*. Published as a conference paper at ICLR 2021.
- Zhuoyan Li, Hangxiao Zhu, Zhuoran Lu, and Ming Yin. 2023. [Synthetic data generation with large language models for text classification: Potential and limitations](#). *Preprint*, arXiv:2310.07849.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- Vukosi Marivate and Tshephisho Sefara. 2020. [Improving Short Text Classification Through Global Augmentation Methods](#), page 385–399. Springer International Publishing.
- OpenAI. 2024. [Gpt-4.1 and gpt-4.1-mini overview](#). Accessed: 2025-05-20.
- OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altmenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. [Gpt-4 technical report](#). *Preprint*, arXiv:2303.08774.
- Foteini Papadopoulou, Osman Mutlu, Neris Özen, Bas H. M. van der Velden, Iris Hendrickx, and Ali Hürriyetoglu. 2025. [Brightcookies at semeval-2025 task 9: Exploring data augmentation for food hazard classification](#). *Preprint*, arXiv:2504.20703.
- Korbinian Randl, John Pavlopoulos, Aron Henriksson, and Tony Lindgren. 2024. [CICLE: Conformal in-context learning for largescale multi-class food risk classification](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7695–7715, Bangkok, Thailand. Association for Computational Linguistics.
- Areeg Fahad Rasheed, M. Zarkoosh, Shimam Amer Chasib, and Safa F. Abbas. 2025. [Data augmentation to improve large language models in food hazard and product detection](#). *Preprint*, arXiv:2502.08687.
- Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. 2024. A systematic survey of prompt engineering in large language models: Techniques and applications. *arXiv preprint arXiv:2402.07927*.
- Vinay Samuel, Houda Aynaou, Arijit Chowdhury, Karthik Venkat Ramanan, and Aman Chadha. 2024. [Can LLMs augment low-resource reading comprehension datasets? opportunities and challenges](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 307–317, Bangkok, Thailand. Association for Computational Linguistics.
- SemEval. 2025. SemEval 2025 Task 9: The Food Hazard Detection Challenge. <https://food-hazard-detection-semeval-2025.github.io/>. Accessed: 2025-05-30.

Sam Shleifer. 2019. [Low resource text classification with ulmfit and backtranslation](#). *Preprint*, arXiv:1903.09244.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2020. [How to fine-tune bert for text classification?](#) *Preprint*, arXiv:1905.05583.

Jason Wei and Kai Zou. 2019. [Eda: Easy data augmentation techniques for boosting performance on text classification tasks](#). *Preprint*, arXiv:1901.11196.

World Health Organization. 2015. [Who estimates of the global burden of foodborne diseases: foodborne disease burden epidemiology reference group 2007-2015](#).

Appendix

A Prompt Template and Example Outputs

Prompt Template The exact prompt format used to elicit synthetic examples from the model is shown below:

```
You are assisting in the development of
a classifier trained to
automatically extract food-related
hazards from product recall
statements. The classifier predicts
both broad categories ('hazard-
category', 'product-category') and
specific labels ('hazard', 'product
') using the recall's title and text
.
```

Examples:

```
{row['shots']}
```

Task:

Generate a new and unique synthetic example to be added to the training dataset. You are given target values below -- use them to craft a realistic recall title and text that help the model learn new patterns.

Target values: 'hazard-category': {row['hazard-category']}, 'product-category': {row['product-category']}, 'hazard': {row['hazard']}, 'product': {row['product']}

IMPORTANT:

Your output must match the exact format used in the examples above and include all of the following fields: 'hazard-category' 'product-category', 'hazard', 'product', 'title', 'text'

Here is an example of an actual prompt used for generating synthetic example:

```
You are assisting in the development of
a classifier trained to
automatically extract food-related
hazards from product recall
statements. The classifier predicts
both broad categories ('hazard-
category', 'product-category') and
specific labels ('hazard', 'product
') using the recall's title and text
.
```

Examples:

```
{'title': 'Pauls Smarter White Milk -
Microbial Contamination (Listeria
Monocytogenes) - 7 June 2012',
'text': 'Parmalat Australia Ltd has
recalled Pauls Smarter White Milk
(1L) from Woolworths, Coles, BP
Service Stations and smaller
independent outlets in NT due to
Listeria Monocytogenes
contamination. Listeria may cause
illness in pregnant women and their
unborn babies, the elderly and
people with low immune systems.\n',
'hazard-category': 'biological',
'product-category': 'meat, egg and
dairy products',
'hazard': 'listeria monocytogenes',
'product': 'raw milk'}
```

```
{'title': 'Heinz - Dijon Mustard Sauce
With a Hint of Orange',
'text': 'PRA No. 2009/10709 Date
published 3 Apr 2009 Product
description 370ml Plastic Bottle.
Best Before: 19/11/10. Identifying
features N/A What are the defects?
Labelling - Undeclared allergen (
egg and milk). What are the hazards
? Illness. What should consumers do
? Return to place of purchase for
full refund. Supplier Heinz
Australia Where the product was
sold Nationally Coordinating agency
Food Standards Australia New
Zealand is the coordinating agency
for this recall. Product category
Groceries x Close',
'hazard-category': 'fraud',
'product-category': 'soups, broths,
sauces and condiments',
'hazard': 'labelling/misdescription',
'product': 'mustard sauce'}
```

Task:

Generate a new and unique synthetic example to be added to the training dataset. You are given target values below -- use them to craft a realistic recall title and text that help the model learn new patterns.

Target values: 'hazard-category': fraud, 'product-category': soups, broths, sauces and condiments, 'hazard': labelling/misdescription, '

```
product': mustard sauce
```

IMPORTANT:

Your output must match the exact format used in the examples above and include all of the following fields: 'hazard-category' 'product-category', 'hazard', 'product', 'title', 'text'

An example of the generated output of one such prompts:

```
{"title": "Urgent Recall of Orchard  
Fresh Apple Cider Due to Possible  
Chemical Contamination", "text": "  
Orchard Fresh Beverages is issuing a  
voluntary recall of its 750mL  
bottles of Orchard Fresh Apple Cider  
after routine testing detected  
traces of an unidentified chemical  
contaminant. The affected product  
carries the batch number OFAC  
-2024-04 and a best before date of  
June 2025. Distributed nationwide  
through grocery stores and online  
retailers, consumers are strongly  
advised to avoid consumption and  
return any purchased bottles to the  
place of purchase for a full refund.  
No adverse health effects have been  
reported so far. This recall is  
conducted in cooperation with the  
National Food Safety Agency to  
ensure public safety. For further  
information, consumers can contact  
Orchard Fresh Beverages customer  
support at 1-888-555-6789 or visit  
www.orchardfresh.com/recall.", "  
hazard-category": "other hazard", "  
product-category": "alcoholic  
beverages", "hazard": "other not  
classified", "product": "cider"}
```

The LLM's output is validated for format consistency and label alignment before being added to the training set. This process enables us to generate a diverse set of high-quality training examples while controlling for label-specific variation and class balance.

Balancing Accuracy, Efficiency, and Training Cost in Runtime-Constrained Food Hazard Classification

Soban Ali

University of Oslo
sobana@uio.no

Sander Sævik

University of Oslo
sanderhs@uio.no

Aliaan Azam

University of Oslo
aliaanna@uio.no

Abstract

Automated food hazard detection helps speed up product recalls, but to work in, it needs to balance accuracy with how fast and efficient it runs. We present a new way to evaluate models for the SemEval-2025 Food Hazard Detection task that measures trade-off using a metric combining macro-F1 score and inference latency. Within this framework, we evaluate a range of models, from traditional bag-of-words classifiers to fine-tuned transformers and hybrid cascade architectures. Our findings show that a well-designed cascade model outperforms RoBERTa in accuracy (0.78 vs. 0.74 F1) while reducing inference latency by 64%. A class-balanced DistilBERT model achieves the best performance on fine-grained hazard detection, while linear SVMs outperform transformers on rare hazard subtypes in both accuracy and speed. We offer deployment recommendations for environments with different resource constraints, helping make food safety monitoring more efficient and scalable.

1 Introduction

Detecting hazardous food products early is essential to protect public health. In food safety monitoring, delays in identifying and classifying food recalls can have real-world consequences. Scallan et al. (2011b,a) estimate that foodborne illnesses in the U.S. affect approximately 48 million people annually. This includes 9.4 million cases caused by known pathogens and 38.4 million by unspecified agents. The faster we can detect and act on these hazards, the better. One solution is to use natural language processing (NLP) systems that automatically classify food recall notices into structured labels. Like what kind of hazard is involved or what type of product it affects. This kind of automation allows for faster response, better monitoring, and potentially reduces the impact of contaminated products. Many NLP systems developed for such

tasks optimize only for accuracy and neglect real-world deployment constraints such as inference latency, computational resource usage, and training time.

The SemEval-2025 Food Hazard Detection task (Randl et al., 2024) introduces a classification benchmark with two subtasks: (ST1) broad category prediction (hazard-category, product-category) and (ST2) fine-grained prediction (specific hazard and product). Although earlier work focused on improving F1 scores (Randl et al., 2024), it largely ignored whether these models are efficient enough for use in time-sensitive monitoring systems. This oversight is problematic, especially in food safety space. Delays in this process can jeopardize public safety, particularly during widespread contamination events. We address this gap by evaluating models performance-efficiency trade-offs specifically tailored for food hazard monitoring contexts. Our contributions are as follows.

1. A new composite metric M_{eff} that evaluates models based on both their macro-F1 scores and their inference latency. This helps compare models in a way that reflects the real-world need for both speed and accuracy.
2. A systematic comparison of classical, transformer-based, and hybrid cascade classifiers, with specific attention to their performance on imbalanced hazard classes.
3. Empirical analysis on the CICLe dataset highlighting the practical accuracy-runtime implications for food safety monitoring systems.
4. Clear deployment recommendations, tailored for different use cases, from environments with tight resource constraints to systems where maximizing accuracy is the top priority.

2 Related Work

The task of automatically classifying food recall texts, especially under tight runtime or resource limits, sits at the crossroads of three main research areas: domain-specific natural language processing (NLP), cost-efficient inference, and class-imbalanced learning. In this section, we review the foundational work in food hazard classification, advances in NLP-based classification under resource constraints, and recent developments in hybrid and cascade architectures that balance performance with run-time cost.

2.0.1 Foundational Work in Food Hazard Classification

An important piece of related work comes from [Randl et al. \(2024\)](#), who introduced the benchmark dataset and task definition for large-scale food hazard detection, which focuses on predictive accuracy across highly imbalanced class distributions. Their work revealed that classical models such as Support Vector Machines (SVMs) can outperform deep learning baselines in macro-F1 score due to the sparsity and domain specificity of the dataset. Their study was effective in controlled evaluations, but did not address run-time feasibility for real-time deployments, a gap this paper aims to fill. Our work builds on their dataset and metrics but introduces architectures optimized for efficiency, drawing from deployment-centric research.

2.0.2 Runtime-Constrained NLP and Hybrid Models

In the last few years, there has been a growing interest in making NLP models more efficient, especially for situations where speed and resource usage actually matter. [Strubell et al. \(2019\)](#) brought major attention to the environmental and financial cost of running large models, which pushed researchers to explore smaller, faster alternatives that are more practical for real-world use. One popular solution is DistilBERT ([Sanh et al., 2019](#)), a lightweight version of BERT, that manages to keep much of its performance while being considerably smaller and faster.

More recently, researchers have started exploring hybrid and cascade models as a promising solution for balancing inference cost with predictive power. [Li et al. \(2021\)](#), for example, showed that cascaded transformers, where only uncertain predictions get passed to a heavier model, can outperform single-model setups when you account for both accuracy

and efficiency. These systems rely on confidence thresholds to decide when to escalate a prediction, which helps cut down on unnecessary computation. This idea directly inspired our own architecture. By combining a lightweight SVM with Distilbert in a cascade, we aim to get the best of both worlds, fast predictions when confidence is high, and more nuanced classification from the transformer when needed.

2.0.3 Addressing Label Imbalance

Food hazard classification comes with a heavily imbalanced label distribution, where some classes appear often, while many others are rare. This kind of long-tail setup makes it difficult for standard supervised models to perform well, especially when it comes to generalizing beyond the dominant classes.

To tackle this problem, several strategies have been explored. One common approach is using class-weighted loss functions, which give more importance to rare classes during training ([Henning et al., 2023](#)). Others have applied data augmentation techniques like EDA (Easy Data Augmentation) to synthetically boost underrepresented examples ([Wei and Zou, 2019](#)). There is also few-shot learning methods, which aim to make accurate predictions even when only a handful of examples are available for a given class.

3 Methodology

3.1 Task and Data

The SemEval-2025 Food Hazard Detection dataset contains 55,902 food recall notices that have been annotated for analysis. Of these, 6,215 are set aside for validation. Each notice is labeled for two tasks: ST1, which includes general categories for the hazard and the product, and ST2, which involves more specific labels, such as the exact hazard and product mentioned. ST2 is much more detailed, with more than 1,200 unique products and hundreds of different hazards. This is because the dataset is heavily imbalanced across classes. The macro-F1 score is used as the main evaluation metric. The data is only lightly preprocessed, models are fed a simple combination of each notice’s title and description if available. Following the official setup, we tune model parameters using the validation set and report final results on the test set.

3.2 Models Evaluated

BoW + Logistic Regression: This is a simple and fast baseline that uses TF-IDF features based on character n-grams (ranging from 2 to 5). We train a logistic regression model with L2 regularization and a penalty strength of $C = 1.0$. Despite its simplicity, it performs well at capturing frequent text patterns and provides a solid benchmark.

Linear SVM: This model uses the same TF-IDF features as the logistic regression baseline, but is trained using a linear support vector machine with hinge loss and $C = 1.0$, implemented via LinearSVC. To support probability-based decisions in the cascade system, we apply Platt-scaling. Linear SVMs are well-suited for handling high-dimensional sparse data, making them effective for text classification tasks like these.

DistilBERT: We fine-tune a 66 million parameter distilled transformer model (Sanh et al., 2019) separately for ST1 and ST2. Each model has a classification head with two outputs and uses cross-entropy loss. Training is run for 5 epochs on an NVIDIA A100 GPU with a batch size of 64, a maximum sequence length of 256, and a learning rate of 5×10^{-5} using the AdamW optimizer.

RoBERTa: We also fine-tune a larger model with 125 million parameters (Liu et al., 2019), using the same training setup as DistilBERT. RoBERTa achieves higher accuracy. RoBERTa operates at roughly half the speed of other models in both training and inference. We include it as a reference point to benchmark upper-bound performance.

DistilBERT + Class-Balanced Reweighting: To address the pronounced class imbalance, particularly in ST2 where we enhance the DistilBERT model by incorporating a cross-entropy loss function. Each class c is given a weight defined as:

$$w_c = \frac{N}{\text{frequency}(c) \cdot C} \quad (1)$$

where N is the total number of training samples, C is the number of classes, and $\text{frequency}(c)$ is the number of samples belonging to class c . This weighting gives more importance to rare classes during training. As with the vanilla DistilBERT, we train separate models for each subtask with the same hyperparameters.

Cascade (SVM + DistilBERT): We also evaluate a two-stage cascade system. The first stage runs the SVM on all inputs. If the SVM’s confidence

score is above 0.85, we accept its prediction. If the confidence is below 0.85, we fall back to the DistilBERT model for a second opinion. This threshold was chosen based on validation experiments where we tested values from 0.70 to 0.95 in steps of 0.05, and selected the one that gave the highest M_{eff} score on the validation set. We apply this cascade design separately for each subtask and evaluate both the standard and class-balanced versions of DistilBERT in this setup.

Statistical Testing: All performance metrics reported are averaged over three independent training runs to account for variability.

3.3 Efficiency Metric

To evaluate how well models perform under real-time constraints relevant to food safety, we introduce an efficiency score called M_{eff} . It combines accuracy and speed using the following formula:

$$M_{\text{eff}} = F1 \cdot e^{-\lambda \cdot T_{\text{inf}}}$$

In this formula, $F1$ is the macro-averaged F1 score, and T_{inf} is the total inference time in seconds on the test set. The parameter λ controls how much slower models are penalized. We use $\lambda = 0.2$ as our main setting, which reflects a moderate concern for delay.

This metric is directly inspired by the exponential decay formulas. These formulas are common in fields like economics and engineering. They model how the value of something drops quickly the longer you wait. For example, important information can become much less useful if it arrives even a few seconds late. Our formula follows the same idea. It reflects how small delays in model predictions can still have a big impact. This is especially important in food safety, where every second counts during a product recall.

To show that our findings are robust, we also test with $\lambda = 0.1$ and $\lambda = 0.3$, which simulate environments with lower and higher sensitivity to latency. Table 5 in Appendix presents M_{eff} scores across three λ values, simulating varying sensitivity to inference latency.

This metric is designed with real-world deployment in mind. In food safety monitoring, even small delays can have serious consequences. For instance, during an active recall, the speed of classification directly affects how quickly contaminated products can be removed. Our formulation applies a smooth penalty to longer inference.

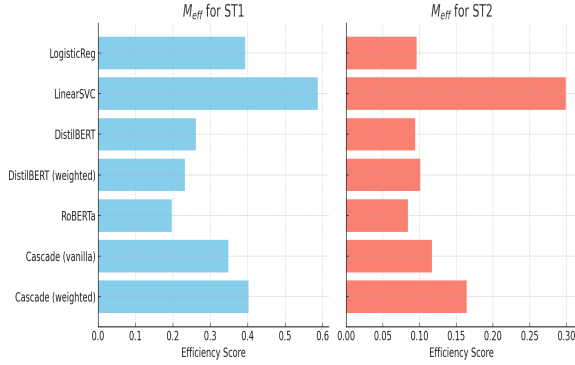


Figure 1: Latency-adjusted efficiency scores (M_{eff}) for each model across ST1 and ST2. Higher is better. LinearSVC achieves the best trade-off between speed and accuracy.

4 Results

Table 1 presents macro-F1 scores, inference times, and latency-adjusted efficiency scores M_{eff} for all evaluated models. Results are based on the full test set of 6,215 samples. Inference was performed on a standardized compute environment (single NVIDIA A100 GPU for transformer models, 32-core CPU for classical models) to ensure fair comparison.

4.1 Hazard and Product Categories

For the broader ST1 classification task, the weighted cascade model delivers the best overall performance. It achieves a macro-F1 score of 0.777, outperforming RoBERTa’s 0.743, while taking just 3.41 seconds to run compared to RoBERTa’s 9.52 seconds. This means that the cascade is not only more accurate, but also much faster, reducing inference time by 64%. When we look at the efficiency metric M_{eff} , the gain is even clearer: the cascade scores 0.402, more than double RoBERTa’s 0.197. This highlights how effective the cascade setup is at balancing accuracy and speed.

DistilBERT on its own does reasonably well, with a macro-F1 of 0.711 and a runtime of 6 seconds, but it still falls short of the cascaded alternatives. This shows that using the transformer selectively, only when needed, is a key strategy to maintain efficiency without giving up too much performance.

Interestingly, LinearSVC also holds its own. Although its F1 score is lower (0.651), its inference time is extremely fast at just 0.51 s. As a result, it achieves a strong M_{eff} of 0.587. This shows that

in latency sensitive settings, a simple linear model can still be a competitive option.

4.2 Fine-Grained Hazards and Products

The results for the more detailed ST2 task tell a different story. LinearSVC and the weighted DistilBERT model achieve nearly identical macro-F1 scores: 0.332 and 0.335, respectively, indicating similar performance in terms of precision. However, LinearSVC clearly stands out in efficiency, achieving a M_{eff} of 0.299 thanks to its much faster inference time.

The weighted cascade model performs about the same as RoBERTa in terms of F1 score (0.316 vs. 0.317), but does so with 64% less latency. Even with this efficiency advantage, it still trails behind LinearSVC by about 4.8 percentage points in F1, suggesting that the simpler model is better suited for this task under real-world constraints.

These results suggest that traditional models like LinearSVC may be better at handling fine-grained classification, especially in highly imbalanced datasets. This might be because they handle rare labels better and don’t need a lot of data for each class. On the other hand the standard DistilBERT model does worse, with a macro-F1 of just 0.257. This shows the difficulty of using smaller transformer models for tasks involving multiple rare classes unless proper techniques like class balancing are applied.

4.3 Efficiency-Accuracy Trade-offs

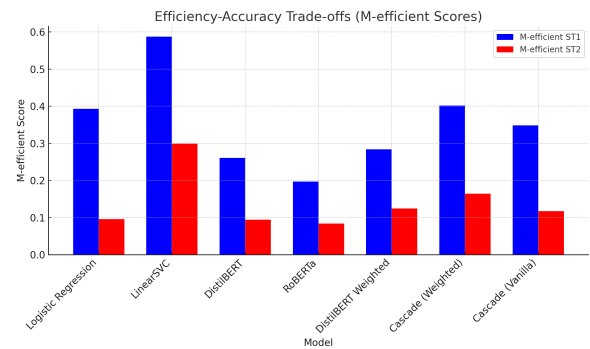


Figure 2: Efficiency-Accuracy trade-offs visualized using M_{eff} scores across ST1 and ST2. Higher values indicate better performance under latency constraints.

The M_{eff} metric provides useful insights into which models are most suitable for deployment. In the ST1 task, the weighted cascade model reduces latency by 40% compared to RoBERTa, while also improving F1 by 4.6%. This combination is valu-

Model	ST1 F1	ST2 F1	Infer (s)	M_{eff} ST1 ($\lambda=0.2$)	M_{eff} ST2 ($\lambda=0.2$)
LogisticReg	0.480	0.118	1.16	0.381	0.094
LinearSVC	0.651	0.332	0.51	0.588	0.300
DistilBERT	0.711	0.257	6.00	0.214	0.077
DistilBERT (weighted)	0.766	0.335	5.96	0.233	0.102
RoBERTa	0.743	0.317	9.52	0.111	0.047
Cascade (vanilla)	0.730	0.245	3.99	0.329	0.110
Cascade (weighted)	0.777	0.316	3.41	0.393	0.160

Table 1: Macro-F1 scores, inference time (in seconds), and efficiency scores $M_{\text{eff}} = F1 \cdot e^{-\lambda T_{\text{inf}}}$ with $\lambda = 0.2$. Bold indicates best performance.

able because the exponential penalty in M_{eff} gives more weight to speed. This means that even small latency improvements can have a big impact on the final score.

In the ST2 task, LinearSVC proves to be the most efficient overall. Although its F1 score is only moderate, its very fast inference time of 0.51 seconds results in an M_{eff} score that is 82% higher than that of the weighted cascade model. This shows that speed alone can make a simpler model more viable in real-world, time-sensitive settings.

RoBERTa, on the other hand, struggles in terms of efficiency. Its high latency of 9.52 seconds makes it a poor choice for deployment, even though it performs well in terms of accuracy on ST1. In fact, its efficiency score in ST2 is so low that it falls behind even Logistic Regression (0.084 vs. 0.096). This highlights the limitations of large models when inference time is a key concern.

4.4 Training Cost Breakdown

Besides how fast a model makes predictions, training time also matters. This is especially important when the model needs to be retrained often, such as when new product types or hazards appear. To show this, we include the total time it takes to fine-tune each model on an NVIDIA A100 GPU, with results shown separately for each subtask (ST1 and ST2).

Although training time is not included in the M_{eff} metric, it still provides a useful estimate of practical concerns, and how easily a model can be deployed or updated in resource-constrained environments.

Analysis: As shown in Table 2, classical models are highly cost-effective to train. LinearSVC is especially efficient, completing training in just 34 seconds, compared to 377 seconds for Logistic Regression. Despite both running on CPU, Lin-

Model	ST1 Train Time (s)	ST2 Train Time (s)	Total (s)
Logistic Regression	21.61	355.43	377.04
LinearSVC	1.82	31.96	33.78
DistilBERT	120.2	120.9	241.1
RoBERTa	262.2	262.2	524.3
DistilBERT (weighted)	475.1	475.9	951.0

Table 2: Training times for each model, measured in seconds on an NVIDIA A100 GPU. Cascade systems reuse their component models. Logistic Regression and LinearSVC measured on a 32 core CPU.

earSVC is faster, making it a better choice for frequent retraining or resource-constrained environments.

In contrast, transformer-based models require much more compute. DistilBERT trains in 241 seconds, RoBERTa in 524 seconds, and the weighted DistilBERT takes 951 seconds due to its data weighting.

Cascade systems allow modular retraining, where only the fast model needs updating while the fallback model remains unchanged. This makes them practical for real-world systems that evolve step by step.

While inference efficiency affects how fast the model runs, training cost affects how easy it is to keep the model updated. For setups that need to be cheap or updated often, LinearSVC gives the best balance. Larger models should only be used if their better performance is worth the extra training time.

4.5 Architectural Implications

Cascaded architectures work best when the second-stage model, in this case DistilBERT, is able to correct systematic mistakes made by the first-stage classifier, which is the SVM. In the ST1 task, the SVM alone correctly handled 68 % of the test samples. DistilBERT was only used for the remaining low-confidence cases. This setup significantly reduced the average time spent using DistilBERT by 43 percent while still maintaining high accuracy.

In contrast, for the more detailed ST2 task, the SVM already achieves higher accuracy than the full cascade. Its macro-F1 score is 0.3316 compared to 0.3163 for the cascade system. As a result, the extra time spent invoking DistilBERT does not lead to better performance and actually adds unnecessary delay.

These findings show that there is no solution that fits all sizes. Different tasks benefit from different approaches. For ST1, where high-level categories are predicted, cascaded transformers offer a strong balance between speed and accuracy. For ST2, where the labels are more specific and imbalanced, simpler models like LinearSVC provide better efficiency with little or no loss in performance.

5 Analysis and Discussion

5.1 Error Analysis and Class Imbalance

To gain deeper insight into model weaknesses, we conducted an error analysis on 100 randomly chosen misclassified instances. Among transformer-based models, a large proportion of errors were associated with rare ST2 labels. These were often highly specific categories with very few training samples, such as processing with 39 examples and packaging defect with 22. Table 4 in Appendix A provides examples that illustrate these types of errors.

These issues were especially common in transformer models that were trained without any class balancing. By applying class weights, the weighted DistilBERT model performed noticeably better on rare labels. It raised F1 scores for underrepresented categories and handled the long tail of the label distribution more effectively. The imbalance in the dataset is large. Common categories like *listeria monocytogenes* with 665 samples and *salmonella* with 621 samples appear 30 times more often than rare ones like yeasts and brazil nut, which each have only 4 examples.

For the ST1 task, where labels are more evenly distributed, we found that most classification errors occurred between categories that are semantically similar. This includes confusion between different types of allergens. In these cases, DistilBERT and the cascade models showed clear advantages by using the context of the text to distinguish between closely related labels.

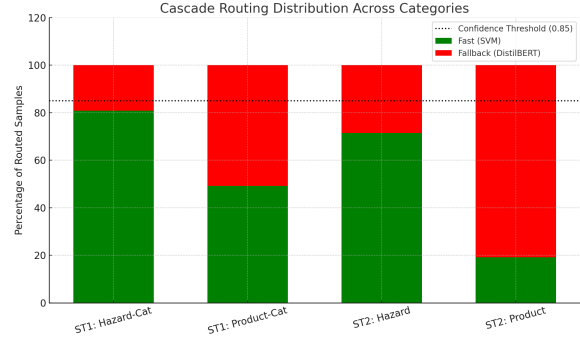


Figure 3: Distribution of SVM confidence scores and routing decisions in the cascade architecture. Green indicates examples handled by SVM, red indicates fallback to DistilBERT. Dotted line shows the 0.85 confidence threshold.

5.2 Runtime Trade-offs and Cascade Behavior

The hybrid cascade architecture demonstrates significant efficiency benefits by routing predictions. Figure 3 visualizes this routing across confidence levels and categories. In ST1, the SVM confidently handled 68% of samples (confidence ≥ 0.85), allowing DistilBERT to focus on harder cases. This reduced average inference time significantly compared to DistilBERT alone, while improving accuracy. The optimal confidence threshold of 0.85 was determined through validation experiments.

As shown in Table 3, setting the threshold to 0.85 results in the highest macro-F1 score of 0.550, outperforming all other tested values. The performance improves steadily from 0.70 up to 0.85, but starts to decline slightly at higher thresholds. These results show that using a threshold of 0.85 offers the best trade-off between accuracy and efficiency. It allows the fast model to handle most predictions confidently, while still deferring uncertain cases to the transformer model when necessary to try to improve overall performance.

Threshold	Macro F1
0.70	0.529
0.75	0.531
0.80	0.533
0.85	0.550
0.90	0.534
0.95	0.537

Table 3: Impact of confidence threshold on cascade performance. The 0.85 threshold balances accuracy and efficiency.

5.3 Error Patterns in Misclassified Examples

Our analysis of misclassified examples revealed several recurring patterns that help explain why the models made certain mistakes.

Confusion between allergen categories: Many errors involved mislabeling one allergen type as another. For example, products containing milk were sometimes labeled as containing eggs or walnuts. This shows that the model struggles to distinguish between allergens with similar descriptions or contexts.

Physical contaminants confusion: There was frequent confusion between physical contaminants such as plastic fragments, metal fragments, and glass fragments. These categories often appear in similar contexts in the recall notices, which likely contributes to the difficulty in telling them apart.

Label and inspection-related errors: The model often confused categories about labeling and inspection. For example, it mixed up wrong or unclear labels with inspection problems. These categories use similar words and talk about different parts of regulatory violations, which may explain why the model got them mixed up.

Processing versus contamination: In some cases, the model confused processing errors with contamination types. This often occurred when a processing issue was the cause of contamination, making it difficult to separate the two based solely on the recall text.

5.4 Practical Deployment Recommendations

Our results show that the most suitable model for food hazard classification depends on the specific deployment context. In this section, we provide practical recommendations for three common scenarios, based on accuracy, inference speed, and training cost.

1. Real-time screening with strict latency requirements

In environments where large volumes of recall notices must be processed quickly, such as first-pass screening systems or low-power devices, **LinearSVC** is the preferred choice. It offers fast and reliable performance for common hazard categories, with inference times under one second. Its low training cost and minimal hardware requirements make it well suited for deployments with limited computational resources. While it may not capture rare cases, it is highly effective for quick filtering.

2. Balanced performance for routine monitoring

For general-purpose monitoring systems where both speed and accuracy are important, such as those used by food safety agencies, the **weighted cascade model** provides an excellent trade-off. This system uses the fast **LinearSVC** model for confident predictions and defers to the more accurate **DistilBERT** model only when needed. By setting the confidence threshold to 0.85, the system routes most inputs through the SVM while still handling difficult cases with a transformer. This significantly reduces latency while maintaining high macro-F1 scores.

3. High-accuracy analysis for rare hazard detection

When it's important to find rare or high-risk hazards as accurately as possible, the **class-balanced DistilBERT** model works best. It does well with labels that don't appear often because it uses a weighted loss. Even though it takes more time to run and train, it's a good choice for tasks like careful audits, batch checks, or any situation where accuracy matters most.

6 Limitations and Future Work

Although our approach demonstrates promise for practical food hazard monitoring, it also presents several limitations that highlight valuable avenues for future investigation.

Dataset and Evaluation Limitations

One major limitation lies in the dataset itself. The SemEval-2025 dataset is highly imbalanced and narrowly focused on food recall texts. This raises concerns about how well the models would generalize to broader domains, or unstructured recall data from other regions or industries. Also, our experiments show macro-F1 and latency through the M_{eff} metric, but other factors such as energy consumption, model interpretability, and trustworthiness were not evaluated. These are also important for deployment in public health settings.

Limitations of the Cascade Design

Our cascade architecture depends on a fixed confidence threshold of 0.85 to decide whether to defer to the transformer model. Although this works well in many cases, it may not offer the best balance between speed and accuracy across all labels or inputs. The static threshold must be tuned manually for each setting, and the model's performance

can suffer if the confidence scores are not well calibrated. More flexible designs could improve scores and reliability.

Metric Design and Efficiency Scope

The M_{eff} metric is helpful for capturing the trade-off between accuracy and inference time, but it uses a fixed penalty value of $\lambda = 0.2$. This setting might not be the best for all deployment scenarios. Also, the metric only considers runtime, while training time and resource usage are also important. This is especially important in applications where frequent retraining is needed. Our current evaluation does not account for these factors.

Architectural Simplicity

We use a fairly standard setup based on fine-tuning DistilBERT, which is a mid-sized transformer. While this balances performance and efficiency well, it may not be optimal for all environments. There is room to explore more efficient and smaller architectures like TinyBERT or quantized models like Q8BERT (Zafrir et al., 2019), also early-exit transformers (Xin et al., 2020) can be used for reducing memory usage and computation.

Future Research Directions

There are many good ways to improve this work. One clear idea is to make the cascade system more flexible. Instead of using one fixed threshold, future systems could learn to use different thresholds based on the class or the input. This could be done by using data to learn better rules, like methods that take uncertainty into account or small decision tools that guess uncertainty more accurately.

Improving domain generalization is also a key challenge. Our current models are trained on English language food recalls, but practical systems will often need to work with multilingual data or in different industries. Using multilingual transformers or prompt-based few-shot learning techniques could make the system more flexible and adaptable to low-resource settings.

In addition, future work should look at larger forms of efficiency. For instance, real-world deployment may require models that not only run quickly but also use less memory and energy. Compression techniques such as pruning, quantization, and knowledge distillation could help make the system more suitable for low-power environments such as mobile devices.

We also see value in exploring more advanced data augmentation techniques. Although basic methods like synonym replacement or sentence shuffling can help with class imbalance, they are often not enough for specialized tasks like food hazard detection. We can use more advanced methods, such as back-translation, contextual augmentation with masked language models can create better training data, especially for rare classes.

An important next step would be to evaluate the system within a real-time deployment pipeline. This would require addressing shifting data distributions and implementing strategies for continuous model updating. A real-time system could include a human-in-the-loop component. This can be used for uncertain or high-risk predictions, which are then reviewed by experts. These improvements would bring the model closer to deployment in actual food safety monitoring systems.

7 Conclusion

This report tackles the challenge of deploying food hazard classification systems that must deliver accurate predictions under real-world constraints such as latency and efficiency. Using the SemEval-2025 Food Hazard Detection dataset, we introduced a performance-based evaluation framework built around a new metric, M_{eff} , which combines macro-F1 with inference time to reflect deployment priorities.

Our evaluation of traditional models, transformer-based models, and hybrid cascade systems showed that the best choice depends on how the model will be used. For general hazard and product classification (ST1), the weighted cascade model gave the best balance between accuracy and speed, beating RoBERTa while cutting latency by more than 60%. For more detailed classification tasks (ST2), LinearSVC turned out to be the most efficient, due to its fast processing and ability to handle unbalanced classes well.

Additionally, we found that techniques such as class reweighting significantly improve performance on rare categories, and that modular cascade architectures support flexible deployment by adapting to different resource environments. These results show the importance of choosing models based on operational goals, such as speed, accuracy, or flexibility.

Finally, further work should look into making

cascades more flexible, improving how models work across different sectors and domains. Also using better efficiency measures that include training time and energy use. To make NLP useful for public health, we need to better connect model performance with how easy it is to actually deploy it in a resource-limited real-world setting.

A Appendix

References

- Sophie Henning, William Beluch, Alexander Fraser, and Annemarie Friedrich. 2023. A survey of methods for addressing class imbalance in deep learning-based nlp. In *Proceedings of EACL*.
- Lei Li, Yankai Lin, Deli Chen, Shuhuai Ren, Peng Li, Jie Zhou, and Xu Sun. 2021. Cascadebert: Accelerating inference of pre-trained language models via calibrated complete models cascade. In *Findings of EMNLP*.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- Lukas Randl, Shreya Strinivasan, Bahaa Makni, John Pavlopoulos, and Shay B. Cohen. 2024. Cicle: Conformal in-context learning for large-scale multi-class food risk classification. In *Findings of the Association for Computational Linguistics: ACL 2024*.
- Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter.
- Elaine Scallan, Patricia M. Griffin, Frederick J. Angulo, Robert V. Tauxe, and Robert M. Hoekstra. 2011a. Foodborne illness acquired in the united states—unspecified agents. *Emerging Infectious Diseases*.
- Elaine Scallan, Robert M. Hoekstra, Frederick J. Angulo, and 1 others. 2011b. Foodborne illness acquired in the united states—major pathogens. *Emerging Infectious Diseases*.
- Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. In *Proceedings of ACL*.
- Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of EMNLP-IJCNLP*.
- Ji Xin, Rodrigo Nogueira, Yaoliang Yu, and Jimmy Lin. 2020. Early exiting bert for efficient document ranking. In *Proceedings of ACL*.
- Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. 2019. Q8bert: Quantized 8bit bert. In *Proceedings of the EMC2 Workshop at NeurIPS*.

Text Excerpt	True Label	DistilBERT Prediction	LinearSVC Prediction
"Allergy Alert Issued by Santa Rosa Whole Foods Market for Undeclared Walnut, Milk and Eggs in 'Hipst...'"	milk and products thereof	walnut	walnut
"Mann Packing Voluntarily Issuing Class 1 Recall of Organic Veggies Snacking Tray..."	milk and products thereof	eggs and products thereof	eggs and products thereof
"Parmalat Australia Ltd—Vaalia My First Yoghurt Apple & Raspberry, Vaalia My First Yoghurt Banana..."	packaging defect	plastic fragment	plastic fragment

Table 4: Representative classification errors from both models. Confusions often occur between allergens and physical contaminants.

Model	ST1 ($\lambda=0.1$)	ST1 ($\lambda=0.2$)	ST1 ($\lambda=0.3$)	ST2 ($\lambda=0.1$)	ST2 ($\lambda=0.2$)	ST2 ($\lambda=0.3$)
LogisticReg	0.427	0.381	0.339	0.105	0.094	0.083
LinearSVC	0.619	0.587	0.559	0.315	0.300	0.285
DistilBERT	0.390	0.214	0.118	0.141	0.077	0.042
DistilBERT (weighted)	0.422	0.233	0.128	0.185	0.102	0.056
RoBERTa	0.287	0.111	0.043	0.122	0.047	0.018
Cascade (vanilla)	0.490	0.329	0.221	0.164	0.110	0.074
Cascade (weighted)	0.552	0.393	0.279	0.225	0.160	0.114

Table 5: Latency-adjusted efficiency scores $M_{\text{eff}} = F1 \cdot e^{-\lambda T_{\text{inf}}}$ for ST1 and ST2 across $\lambda = \{0.1, 0.2, 0.3\}$, showing sensitivity to different latency penalty levels.

2 Parameter-efficient definition modeling track

Parameter-Efficient Definition Modeling in Norwegian with LoRA and IA3

Yuanyuan Jiang and Leheng Chieu

Department of Informatics

University of Oslo

{yuanyuaj, lehc}@uio.no

Abstract

This study explores parameter-efficient fine-tuning (PEFT) methods for the task of definition modeling on Norwegian dataset. We investigate the effectiveness of two PEFT approaches: Low-Rank Adaptation (LoRA) and Infused Adapter by Inhibiting and Amplifying Inner Activations (IA3), across both encoder-decoder (mt0-xl) and decoder-only (Unbabel/TowerInstruct-7B-v0.2) language models. Definition modeling requires generating dictionary-style definitions based on contextual word usage, a task that benefits from fine-grained control over linguistic outputs. We evaluated the models using BLEU and BERTscore metrics and use two-sample t-tests to assess the statistical significance of performance differences. Our findings show that LoRA and IA3 can both improve performance, with the encoder-decoder architecture generally outperforming the decoder-only one when referring to BERTScore F1. All performance differences were statistically significant ($p < 0.05$), confirming the importance of choosing appropriate PEFT strategies and hyperparameter settings for high-quality lexical generation. IA3 configurations were consistently faster to train than LoRA, due to fewer parameters being introduced and updated.

1 Introduction

Definition modeling (Mickus et al., 2022; Fedorova et al., 2024) is a natural language processing task aimed at generating dictionary-style definitions for a target word based on its usage in context. This task not only supports lexical resource development, but also offers insights into a model’s semantic generalization capabilities. With the growing size of language models, parameter-efficient fine-tuning (PEFT)¹ (Han et al., 2024) techniques provide a practical solution for adapting models without retraining all parameters.

In this study, we focus on two PEFT methods: Low-Rank Adaptation (LoRA)² (Hu et al., 2021)

and Infused Adapter by Inhibiting and Amplifying Inner Activations (IA3)³ (Liu et al., 2022), and apply them to the definition modeling task in Norwegian. LoRA adapts language models by injecting trainable low-rank matrices into each layer, while IA3 modifies layer activations through input-conditioned scaling factors. Although both methods are widely supported in frameworks such as Hugging Face’s PEFT library, their comparative performance under varying hyperparameters has not been systematically studied for definition modeling.

We conduct experiments using two foundational architectures: the encoder-decoder model mt0-xl (Muennighoff et al., 2022) and the decoder-only model Unbabel/TowerInstruct-7B-v0.2 (Alves et al., 2024). Each model is fine-tuned using both LoRA and IA3, with multiple hyperparameter configurations. Evaluation is carried out using BLEU (Papineni et al., 2002) and BERTscore (Zhang et al., 2020; Hanna and Bojar, 2021), which are well-established metrics for definition generation, alongside two-sample t-tests⁴ (Cressie and Whitford, 1986) to assess statistical significance of performance differences.

The objectives of this research are to: 1) analyze the impact of different hyperparameters on model performance, 2) correlate subjective impressions with BLEU and BERTScore metrics, 3) compare behavior between encoder-decoder and decoder-only models, 4) evaluate the trade-off between speed (efficiency) and quality in PEFT configurations.

¹<https://huggingface.co/docs/peft/main/en/index>

²https://huggingface.co/docs/peft/main/en/developer_guides/lora

³https://huggingface.co/docs/peft/main/en/task_guides/ia3

⁴https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ttest_ind.html

2 Background and Related Work

2.1 Definition modeling

Definition modeling (Mickus et al., 2022; Fedorova et al., 2024) is to produce a dictionary-style definition of a given word, typically conditioned on its usage in context. The task has garnered increasing interest through dedicated shared tasks such as SemEval-2022 Task 1: “CODWOE – Comparing Dictionary and Word Embeddings” (Mickus et al., 2022), which investigated whether word embeddings can be meaningfully converted into dictionary-style definitions by framing this as a multilingual definition modeling challenge. More recently, AXOLOTL’24: “Shared Task on Multilingual Explainable Semantic Change Modeling” (Fedorova et al., 2024) focused on explainable semantic change modeling, where systems generate dictionary-like definitions for newly detected word senses to enhance explainability in semantic change modeling.

Despite progress, the impact of model architecture and parameter-efficient fine-tuning techniques on definition quality remains underexplored, particularly in low-resource or non-English contexts. This motivates our systematic comparison of PEFT methods for Norwegian definition modeling.

2.2 LoRA

LoRA (Low-Rank Adaptation)² (Hu et al., 2021) is a reparametrization-based parameter-efficient fine-tuning technique designed to adapt large pre-trained models to specific tasks without modifying the entire model. It achieves this by introducing low-rank matrices into the layers of the model, which allow for a smaller number of parameters to be learned during fine-tuning. This reduces the computational cost and memory requirements compared to traditional fine-tuning methods. LoRA has been shown to be effective in achieving strong performance on downstream tasks while being resource-efficient.

Formally, given a weight matrix $W \in R^{d \times k}$ in a transformer layer, LoRA modifies the forward pass by adding a learnable perturbation BA such that $W' = W + \alpha BA$, where $A \in R^{r \times k}$ and $B \in R^{d \times r}$ are low-rank matrices with $r \ll \min(d, k)$ and α is a scaling factor. During training, only A and B are updated, leaving the original model weights W frozen. As discussed in Section 5, the primary tunable hyperparameters are rank r and scaling factor α , which jointly control the expressiveness

and impact of LoRA adaptation.

LoRA has been successfully applied across various domains, including natural language understanding (Borhanifard et al., 2025), text generation (Yu et al., 2022), and machine translation (Wang et al., 2022). Its compatibility with large-scale language models and effectiveness in low-resource adaptation make it particularly appealing for tasks like definition modeling, where full fine-tuning may be prohibitively expensive.

2.3 IA3

IA3 (Infused Adapter by Inhibiting and Amplifying Inner Activations)³ (Liu et al., 2022) is another parameter-efficient fine-tuning (PEFT) method. Compared to LoRA, IA3 introduces a smaller size of trainable parameters by incorporating scaling vectors instead of low-rank matrices. A scaling vector means a learnable vector that modulates activations in a transformer model by scaling them, and it is similar to attention but without introducing large matrices.

IA3 is particularly well-suited for definition modeling tasks, especially in low-resource settings such as Norwegian. IA3 enables parameter-efficient fine-tuning without altering the core weights of the pre-trained language model. This makes it ideal for semantic generation tasks like definition modeling, which require precise adaptation to subtle contextual and lexical cues. Its lightweight nature and compatibility with existing transformer architectures also make it a practical choice for comparative studies involving multiple model types and adaptation strategies.

2.4 BLEU

The BLEU (Bilingual Evaluation Understudy)¹⁰ (Papineni et al., 2002) score is a widely used automatic metric for evaluating machine translation quality. It measures the overlap between machine-generated translations and one or more human reference translations by comparing shared n-grams—contiguous sequences of words—typically from 1-gram to 4-gram. BLEU calculates a modified precision for these n-grams and combines them using a weighted geometric mean. To penalize overly short translations that may artificially achieve high precision, BLEU includes a brevity penalty. A higher BLEU score generally indicates closer similarity to human translation, though it does not account for semantic meaning or fluency. The BLEU scores here are percentages and thus be-

tween 0 and 100. Due to the very short and highly variable nature of definitions, BLEU scores of 3-6 for most languages are normal.

2.5 BERTscore F1

BERTscore F1 (Zhang et al., 2020; Hanna and Bojar, 2021) is an advanced evaluation metric for natural language generation tasks such as machine translation, summarization, and text generation. Unlike traditional metrics like BLEU, which rely on exact n-gram overlap, BERTscore leverages contextual embeddings from pre-trained language models (such as BERT or RoBERTa) to assess the semantic similarity between candidate and reference sentences. To compute BERTscore, each token in both candidate and reference sentences is represented using contextualized embeddings. Then, the metric matches tokens from the candidate to the most similar tokens in the reference (based on cosine similarity), and vice versa. Precision and recall are calculated from these matches, and the F1 score, which means the harmonic mean of precision and recall, represents the final BERTscore. Because BERTscore evaluates meaning rather than surface-level form, it correlates better with human evaluation, especially when paraphrasing or synonym usage is involved.

2.6 Two-Sample t-Test

A t-test⁴ is a statistical method used to determine whether there is a significant difference between the means of two groups. It is particularly useful when comparing results from experiments or different model configurations to check if the observed differences are due to chance or are statistically meaningful. The independent two-sample t-test (Cressie and Whitford, 1986) refers to compare the means of two independent groups (e.g., results from model A vs. results from model B), and the goal is to determine whether the observed difference is statistically significant. The input value should be two independent sets of numerical values, e.g., BLEU scores or BERTScores from two different models over multiple samples. The output includes t-statistic which measures the difference between the two group means relative to the variability of the samples, and p-value which indicates the probability that the observed difference happened by chance. A small p-value (typically ≤ 0.05) indicates strong evidence against the null hypothesis, leading researchers to reject it and conclude the result is statistically significant. Conversely, a large

p-value (> 0.05) suggests weak evidence against the null hypothesis, and thus, it is not rejected.

In the context of definition modeling and evaluation of different PEFT configurations, the t-test is appropriate for several reasons. Firstly, the p-value will tell you whether the performance difference is statically significant or just due to random variation of data. Secondly, the t-tests are easy to implement via `scipy.stats`⁴ and the documentation is good.

3 Language and Dataset Selection

The selected target dataset for this study is in Norwegian and from DBnary (Sérasset, 2012). DBnary is a multilingual lexical resource extracted automatically from Wiktionary⁵, structured as a linked data resource in RDF format. It contains dictionary definitions, translations, and lexical relations for words across many languages. By leveraging Wiktionary’s rich, crowd-sourced content, DBnary provides a large-scale, openly accessible dataset ideal for tasks like definition modeling and cross-lingual lexical semantics. Its standardized format facilitates integration with other linguistic resources and supports multilingual natural language processing research. Table 1 shows one example from the training dataset where *target* means the word to be defined, *example* is the sentence containing the target word, *definition* is the dictionary-style definition, and *example_with_prompt* is the example prompt question to guide the model in generating definitions.

The dataset used in our experiments is divided into three parts: 4,522 samples in the training set, 314 in the validation set, and 716 in the test set. Although the overall size of the dataset is relatively modest compared to large-scale multilingual corpora, it still provides sufficient examples to evaluate the performance of models in low-resource language scenarios. The training data are used to optimize the model parameters, while the validation data are used to tune hyperparameters and prevent overfitting. The test set serves as a final evaluation benchmark for assessing the model’s performance.

This setup reflects a common situation in cross-lingual or multilingual NLP, where many languages, such as Norwegian, lack abundant labeled data. Working with such datasets poses challenges, but also highlights the importance of developing

⁵<https://www.wiktionary.org/>

target	example	definition	example_with_prompt
Polstring	Han arbeidet med polstring av møbler.	Det å polstre noe.	Han arbeidet med polstring av møbler. Hva betyr Polstring?

Table 1: An example entry from the dataset containing a target word, example sentence, definition, and prompt for definition generation.

Aspect	TowerInstruct-7B-v0.2	mt0-xl
Model Size	7 billion parameters	3.7 billion parameters
Architecture	Decoder-only transformer	Encoder-decoder transformer
Training Data	Fine-tuned on TowerBlocks ⁸	Fine-tuned on xP3 ⁹
Languages	English, Portuguese, Spanish, French, German, Dutch, Italian, Korean, Chinese, Russian	Multilingual support across many languages, designed for cross-lingual transfer
Strengths	Specialized for translation and related language processing tasks with fine-tuning	Strong zero/few-shot generalization across diverse NLP tasks

Table 2: Comparison between TowerInstruct-7B-v0.2 and mt0-xl

methods that are robust and effective under low-resource conditions.

4 Models

In this research, we chose two models from two different architectures to fine-tune, one is an encoder-decoder model named mt0-xl (Muenighoff et al., 2022), another is a decoder-only model named Unbabel/TowerInstruct-7B-v0.2 (Alves et al., 2024). Encoder-decoder models, also called sequence-to-sequence models, consist of two components encoder and decoder. The separation allows the encoder to fully understand the input before the decoder starts producing output, enabling better handling of long or complex inputs. This architecture is effective for tasks involving transforming one sequence to another. However, decoder-only models consist solely of a single transformer decoder stack. They generate output tokens one by one, using all previously generated tokens as context.

Table 2 shows the features of mt0-xl⁶ and TowerInstruct-7B-v0.2⁷.

⁶<https://huggingface.co/bigscience/mt0-xl>

⁷<https://huggingface.co/Unbabel/TowerInstruct-7B-v0.2>

⁸<https://huggingface.co/datasets/Unbabel/TowerBlocks-v0.1>

⁹<https://huggingface.co/datasets/bigscience/xP3>

¹⁰<https://huggingface.co/spaces/evaluate-metric/sacrebleu>

5 Experiments with LoRA

In our experiments with Low-Rank Adaptation (LoRA), we explored several key hyperparameters that significantly impact model performance and generalization ability:

- **LoRA Rank:** This parameter determines the rank of the LoRA matrices used in the model. Configuration 1 uses 256, Configuration 2 uses 512, and Configurations 3, 4, and 5 use different values ranging from 64 to 256.
- **LoRA Alpha:** It controls the scaling factor in the LoRA mechanism. The values for all configurations are relatively consistent with 512 or 1024.
- **LoRA Dropout:** This parameter helps to regularize the model, and values are either 0.1 or 0.2.

Table 3 presents the evaluation results for various LoRA configurations with a focus on the BLEU and BERTscore F1 metrics. It also includes p-values obtained through statistical significance testing, highlighting the best-performing configurations with red text. Metric scores are as follows:

- **BLEU:** The best BLEU score for mt0-xl (encoder-decoder model) is obtained in Configuration 3, with 6.183. The best BLEU

Parameter	Config 1	Config 2	Config 3	Config 4	Config 5
LoRA Rank	256	512	256	128	64
LoRA Alpha	512	512	1024	512	512
LoRA Dropout	0.1	0.1	0.1	0.1	0.2
Metric scores					
BLEU (mt0-xl)	5.77	5.947	6.183	5.96	5.655
BLEU (Tower-7B)	3.81	3.805	4.249	4.056	4.623
p-value(BLEU)	2.84e-15	2.05e-17	1.83e-13	9.89e-09	8.59e-05
BERTscore F1 (mt0-xl)	0.693	0.692	0.691	0.694	0.695
BERTscore F1 (Tower-7B)	0.670	0.671	0.675	0.676	0.682
p-value(BERT F1)	8.83e-14	7.39e-13	2.55e-07	1.11e-09	3.63e-05

Table 3: Evaluation Results for Different LoRA Configurations & Models on Norwegian Data (Best scores highlighted in red). mt0-xl: encoder-decoder model (mt0-xl 3.7B), Tower-7B: decoder-only model (Unbabel/TowerInstruct-7B-v0.2)

score for Tower-7B (decoder-only model) is obtained in Configuration 5, with 4.623.

- BERTscore F1: The best BERTscore F1 for models mt0-xl and Tower-7B are 0.695 and 0.682 respectively in Configuration 5.
- P-value: For BLEU, the p-values are extremely small, ranging from 2.84e-15 to 8.59e-05, indicating differences between the configurations. For BERTscore F1, the p-values are similarly very small, ranging from 8.83e-14 to 3.63e-05, confirming the statistical significance of the differences in performance between the configurations. A p-value below 0.05 suggests that the observed differences in the metric scores are statistically significant.

6 Experiments with IA3

Facing issues with memory, we tried changing the batch size to 4 and 8. Due to the different hyperparameters to be trained for the encoder-decoder model compared to the decoder-only model for IA3, we do not perform a t-test between them. The performance of these models are shown in Tables 4 and 5.

For the encoder-decoder model, we explored the effect of rpenalty (repetition penalty) and the number of beams in the beam search. For the decoder-only model, we were inspired by the different lengths of generated text in Table 6 and choose to investigate the effect of length penalties. We elaborate on the results in section 8.

7 Manual Analysis

In table 6, we compared predictions for 3 Norwegian words such as *rettferdig*, *parallell* and *nokon* from the dataset. The two models fine-tuned with LoRA produced relatively concise outputs, with definitions that were structurally sound, appropriately sized, and complete sentences. In contrast, the models fine-tuned using IA3 exhibited more extreme behavior: the decoder-only variant often generated overly lengthy definitions, while the encoder-decoder variant tended to produce overly brief outputs, sometimes reducing definitions to a single word.

Nokon, meaning “eit uspesifisert kvantum av” (an unspecified quantity of) according to actual definition, was handled with varying quality across models. The LoRA encoder-decoder model generated the definition “som har noko i seg” (something that has something in it) based on the sentence “Eg hadde i [[noko]] mjøl” (I had some flour). The model appeared to have learned from the context, but the output was essentially a paraphrase of the sentence rather than a true definition, failing to capture the precise quantificational meaning expressed in the actual definition. IA3 decoder-only performed poorly, hallucinating an incorrect definition by claiming that *nokon* translated to “kakerlakker” (cockroaches). This was a clear semantic error and indicated a major failure in factual grounding. The IA3 encoder-decoder generated only the word “noko”, which is a synonym of *nokon*, but it was insufficient as a definition. This suggested that

Parameter	Config 1	Config 2	Config 3	Config 4	Config 5
rpenalty	1.2	1.2	2.0	1.2	2.0
num_beams	1	2	2	4	4
Metric scores					
BLEU	0.14	0.12	0.13	0.19	0.18
BERTscore F1	0.645	0.643	0.644	0.641	0.643

Table 4: IA3 encoder-decoder model (mt0-xl) hyperparameter tuning

Parameter	Config 1	Config 2	Config 3	Config 4
rpenalty	1.1	1.1	1.5	1.5
length penalty	2.0	0.5	2.0	0.5
Metric scores				
BLEU	3.46	3.46	1.67	1.67
BERTscore F1	0.648	0.648	0.627	0.628

Table 5: IA3 decoder-only model (Unbabel/TowerInstruct-7B-v0.2) hyperparameter tuning

the model either failed to learn the task structure properly or was unable to generate a complete and meaningful output.

Parallell was also an interesting case because the actual definition given in the dataset was “om to eller flere linjer, overflater eller lignende” (about two or more lines, surfaces, or similar.) This definition pointed out that *parallell* involves two or more lines, surfaces, or similar objects, but it did not explain the relationship or property between them. In other words, it only listed the types of objects without describing how they are parallel. This suggested that the original data’s definition may be incomplete or insufficiently clear. The given sentence “to vektorer regnes som parallelle hvis og bare hvis det finnes et tall som kan ganges med den ene vektoren slik at man får den andre” (two vectors are considered parallel if and only if there exists a number that can be multiplied with one of the vectors to get the other), was already a definition. Both IA3 and LoRA fine-tuning methods for encoder-decoder models suffered from unclear and imprecise definitions. IA3’s version “hvis og bare hvis det finnes et tall som kan ganges med den” (if and only if there is a number that can be multiplied by it) lacked specific context, making it unclear and potentially confusing. LoRA’s version “som er parallellt forbundet med noen” (which is parallelly connected to someone), lacked specific meaning, that is, it is unclear what *parallell* refers to here.

8 Discussion

We conducted a series of experiments using two prominent PEFT techniques—LoRA and IA3—on a sequence-to-sequence dataset, evaluating their impact on both encoder-decoder (mt0-xl) and decoder-only (TowerInstruct-7B) architectures.

Effect of Hyperparameters: Both methods exhibited clear sensitivity to hyperparameter settings. For LoRA, performance improved when the rank and alpha parameters were better tuned. For instance, increasing alpha to 1024 (Config 3) while maintaining a moderate rank of 256 yielded the highest BLEU score on the encoder-decoder model (BLEU (mt0-xl) = 6.183). On the decoder-only model, lower ranks (e.g., 64 in Config 5) combined with higher dropout led to the best BERTscore and BLEU (BLEU (Tower-7B) = 4.623, BERT F1 = 0.682), indicating that lightweight configurations can still be effective. For IA3, we can see the original configuration of rpenalty=1.2 and 1 beam gives a marginally better BERTScore F1, but more importantly, increasing the number of beams to 4 gives a better BLEU score. For decoder-only, changing length penalty barely affected the results. However, the original rpenalty of 1.1 greatly outperformed a rpenalty of 1.5.

Encoder-Decoder vs Decoder-Only: The encoder-decoder model tends to have a slightly higher BERTscore F1 compared to the decoder-only model. This can be attributed to the fact that

Model		Rettferdig	Parallell	Nokon
		Sentence: Eksempellnol Innen politikken prøver man å fordele samfunnets goder rettferdig.	Sentence: To vektorer regnes som parallelle hvis og bare hvis det finnes et tall som kan ganges med den ene vektoren slik at man får den andre	Sentence: Eg hadde i [[noko]] mjøl
		Actual Definition: Som skjer på en rettferdig måte.	Actual Definition: Om to eller flere linjer, overflater eller lignende.	Actual Definition: Eit uspesifisert kvantum av.
LoRA	Decoder-only	Som er rett og slett, ikke urett.	Hvordan to ting er rettet mot hverandre, dvs. at de ligger i samme retning.	Nokon er en av flere synonymer for noe, og brukes som et relativt adjektiv.
LoRA	Encoder-decoder	Som er rett, riktig.	Som er parallellt forbundet med noen.	Som har noko i seg.
IA3	Decoder-only	Retteffertig er et adjektiv som beskriver en situasjon hvor dette ordet brukes, og hvis denne tilstanden kan være i overensstemming med de faktiske omgansomheter eller kravene på retterførtigheten av menneskerettene (human rights). Det vil si at alle personer har like verdier uavhengige fra deres raseri, religionsevangeli, seksualitet, sosial status osv., og skal behandles likestilt basert på disse grunner. Dermed innebærer riktforhold også ikke-discriminering mot nobe grupperinger inkludere kvinnersamfundets interesser, minoritetersinteresse etc..	Stående i samme retning eller rett ut fra hinanden, ikke korsede: «parallelt»;	Det er en norsk ord som kan oversettes til «kakerlakker».
IA3	Encoder-decoder	Equal	Hvis og bare hvis det finnes et tall som kan ganges med den	Noko

Table 6: Predicted definitions for “rettferdig”, “parallell” and “nokon” by different models. The columns are the 3 different words (and the sentence they appear in and actual definition), while the rows are the 4 models.

the encoder-decoder architecture processes the entire input sequence through the encoder first, providing it with a richer understanding of the context. This allows the decoder to generate more semantically accurate outputs, which might explain the marginal improvement in BERTscore F1. Although the decoder-only model has a slightly lower BERTscore F1, it is still within a comparable range, indicating that it can also capture the semantic content of the reference translations effectively. However, the lack of an encoder to fully process the input sequence might limit its ability to retain complex contextual information, which could result in a slightly lower semantic alignment.

Speed vs Quality Tradeoff: Regarding training efficiency, IA3 configurations were consistently faster to train than LoRA, due to fewer parameters being introduced and updated. However, LoRA achieved slightly better quality when carefully tuned, especially for BLEU. This reveals a tradeoff: IA3 is more efficient and easier to tune, whereas LoRA offers better ceiling performance if computational budget allows for broader hyperparameter exploration.

9 Conclusion

In this study, we explored the effects of two PEFT methods, LoRA and IA3, on both encoder-decoder and decoder-only architectures for sequence-to-sequence tasks. Our experiments demonstrated that hyperparameter tuning plays a critical role in optimizing performance for both approaches. LoRA showed potential for achieving higher translation quality when parameters like rank and alpha are carefully adjusted, while IA3 provided faster training and more stable results with fewer tuning demands.

Comparing architectures, the encoder-decoder model generally outperformed the decoder-only model in semantic accuracy, as evidenced by higher BERTscore F1, likely due to its ability to better contextualize input sequences. Nevertheless, decoder-only models remain competitive and effective in capturing semantic meaning.

Ultimately, the choice between LoRA and IA3 involves a tradeoff between training efficiency and maximum performance: IA3 excels in speed and simplicity, whereas LoRA offers higher peak quality given sufficient tuning resources. These findings can guide future fine-tuning efforts for adapting large language models in translation and related

tasks.

10 Further Research

An important direction for future research is experimenting with different foundational models. The baseline models used in current setups may not generalize well across all languages, especially those with unique morphological or syntactic properties. For example, multilingual models might underperform on specific languages due to limited representation or interference from unrelated languages. It would be valuable to investigate whether monolingual models, trained exclusively on high-quality data in the target language, offer better performance. Such exploration could lead to more robust and language-specific adaptations of semantic change modeling systems.

Given the different sentence lengths in figure 1, the datasets for different languages could be studied since they can provide valuable insights. Dictionaries compiled for different languages might have different styles. We see that Norwegian, Finnish and Italian have sentences with much fewer characters than the others.

Definition modeling for Nynorsk is hitherto unexplored, and worth exploring. Unfortunately, the authors are only familiar with one to three of the languages given. In addition to stylistic and structural variation, future research could examine the impact of out-of-vocabulary (OOV) rates, particularly for low-resource languages like Norwegian.

Another avenue is to analyze the distribution of target-word senses, including how frequently polysemous words appear and whether certain senses dominate, which may influence model behavior and evaluation.

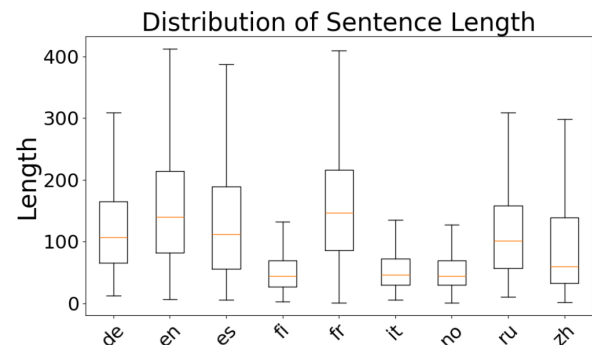


Figure 1: Distribution of sentence lengths (number of characters) for different languages

This work has so far only explored LoRA and IA3 as PEFT methods. However, the field of

parameter-efficient fine-tuning is rapidly evolving, and many other PEFT methods can be explored. Approaches such as prefix tuning, prompt tuning, adapters, and orthogonal tuning (OFT and BOFT) could offer alternative trade-offs between parameter count, training stability, and performance. These methods may perform differently across language families or in low-resource settings.

References

- Duarte M. Alves, José Pombal, Nuno M. Guerreiro, Pedro H. Martins, João Alves, Amin Farajian, Ben Peters, Ricardo Rei, Patrick Fernandes, Sweta Agrawal, Pierre Colombo, José G. C. de Souza, and André F. T. Martins. 2024. [Tower: An open multilingual large language model for translation-related tasks](#). *Preprint*, arXiv:2402.17733.
- Zeinab Borhanifard, Heshaam Faili, and Yadollah Yaghoobzadeh. 2025. [Combining replay and lora for continual learning in natural language understanding](#). *Computer Speech Language*, 90:101737.
- N. A. C. Cressie and H. J. Whitford. 1986. [How to use the two sample t-test](#). *Biometrical Journal*, 28(2):203–210.
- Mariia Fedorova, Timothee Mickus, Niko Partanen, Janine Siewert, Elena Spaziani, and Andrey Kutuzov. 2024. [AXOLOTL’24 shared task on multilingual explainable semantic change modeling](#). In *Proceedings of the 5th Workshop on Computational Approaches to Historical Language Change*, pages 72–91, Bangkok, Thailand. Association for Computational Linguistics.
- Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. [Parameter-efficient fine-tuning for large models: A comprehensive survey](#). *arXiv preprint arXiv:2403.14608*. Version 7, last revised 16 Sep 2024.
- Michael Hanna and Ondřej Bojar. 2021. [A fine-grained analysis of BERTScore](#). In *Proceedings of the Sixth Conference on Machine Translation*, pages 507–517, Online. Association for Computational Linguistics.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. [Lora: Low-rank adaptation of large language models](#). *arXiv preprint arXiv:2106.09685*.
- Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohata, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022. [Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning](#). *arXiv preprint arXiv:2205.05638*.
- Timothee Mickus, Kees Van Deemter, Mathieu Constant, and Denis Paperno. 2022. [Semeval-2022 task 1: CODWOE – comparing dictionaries and word embeddings](#). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 1–14, Seattle, United States. Association for Computational Linguistics.
- Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, and 1 others. 2022. [Crosslingual generalization through multitask finetuning](#). *arXiv preprint arXiv:2211.01786*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. [Bleu: A method for automatic evaluation of machine translation](#). In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.
- Gilles Sérasset. 2012. [Dbnary: Wiktionary as a lmf based multilingual rdf network](#). In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC’12)*, pages 2466–2472, Istanbul, Turkey. European Language Resources Association (ELRA).
- Haifeng Wang, Hua Wu, Zhongjun He, Liang Huang, and Kenneth Ward Church. 2022. [Progress in machine translation](#). *Engineering*, 18:143–153.
- Wenhao Yu, Chenguang Zhu, Zaitang Li, Zhiting Hu, Qingyun Wang, Heng Ji, and Meng Jiang. 2022. [A survey of knowledge-enhanced text generation](#). *ACM Comput. Surv.*, 54(11s).
- Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. [BERTScore: Evaluating Text Generation with BERT](#). In *International Conference on Learning Representations (ICLR)*. Available at OpenReview.

3 Pretraining Tiny Language Models track

1 hour to train, time constrained pre-training for LLMs

Greg Taube

Department of Informatics

University of Oslo

gregorjt@uio.no

Abstract

Current SOTA language models require long training runs on clusters of GPUs, placing them out of reach for most researchers and students. Inspired by BabyLM shared task, we conducted experiments with encoder-only architectures and hyperparameter settings, while restricting training time to 1 hour, and training hardware to 1 GPU. We also fine-tune on COLA, one of the benchmarks used in GLUE (General Language Understanding Evaluation). We find that: Our variations of the baseline encoder was not improved. We were not able to reproduce “superconvergence” with the onecycleLR learning rate scheduler. Code, configs and checkpoints to support reproducibility and further research on compute-efficient pretraining will be available on Github.

1 Introduction

Large language models (LLMs) has over the last years been used to solve increasingly harder tasks, and new benchmarks are created (Chollet et al., 2025) as the old benchmarks get “solved”. Part of the success is not due to new architectures improving upon the transformer (Vaswani et al., 2023), but simply using more data, and training for a longer time (Kaplan et al., 2020). As more companies start training, fine-tuning and deploying/using LLMs (inference), focusing on efficiency becomes even more important for the following reasons:

- Cost of compute, mostly compute time on GPUs, becomes prohibitively expensive
- Environmental costs, including CO2 emissions and data center footprint
- Democratizing LM research

Research from “open” models like Llama (Touvron et al., 2023) shows us that the cost of compute

is of such an order that it effectively blocks most researchers from competing in creating the “best” LLMs. Thus the industry is in need of challenges that makes it possible for all scientists, regardless of budget, to contribute to innovations in the LLM space. BabyLLM (Hu et al., 2024) is such a challenge, and we will be exploring a variant with the following changes:

- Compute time limited to 1 h
- Training on a single GPU
- Unlimited data

Note that due to the time limit, we do have a data limit in practice, as we will have to spend the 1 hour wisely, preferably on high quality data.

Our goal in this paper is to try to achieve good results, train the models fast and keeping the model as tiny as possible.

For pre-training, we will use the same dataset as was used for the best model in the 2024 BabyLLM task (Charpentier and Samuel, 2024) ¹.

Finally, we will finetune on the COLA ² where we use MCC (Matthews, 1975) as the evaluation metric.

Specifically we will try to achieve fast learning by trying to reproduce the “superconvergence” described in the paper:

Super-Convergence: Very Fast Training of Neural Networks Using Large Learning Rates (Smith and Topin, 2018).

The learning rate scheduler described in this paper is implemented in **torch.optim.lr_scheduler.OneCycleLR**.

We will test with different amounts of max steps, while ensuring training time remains under

¹ Available at: <https://huggingface.co/datasets/lgt/babyllm-2024-baby-cosmo-fine-10m>

² Available at: <https://huggingface.co/datasets/nyu-mll/glue/viewer/cola>

1h, to see if we can see something that looks like “superconvergence” in under 1 hour of training.

2 Background and Related Work

We will first discuss the babyLM challenge which directly inspired this research, and then discuss research on small/efficient LLMs in general, which is a broader topic than the more focused babyLM task.

2.1 BabyLM

As described in (Hu et al., 2024) “BabyLM Challenge is a community effort to close the data-efficiency gap between human and computational language learners”. According to the babyLM papers and website (babylm.github.io), a 13 year old has generally been exposed to less than 100 million words, while GPT-3 (Brown et al., 2020) was trained with 200 billion words. These numbers seen in isolation ignores that vision and other senses play a role in learning. This is also noted in the paper:

Non-linguistic information, such as visual input, potentially plays a large role in child language acquisition

And the challenge has been expanded with a multi-modal track to address this issue.

The 2024 edition broadened the rules for the *Strict* and *Strict-Small* text tracks: teams could assemble *any* corpus, provided it stayed under the 100 M- or 10 M-word limits. As explained in the paper:

The motivation behind this decision is that pretraining data quality has been linked to large improvement gains in at-scale language models(Hu et al., 2024).

Many high-scoring entries therefore curated their own dataset to improve the models.

From the results of the two text tracks (10M words and 100M words), we can see that we reconfirm two “facts” about training LLMs:

- More data helps: Models trained on 100M words performs better than models trained on 10M words
- Longer training time helps: Hyperparameters of leaderboard winners have higher number of FLOPS / model GPU hours

Despite the low-data setting, the organisers still observed a nearly log-linear correlation between total training FLOPs and average benchmark score across 64 submissions (Hu et al., 2024). Compute is therefore *not* a solved problem: given only 1 GPU hour, we cannot expect our model to reach parity with the models that trained for several hundred hours.

For instance, the winner of BabyLM 2024, gpt-bert-strict (Charpentier and Samuel, 2024), trained the winner model for 570 hours. This means, with 1 hour, our model architecture would have to be significantly more efficient in training to reach gpt-bert-strict’s performance.

Another take-away was that architectural changes alone were rarely decisive. The strongest systems combined **multiple training objectives**. The overall winner, GPT-BERT, combined causal and masked language modelling (MLM) (Devlin et al., 2019) in a 1:7 ratio, while other successful teams alternated objectives epoch-by-epoch (Hu et al., 2024). One interesting from the meta-analysis of the submissions, was that:

Our interpretation of this result is that there are likely benefits from certain backbone architectures, but that these effects might not be strong enough to be picked up in a statistical analysis of 64 models. Interestingly, recent work has noted that different architectures and training setups often tend to converge to neural representations with similar properties and capabilities

The paper further suggests that there is a trade-off between high scores and how easy it is to optimize the model and have low variance.

2.2 Other small/efficient LM research

In addition to the research effort incentivized by the babyLM challenge, there is also interest from many competitors in the “AI race” in small models, for differing reasons. Apple, which needs efficient LLMs to run locally on millions of iPhones, released OpenELM (Mehta et al., 2024). Allen Institute released Olmo (Groeneveld et al., 2024) to “Accelerating the Science of Language Models”. There are many others, and many of the AI labs provide models of differing sizes to make them possible to run in different compute environments. For example Gemma 3 (Team et al., 2025) is provided in 1B, 4B, 12B and 27B. In addition

to the number of parameters, the models are often provided at different quantization levels:

Along with the raw checkpoints, we also provide quantized versions of our models in different standard formats

Quantization is compression by using lower precision numbers. We won't discuss it further or use it in this work, but note that research on this like QLoRA (Dettmers et al., 2023) is also an interesting research direction.

The OpenELM paper architecture section gives a list of techniques used in "state-of-the-art" LLMs, which we repeat here:

- do not use learnable bias parameters in any fully-connected
- Pre-normalization using RMSNorm
- Rotary positional embedding
- grouped query attention (GQA)
- SwiGLU FFN
- Flash Attention
- non-uniform allocation of parameters across layers using layer-wise scaling

We will explore some of these techniques in our model.

Finally we want to mention MoE models like Mixtral (Jiang et al., 2024), which provides insight into how we can decompose models in such a way that we can use smaller fewer active parameters under inference. Inference compute is one of the major costs for providers of LLM APIs and might surpass pre-train compute over time.

2.3 Fast training and convergence

In this paper, we are not only interested in training small models, but also in speed of training. A major factor contributing to the speed of training is how fast one goes down the loss landscape. The goal is to use as few steps as possible to reach the lowest point possible.

To achieve this goal we have optimizers and learning rate schedulers, that can work together to efficiently "navigate" the loss landscape, while:

- Avoid being stuck in local minima

- Avoid overshooting (achieved by annealing/decay)
- Preferably reach low loss in few steps
- Most importantly, converge towards the global minimum, even if using longer time, more steps

Some interesting approaches might be variations of Adam (Kingma and Ba, 2017) and LION (Chen et al., 2023), but in this paper we will as mentioned in the introduction focus on large learning rates for "superconvergence" (Smith and Topin, 2018). We will not investigate LION in this paper.

2.4 Superconvergence

The paper mentioned above describe the concept as:

a phenomenon, which we named "super-convergence", where neural networks can be trained an order of magnitude faster than with standard training methods

They further elaborate:

One of the key elements of super-convergence is training with one learning rate cycle and a large maximum learning rate. A primary insight that allows super-convergence training is that large learning rates regularize the training, hence requiring a reduction of all other forms of regularization in order to preserve an optimal regularization balance

The paper also provides a method to derive an estimate of the optimal learning rate. We do not use this method in this paper, but it would be interesting to test out in future work.

The empirical tests showcase that the phenomenon on a "variety of datasets and architectures", but it remains to be seen if we can see similar results generalize across tests different from what the paper explored (primarily image datasets).

The author also sees a boost in performance compared to standard training when the amount of labelled training data is limited. This seems to indicate that it might be a good fit for challenges similar to babyLM:

Demonstrates that the effects of super-convergence are increasingly dramatic when less labeled training data is available.

3 Data

3.1 Pre-training Corpus

We pre-train exclusively on the same corpus that was used for the best performing model on the babyLM challenge.³

The dataset contains:

- **BabyLM core corpus:** dialogue-heavy, child-directed material extracted from CHILDES, spoken British National Corpus, Simple English Wikipedia, public-domain children’s stories, OpenSubtitles and Switchboard.
- **FineWeb-Edu:** curated educational web-pages
- **Cosmopedia:** synthetic textbooks, blog-posts, stories, posts and WikiHow articles⁴

The BabyLM core corpus is described in detail in (Hu et al., 2024). FineWeb-Edu was created by HuggingFace and is a subset of the FineWeb dataset (Penedo et al., 2024). The Cosmopedia dataset is described in detail in (Ben Allal et al., 2024).

We use this dataset for the same reasons stated in the GPT or Bert paper (Charpentier and Samuel, 2024):

The main purpose of training on this mixture is to provide the model with more factual knowledge and more diverse language.

3.2 Fine-tuning Dataset: CoLA

For downstream evaluation we use the **Corpus of Linguistic Acceptability (CoLA)** from GLUE (Wang et al., 2019). CoLA contains 8.5 k English sentences annotated as *grammatical* or *ungrammatical*. We choose CoLA because:

- It tests the models “understanding” of grammar.

³Available on HuggingFace at <https://huggingface.co/datasets/litg/babylm-2024-baby-cosmo-fine-10m>.

⁴<https://github.com/huggingface/cosmopedia>

- The dataset is small, so fine-tuning completes fast
- Matthews correlation (MCC) provides a good metric that is affected too much by an imbalanced dataset

4 Experiments

This section reports all training settings and outcomes. We first describe the baseline configuration, then introduce a single variation that replaces the cosine scheduler with the One-Cycle LR policy. We do pre-training with the dataset mentioned in the previous section and fine-tuning using PEFT (Mangrulkar et al., 2022)

4.1 Baseline experimental setup

We will use a hyperparameter setup without “superconvergence” LR scheduler so we have a baseline to compare against. The configuration is described below and uses an encoder architecture.

Parameter	Value
Hidden size	192
Intermediate size	512
Layers	12
Attention heads	3
Max sequence length	128
Activation (FFN)	GEGLU (Shazeer, 2020)
Activation (classifier)	GELU
Batch size	256
Dropout (all) (Hinton et al., 2012)	0.1
Positional encoding	Learned relative
Training steps	10,000

Table 1: Baseline hyper-parameters.

The optimizer used is AdamW with weight-decay selectivity, decay on weight matrices, none on biases/LayerNorm.

For the positional encoding we use a variation of “relative positional representations” (Shaw et al., 2018) with buckets which length is specified by a positional bucket size hyperparameter set to 32, similar to the T5 (Text-To-Text Transfer Transformer) paper (Raffel et al., 2023)

4.1.1 Baseline pre-training results

Metric	Value
Steps	10,000
Loss	5.05
Accuracy	16.9%
Elapsed time	36.5 minutes

Table 2: Results of the baseline MLM pre-training run (10 k steps).

The pre-training run ended up with a loss of 5.05 with only 16.9 % accuracy. This relatively high loss and low accuracy are expected given the very small size and training time of the network. We would have seen further reduction in loss and increased accuracy if we trained for more steps.

For a complete view of how the loss and accuracy evolved over time, we refer to the full SLURM logfile can be found in the Github repo mentioned in the implementation details section.

4.1.2 Baseline fine-tuning results

Split	Best MCC	Epoch
Train	0.4250	29
Validation	0.1417	19

Table 3: LoRA-based fine-tuning results on CoLA (class-weighted loss, rank 32, top 4 layers unfrozen).

Even with class-balanced sampling and a moderately sized LoRA adapter (rank 32), the tiny encoder only reaches a validation MCC of 0.1417 after 19 epochs. This gap between train (0.425) and validation clearly shows that the model does not generalize well. However, achieving a non-zero MCC confirms the model has at least learned to distinguish acceptable from unacceptable sentences better than chance, thus providing a baseline for subsequent scheduler or architectural variations.

We will now see if we can get better results with “superconvergence”.

4.2 One-Cycle LR variation

We swap the cosine schedule for ONECYCLELR with the following parameters:

- max_lr = 0.01
- default values of all other parameters ⁵

All other settings remain the same.

Then we ran the same code, for the same number of steps.

4.3 Results with One-Cycle LR.

Unfortunately, as can be seen by the results table, the loss and accuracy from pre-training was worse. See slurm-1615283-onecycleLR.out for the full training logs.

Metric	Value
Steps	10,000
Loss	5.45
Accuracy	12.66%
Elapsed time	38.5 minutes

Table 4: Results of the variation using oneCycleLR

4.3.1 One-Cycle LR fine-tuning results

We do the same finetuning as for the baseline, and get the following results, which is clearly worse than the baseline.

Split	Best MCC	Epoch
Train	0.1503	29
Validation	0.0887	19

Table 5: LoRA-based fine-tuning results on CoLA for the onecycleLR model

4.4 Training and finetuning with more steps

We increased the step size to 15 000, and got the following results for the baseline

Metric	Value
Steps	15,000
Loss	4.91
Accuracy	18.69%
Elapsed time	57.5 minutes

Table 6: Baseline architecture increased max_steps

When increasing to 15 000 steps, it seems like the onecycle variation is improving faster than the baseline. Since increasing max steps further would lead to more than 1 hour of training, we will leave further experiments to future work, to see if the onecycle variation can overtake the baseline.

Finetuning the models gives the following results:

Since we finetune with the same settings as described in the finetuning section, and we get better results, that indicates the “base models” has improved. It is good reason to believe that the models trained on 15 000 steps in general will perform better on downstream tasks it is finetuned on.

4.5 Summary

Unfortunately, we did not see better results in either pre-training, nor finetuning, when compared to the baseline. Thus in our case it does not seem like the onecycle learning rate scheduler is better. This seems to be true for all step sizes tested.

We are not sure if this is due to misconfiguration or a misunderstanding of how the different hyperparameters work together. It might also be the case

⁵Pytorch’s OneCycleLR

Metric	Value
Steps	15,000
Loss	5.05
Accuracy	17.24%
Elapsed time	54.5 minutes

Table 7: onecycle variation with increased max_steps

Split	Best MCC	Epoch
Train	0.7033	29
Validation	0.1717	19

Table 8: LoRA-based fine-tuning results on CoLA baseline 15k steps

that we simply trained for too little time to realistically see a significant difference, independent of architecture changes.

As discussed in the background, super-convergence means “neural networks can be trained an order of magnitude faster than with standard training methods”. We did not see this in our experiments and we suggest further work to more closely examine the original paper, for instance try out their method to find an estimate of the optimal learning rate and try it instead of the “guess” we did in this paper.

5 Limitations

We cover English only and evaluate on a narrow set of tasks. Real-world applicability to generative use-cases and multi-linguality was not explored.

Given that the current leader of the GLUE leaderboard is a multi-lingual model (Patra et al., 2022) we believe that multi-lingual models might have a better language-understanding than mono-lingual ones.

6 Future Work

In this paper we have discussed data-efficiency, meaning learning the same or more from less data. Training compute efficiency, to update the weights “towards intelligence” faster. And finally fine-tuning efficiency, for updating an existing model in such a way that it performs better at a particular task faster.

We think that future work should also look at producing smaller models in terms of parameter counts, an interesting idea in this regard is the “Lottery ticket hypothesis” (Frankle and Carbin, 2019). The idea is that large neural networks contains subnetworks that performs as good as the network as a whole. Thus with smart weight pruning one can reduce the parameter count consider-

Split	Best MCC	Epoch
Train	0.4186	29
Validation	0.1423	19

Table 9: LoRA-based fine-tuning results on CoLA for the onecycleLR 15k steps

ably without affecting performance. We hypothesize that this pruning can be done during training, which will save computation time since “dead weights” are not updated.

Improvements to the transformer architecture, like Multi-head Latent Attention (DeepSeek-AI et al., 2025) would also be interesting for future research.

Finally, we want to mention that research into mechanistic interpretability and explainable AI in general, is something that will lead to better models in “all” ways.

An analogy here is complicated software systems: A developer that understands a codebase, can more easily see which parts can be removed (make codebase smaller) and how to build upon the existing system (new features to a legacy system). While a new developer on a big legacy project cannot change much without breaking the system. We think something similar applies to AI, and that there are massive gains waiting for us if we **really** understands how LLMs work. An example of this kind of research is the research on superposition by Anthropic (Elhage et al., 2022).

Acknowledgements

We thank the IN5550 teaching staff for code and datasets and the track chair for help.

7 Implementation Details

Training and evaluation scripts are available at: [available in final manuscript]

References

- Loubna Ben Allal, Anton Lozhkov, Guilherme Penedo, Thomas Wolf, and Leandro von Werra. 2024. *Cosmopedia*.
- Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. *Language models are few-shot learners*. *Preprint*, arXiv:2005.14165.

- Lucas Georges Gabriel Charpentier and David Samuel. 2024. [Gpt or bert: why not both?](#) *Preprint*, arXiv:2410.24159.
- Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Yao Liu, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, Yifeng Lu, and Quoc V. Le. 2023. [Symbolic discovery of optimization algorithms](#). *Preprint*, arXiv:2302.06675.
- Francois Chollet, Mike Knoop, Gregory Kamradt, and Bryan Landers. 2025. [Arc prize 2024: Technical report](#). *Preprint*, arXiv:2412.04604.
- DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, and 181 others. 2025. [Deepseek-v3 technical report](#). *Preprint*, arXiv:2412.19437.
- Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. [Qlora: Efficient finetuning of quantized llms](#). *Preprint*, arXiv:2305.14314.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of deep bidirectional transformers for language understanding](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. 2022. Toy models of superposition. *Transformer Circuits Thread*. https://transformer-circuits.pub/2022/toy_model/index.html.
- Jonathan Frankle and Michael Carbin. 2019. [The lottery ticket hypothesis: Finding sparse, trainable neural networks](#). *Preprint*, arXiv:1803.03635.
- Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Raghavi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, and 24 others. 2024. [Olmo: Accelerating the science of language models](#). *Preprint*, arXiv:2402.00838.
- Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. 2012. [Improving neural networks by preventing co-adaptation of feature detectors](#). *Preprint*, arXiv:1207.0580.
- Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Leshem Choshen, Ryan Cotterell, Alex Warstadt, and Ethan Gotlieb Wilcox, editors. 2024. [The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning](#). Association for Computational Linguistics, Miami, FL, USA.
- Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, L  lio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, and 7 others. 2024. [Mixture of experts](#). *Preprint*, arXiv:2401.04088.
- Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. [Scaling laws for neural language models](#). *Preprint*, arXiv:2001.08361.
- Diederik P. Kingma and Jimmy Ba. 2017. [Adam: A method for stochastic optimization](#). *Preprint*, arXiv:1412.6980.
- Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. Peft: State-of-the-art parameter-efficient fine-tuning methods. .
- B.W. Matthews. 1975. [Comparison of the predicted and observed secondary structure of t4 phage lysozyme](#). *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442–451.
- Sachin Mehta, Mohammad Hossein Sekhavat, Qingqing Cao, Maxwell Horton, Yanzi Jin, Chenfan Sun, Iman Mirzadeh, Mahyar Najibi, Dmitry Belenko, Peter Zatloukal, and Mohammad Rastegari. 2024. [Openelm: An efficient language model family with open training and inference framework](#). *Preprint*, arXiv:2404.14619.
- Barun Patra, Saksham Singhal, Shaohan Huang, Zewen Chi, Li Dong, Furu Wei, Vishrav Chaudhary, and Xia Song. 2022. [Beyond english-centric bitexts for better multilingual language representation learning](#). *Preprint*, arXiv:2210.14867.
- Guilherme Penedo, Hynek Kydl  cek, Loubna Ben al-lal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. [The fineweb datasets: Decanting the web for the finest text data at scale](#). *Preprint*, arXiv:2406.17557.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. [Exploring the limits of transfer learning with a unified text-to-text transformer](#). *Preprint*, arXiv:1910.10683.
- Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. [Self-attention with relative position representations](#). *Preprint*, arXiv:1803.02155.

- Noam Shazeer. 2020. [Glu variants improve transformer](#). *Preprint*, arXiv:2002.05202.
- Leslie N. Smith and Nicholay Topin. 2018. [Super-convergence: Very fast training of neural networks using large learning rates](#). *Preprint*, arXiv:1708.07120.
- Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. [Gemma 3 technical report](#). *Preprint*, arXiv:2503.19786.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. [Llama: Open and efficient foundation language models](#). *Preprint*, arXiv:2302.13971.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. [Attention is all you need](#). *Preprint*, arXiv:1706.03762.
- Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. [Glue: A multi-task benchmark and analysis platform for natural language understanding](#). *Preprint*, arXiv:1804.07461.

The Impact of Masking Rate and Strategy on Tiny Language Models

Henrik Johansen*

Department of Informatics
University of Oslo
hjo@uio.no

Hannah Scheufele*

Department of Mathematics
University of Oslo
hannarsc@uio.no

Tharusan Julian*

Department of Informatics
University of Oslo
tharusaj@uio.no

Abstract

Recent work on pre-training tiny language models has demonstrated that even with limited data and computational budgets, meaningful linguistic competence can be achieved. A commonly used approach for encoder-based models is Masked Language Modeling (MLM), which typically employs a fixed masking rate of 15%. In this study, we investigate the impact of varying the masking rate during pretraining on the downstream performance of tiny language models. Specifically, we train encoder-based transformers with different masking probabilities and evaluate their generalization abilities across both fine-tuning tasks and zero-shot linguistic benchmarks. Our results reveal interesting performance differences across applications: masking rate adjustments have significant impact on fine-tuning accuracy (up to 6.42% variation between masking rates), with span masking showing greater sensitivity to rate selection than random token masking. However, the same adjustments produce substantial effects on zero-shot linguistic capabilities as measured by BLiMP (variations up to 5.73%). These findings indicate that the importance of masking rate optimization depends on the intended application.

1 Introduction

The success of large-scale language models has led to increased research into their architectures, training regimes, and scaling laws. However, the large costs associated with training and deploying these models have motivated a parallel line of research focusing on tiny language models, which are models trained under limited resource constraints with a small number of parameters. This paper is part of the "Pretraining of Language Models Under Time Constraint" track (Charpentier, 2025), which challenges to develop efficient language models within a restricted training time budget (e.g., under

24 hours on a single GPU). The track is inspired by the BabyLM challenge (Warstadt et al., 2023; Hu et al., 2024), which emphasizes sample-efficient learning on developmentally plausible corpora.

A central component of pretraining encoder-only language models is the Masked Language Modeling (MLM) objective, originally introduced in BERT (Devlin et al., 2019). In MLM, a percentage of input tokens are masked and the model is trained to predict the original values. While a masking rate of 15 % has become the standard, it is not obvious whether this value is optimal for tiny models under strict time constraints. A masking rate that is too low may provide an insufficient learning signal, while a rate that is too high may obscure too much of the input, making the prediction task unnecessarily difficult (Wettig et al., 2023). In this paper, we systematically explore how varying the masking rate during pretraining affects the performance of tiny language models. We train encoder models with different values of the masking hyperparameter, ranging from 10 % to 50 %, using two masking strategies-random masking by Devlin et al. (2019) and span-based masking as in SpanBERT by Joshi et al. (2020) - and evaluate them on both a fine-tuned classification task and zero-shot linguistic benchmarks such as BLiMP and LAMBADA. In this exploration we aim to answer the question: Is the conventional 15 % masking rate still optimal when training small models with limited resources?

2 Related Work

Previous studies have begun to question the conventional use of a 15 % masking rate. Wettig et al. (2023) systematically examined masking rates across various model sizes and found that larger models tend to benefit from higher masking rates up to 40 %, and that even extreme rates like 80 % can retain most downstream performance despite high perplexity. However, their work focused pri-

*Equal contribution

marily on large-scale models, leaving the question of how masking rates interact with optimization and generalization in tiny models under resource constraints open.

Another relevant line of research investigates the selection of masked content during pretraining. SpanBERT by [Joshi et al. \(2020\)](#) proposed span-based masking to encourage learning representations for contiguous text segments, showing significant gains on span-sensitive tasks such as question answering and conference resolution. This strategy contrasts with the uniform random masking used in BERT, and further suggests that both the rate and structure of masking influence pretraining effectiveness. [Wettig et al. \(2023\)](#) also found that optimal masking rates can vary depending on the masking strategy, where uniform masking may benefit from higher rates than span-masking, implying an interaction between rate and strategy that has not been fully explored in low-resource settings.

Our work extends on this research by evaluating both the masking rate and masking strategy in the context of tiny models trained under time constraints.

3 Background

3.1 Masked Language Modeling (MLM)

Masked Language Modeling (MLM) is a self-supervised learning objective commonly used to pretrain large language models (LLMs), particularly those based on transformer encoder architectures. Introduced in BERT ([Devlin et al., 2019](#)), MLM involves randomly masking a subset of input tokens and training the model to predict the original tokens based on their surrounding context. This objective enables the model to learn deep bidirectional representations of language, as it can attend to both the left and right context of each masked token. In a typical MLM setup, as described by [Devlin et al. \(2019\)](#), a fixed percentage of input tokens—commonly 15%—are selected for masking. Of these, 80% are replaced with a special [MASK] token, 10% with a random token from the vocabulary, and 10% are left unchanged. This strategy encourages the model to develop robust representations that generalize beyond the pretraining objective. In particular, the random substitutions and unmasked tokens help prevent the model from becoming overly reliant on the [MASK] token as a prediction cue, which is absent during downstream fine-tuning tasks. The model is then trained

to predict the original token at all masked positions, while unmasked tokens are ignored in the loss computation. This setup balances the need for supervision with the preservation of input context.

3.2 Evaluation Metrics

Evaluating the performance of language models is inherently complex, as there can often be multiple correct answers. To evaluate the impact of different masking rates on tiny encoder-based language models, we employ both zero-shot and fine-tune evaluation metrics. Specifically, we evaluate on two widely used benchmarks, LAMBADA ([Paperno et al., 2016](#)) and BLiMP ([Warstadt et al., 2020](#)), as well as a supervised downstream task to measure transferability.

3.2.1 BLiMP

The Benchmark of Linguistic Minimal Pairs (BLiMP) is a diagnostic data set designed to evaluate the understanding of a model of English grammar through minimal pair sentence comparisons. Each example in BLiMP consists of two nearly identical sentences, one grammatical and one ungrammatical, differing in a single syntactic or morphological aspect. The model is then expected to assign a higher likelihood to the correct sentence in each pair ([Warstadt et al., 2020](#)).

BLiMP is composed of 67 sub-datasets, each containing 1000 sentence pairs, covering 12 linguistic phenomena, including subject-verb agreement, anaphore resolution, and argument structure. The overall accuracy is computed as the proportion of times the model selects the grammatical sentence across all pairs. Because this is a zero-shot evaluation, it provides insight into the general linguistic knowledge the model has acquired during pre-training, without further fine-tuning.

3.2.2 LAMBADA

Language Modeling Broadened to Account for Discourse Aspects (LAMBADA) by [Paperno et al. \(2016\)](#) tests a model’s ability to understand broad-discourse-level context. Each sample includes a multi-sentence context followed by a final sentence where the last word is masked. The model must predict the missing word, which typically cannot be inferred from the final sentence alone, but is recoverable when the full context is considered.

This benchmark challenges the model’s long-range dependency tracking, and is particularly relevant in settings where masking too much or too lit-

tle might impair the model’s contextual reasoning. LAMBADA reports both accuracy, the proportion of correctly predicted final words, and perplexity, which reflects the model’s confidence. A lower perplexity suggests more confident and coherent predictions, while higher accuracy indicates better contextual understanding.

3.2.3 Fine-Tuning Metric

In addition to zero-shot tasks, we evaluate each model on a fine-tuned classification task to assess how well learned representations transfer to downstream supervised settings. To do this, we fine-tune each model on the SST-2 dataset (Stanford Sentiment Treebank 2), which is a binary sentiment classification task. In SST-2, the model is required to classify the sentiment of a sentence as either positive or negative, making it a suitable test of a model’s ability to encode and use semantic information (Socher et al., 2013). We report accuracy as the primary metric, which reflects the percentage of correct predictions. This is done as the SST-2 dataset is already a balanced classification task.

3.3 Pretraining Dataset

For pretraining, we use the BabyLM 2024 10M corpus, which is a dataset consisting of 10 million words of child-directed and developmentally appropriate English. The data set is compiled from a diverse set of sources, including the British National Corpus (BNC Consortium, 2007), CHILDES (MacWhinney, 2000), Project Gutenberg (Gerlach and Font-Clos, 2020), OpenSubtitles (Lison and Tiedemann, 2016), and Switchboard (Godfrey et al., 1992). These sources collectively span various domains, such as dialogue, fiction, nonfiction, and conversations (Hu et al., 2024). This dataset is specifically designed to approximate the linguistic exposure of a child language learner, making it ideal for evaluating the sample efficiency and generalization abilities of tiny language models.

4 Method

4.1 Transformer Encoder Models

Transformer architectures have become the foundation of modern language models due to their ability to model complex dependencies through attention mechanisms. Originally introduced in the context of machine translation by Vaswani et al. (2017), the transformer consists of two main components: an encoder and a decoder. However, for tasks involving MLM, only the encoder is necessary as it

enables bidirectional attention over the input sequence. This is in contrast to decoder-only models, which are trained autoregressively and only attend to past tokens.

In our study, we focus extensively on the encoder model, which is better suited for MLM pre-training, as reported in Qorib et al. (2024), and more efficient to train within the given resource constraints. The encoder learns to predict randomly masked tokens using context from both directions in the input sequence. Varying the masking rate or the proportion of tokens masked directly affects the encoder’s learning behavior, which makes it the most relevant architecture for our study. Figure 1 illustrates the architecture of the transformer encoder model.

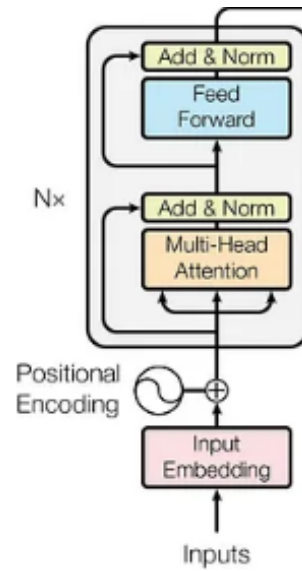


Figure 1: Transformer encoder architecture. (Vaswani et al., 2017)

The input tokens are first mapped to dense vector representations through an embedding layer, which are then combined with positional encodings to preserve word-order information. In our implementation, we use relative positional encoding, which is more parameter-efficient and is better suited for small-scale models compared to absolute position embeddings (Huang et al., 2020).

The resulting representations are passed through a stack of identical encoder layers, each consisting of a multi-head self-attention mechanism followed by a feed-forward network. Both components incorporate residual connections and layer normalization to enhance stability and convergence during training.. This architecture allows each token to attend to all others in the sequence, making it well-suited for bidirectional context modeling, which is

a key requirement for masked language modeling (Devlin et al., 2019).

For our implementation we follow the official preprocessing pipeline provided by Charpentier (2025), which includes normalization, sentence segmentation, and training a custom WordPiece tokenizer. The resulting sequences are tokenized and then cached with a maximum length of 128 tokens. These cached sequences are used as input for masked language model training.

Our model implementation consists of 12 encoder layers with an embedding dimension of 192. Each encoder layer contains a self-attention mechanism with query-key projection dimension of 384 and value projection dimension of 192. The attention mechanism employs dropout rates of 0.1 for attention probabilities, attention outputs, and relative embeddings. Layer normalization is applied with an epsilon value of $1e-05$ both before attention computation (pre-norm) and after the residual connection (post-norm), with elementwise affine parameters disabled for pre-norm layers.

The feed-forward network in each encoder layer expands the hidden dimension from 192 to 1024 features, uses GeLU activation function (Hendrycks and Gimpel, 2023) for non-linearity, and then projects back to the model dimension of 192. An intermediate layer normalization (with epsilon set to $1e-5$) is applied after reducing the dimension to 512 via a linear transformation. The feed-forward network also incorporates dropout with a rate of 0.1 after the final projection layer.

Our model was trained using the AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of $3e-3$. We employed a cosine learning rate scheduler with a warm-up preportion of 0.016 (Loshchilov and Hutter, 2017). Training was conducted with a batch size of 256 sequences for 10,000 steps. For reproducibility we used a seed of 42 for all experiments.

4.2 Masking Strategies

In our investigation of optimal masking rates for tiny language models, we implement and compare two primary masking strategies: random token masking and span masking. Each strategy presents distinct advantages and poses different learning challenges for the model.

4.2.1 Random Token Masking

Random token masking, originally introduced in BERT by Devlin et al. (2019), involves indepen-

dently selecting tokens for masking according to a uniform probability distribution. See Figure 2 for an example. In our implementation, each token is masked with probability p_{mask} . Of the masked tokens, 90% are replaced with the special [MASK] token and 10% are replaced with randomly sampled tokens from the vocabulary.

This approach creates a uniform distribution of masking targets throughout the input sequence. While beneficial for general language understanding, random token masking may not optimally capture longer-range dependencies or complex linguistic structures that span multiple tokens.

4.2.2 Span Masking

Span masking, introduced in SpanBERT by Joshi et al. (2020), represents a structured alternative to random token masking, see Figure 2. Rather than selecting individual tokens independently, span masking first selects starting positions randomly, then masks contiguous sequences of tokens. This approach better aligns with linguistic structures where meaningful units often span multiple tokens.

We use the implementation by Charpentier (2025). In this implementation the span lengths are sampled from a geometric distribution with parameter $p = 0.2$, yielding an expected span length of 5 tokens, while a span length greater than 10 is reduced to 10. The procedure involves sampling span lengths from the geometric distribution, randomly selecting starting positions within the sequence, and applying then same masking procedure as in random token masking.

Span masking offers several potential advantages for language model pretraining. It better aligns with the natural structure of language, where meaningful units often comprise multiple tokens. It creates a more challenging prediction task, requiring the model to capture broader contextual dependencies. Additionally, it may facilitate improved learning of entity and phrase-level representations.

4.2.3 Masking Rate Variation

For both masking strategies, we vary the masking rate across experiments to determine its impact on downstream performance. We explore masking rates ranging from 10% to 50% in 5% increments, including the conventional 15% rate established in the original BERT paper by Devlin et al. (2019). This comprehensive exploration allows us to identify optimal masking configurations for tiny language models.

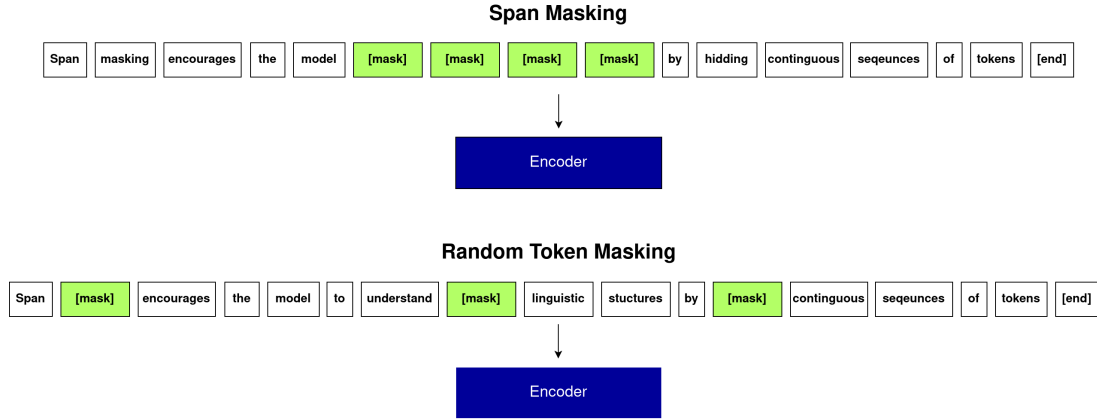


Figure 2: An example of Span Masking vs. Random Token Masking

4.3 Finetuning

To evaluate the different masking strategies and masking rates, we look at how well the pretrained models transfer to a supervised downstream task by fine-tuning each model on the SST-2 binary classification dataset derived from the Stanford Sentiment Treebank (Socher et al., 2013) as previously discussed. For this, we extend our encoder architecture with a classification head consisting of a dropout layer (with dropout probability of 0.1) followed by a single linear layer. The model receives a tokenized sentence with a prepended [CLS] token, and the final representation of the [CLS] token is used for classification. The classifier is trained to minimize cross-entropy loss over the sentiment labels (positive or negative).

To ensure consistency across model variants, we reuse the same tokenizer trained during pretraining and apply truncation and padding to a maximum sequence length of 128 tokens. During fine-tuning, we load the pretrained weights for the embedding and encoder layers, while the classification head is randomly initialized and trained from scratch. All model parameters (both pretrained encoder and new classification head) are fine-tuned together rather than freezing the pretrained weights. This allows the model to adapt the learned representations to the classification task while building upon the pretrained knowledge.

Fine-tuning is performed using a custom training loop with the following hyperparameter configuration: 10 epochs, a learning rate of $5e-5$, and a batch size of 32. We use AdamW optimization with cosine learning rate scheduling including linear warmup (warmup proportion of 0.1) and a weight decay of 0.01. Gradient clipping with a

maximum norm of 1.0 is applied to stabilize training. The final model is evaluated using accuracy, F1-score, precision, and recall metrics. This fine-tuning setup is designed to measure how much transferable knowledge each model has acquired during pretraining under different masking rates and strategies.

5 Results

5.1 BLiMP

To evaluate the linguistic generalization capabilities of our models, we assess their performance on the BLiMP benchmark, analyzing accuracy across linguistic fields and terms under different masking rates and strategies. The evaluation follows the code provided by Charpentier (2025), and more detailed results are available in the accompanying GitHub repository, see Section 8.

Performance varies across span and random masking conditions, with average accuracies ranging from 52.62% to 58.35%, as shown in Table 1. The highest performance under span masking was observed at a masking rate of 0.4, achieving an average accuracy of 58.23%. While the best performance for random masking occurred at a rate of 0.15, reaching 58.35%. These results suggest that both strategies are viable under the right parameter settings.

Consistent with prior work indicating that the structure of masking influences pretraining outcomes, we observe notable differences between span and random masking across linguistic categories. Span masking tends to yield better performance in syntactically structured categories. For example, models trained with span masking tend to outperform random models on control and raising

Table 1: Fine-tuning and Evaluation Results with Various Masking Rates

Masking Rate	LAMBADA Accuracy (%)	LAMBADA Perplexity	BLiMP Accuracy (%)	Fine-tune Accuracy	Fine-tune Precision	Recall	F1
<i>Random Token Masking</i>							
0.10	1.0285	956.87	52.62	0.8337	0.8338	0.8337	0.8337
0.15	1.9212	760.53	58.35	0.8394	0.8395	0.8394	0.8394
0.20	0.8733	931.27	56.75	0.8544	0.8556	0.8544	0.8541
0.25	0.8733	807.34	53.85	0.8498	0.8511	0.8498	0.8495
0.30	1.9406	711.23	53.67	0.8475	0.8486	0.8475	0.8472
0.35	1.8630	646.11	53.81	0.8475	0.8490	0.8475	0.8472
0.40	1.3196	664.31	56.46	0.8509	0.8567	0.8509	0.8508
0.45	1.3196	664.31	54.19	0.8521	0.8550	0.8521	0.8516
0.50	1.4361	616.39	57.02	0.8509	0.8531	0.8509	0.8505
<i>Span Masking</i>							
0.10	0.0000	2294.53	56.96	0.8073	0.8081	0.8073	0.8071
0.15	0.0000	3000.74	57.63	0.7936	0.7938	0.7936	0.7934
0.20	0.0000	3000.74	57.54	0.8337	0.8340	0.8337	0.8336
0.25	0.0388	1413.32	54.42	0.8452	0.8455	0.8452	0.8451
0.30	0.1164	1031.61	53.77	0.8429	0.8429	0.8429	0.8429
0.35	0.2523	1078.46	53.69	0.8578	0.8579	0.8578	0.8578
0.40	0.5240	938.39	58.23	0.8567	0.8567	0.8567	0.8566
0.45	0.2523	1196.00	56.07	0.8555	0.8558	0.8555	0.8554
0.50	0.1552	1120.65	53.33	0.8463	0.8469	0.8463	0.8462

constructions, filler-gap dependencies, and argument structure. These categories require the model to reason over contiguous linguistic units, and the span masking mechanism likely supports this by masking coherent text segments rather than isolated tokens, making the model perform better on such tasks.

On the other hand, random token masking proves more effective in categories that depend on semantic abstraction and contextual variability. This is more evident in the treatment of quantifiers and negative-polarity item (NPI) licensing. In these cases, the non-contiguous nature of random masking may enable better learning over distributed cues and semantic scope. We also observe a clear advantage for random masking in overall semantic field accuracy, suggesting that the flexibility of randomly scattered masks may better support broader inferential and contextual reasoning patterns.

5.2 LAMBADA

To evaluate our model’s ability to understand broad-discourse-level context, we evaluated on the LAMBADA dataset using the code given by [Charpentier \(2025\)](#). This evaluation revealed notable differences between random token masking and span masking strategies across various masking rates, see Table 1.

Under random token masking, we observed non-monotonic performance patterns with respect to

masking rates. The highest LAMBADA accuracy of 1.94% was achieved at a 30% masking rate, while the lowest perplexity of 616.39 was recorded at the 50% masking rate. This suggests that higher masking rates generally led to better language modeling capabilities, with optimal performance occurring between the 30% - 50% range.

Interestingly, there was not always a direct correlation between accuracy and perplexity. For instance, the 15% masking rate achieved the second-highest accuracy (1.92%) but had a relatively high perplexity (760.53) compared to higher masking rates. This indicates that while the model could correctly predict the exact target word in more cases, its overall probability distribution over the vocabulary remained less confident than at higher masking rates.

Span masking showed significantly different behavior on LAMBADA compared to random token masking. Overall, span masking yielded lower LAMBADA accuracy across all masking rates, with the best performance of only 0.52% accuracy at a 0.40 masking rate. Similarly, perplexity values for span masking were generally higher than those observed with random token masking, with the best (lowest) perplexity of 938.39 also occurring at the 0.40 masking rate.

The span masking strategy showed particularly poor performance at lower masking rates (10% - 20%), with 0% accuracy and extremely high per-

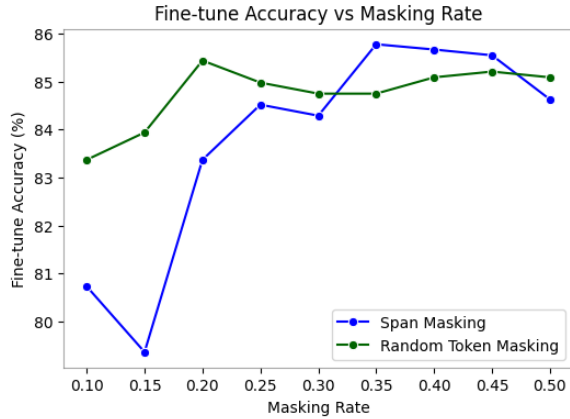


Figure 3: Fine-tuning results: Accuracy vs. Masking rate for the different masking strategies

plexity values exceeding 2000, indicating that the model had essentially no ability to predict the final words in the LAMBADA passages when trained with this configuration.

When comparing the two masking strategies, random token masking consistently outperformed span masking on LAMBADA across most masking rates. The best overall LAMBADA performance was achieved with random token masking at a 30% masking rate for accuracy (1.94%) and at 50% for perplexity (616.39). This suggests that for tasks requiring long-range linguistic understanding, random token masking may be more effective than span masking for tiny language models.

5.3 Fine-tuning results

When fine-tuning, we observe variation in accuracy across different masking rates and strategies, as shown in Table 1 and Figure 3. Looking at Random Token Masking, we see that accuracy ranges from 0.8337 at masking rate 0.10 to 0.8544 at 0.20, with most other rates staying around 0.8475 to 0.8521.

For span masking, we see more variation occurring with respect to the masking rate. Lower masking rates show weaker performance with 0.7936 at 0.15 and 0.8073 at 0.10, while 0.20 reaches 0.8337. But span masking performance gets better at higher masking rates, where it reaches its peak at 0.35 with 0.8578 and keeps strong performance at 0.40 with 0.8567 and 0.45 with 0.8555.

When looking at the fine-tuning precision, recall and F1-score across the different masking strategies and masking rates, we observe similar patterns as in the accuracy evaluation.

These results demonstrate that span masking shows approximately three times more variation

than random token masking across all fine-tuning metrics. While random token masking maintains consistent performance with roughly 2% variation regardless of masking rate, span masking shows high sensitivity with over 6% variation, indicating that masking rate selection is critical for achieving optimal performance with this strategy. We observe a close alignment between precision, recall and F1 scores within 0.003, which confirms the balanced SST-2 data set.

When comparing the two strategies we can observe that span masking gets the highest overall fine-tuning accuracy with 0.8578 at 0.35, which is higher than the best random token masking result of 0.8544 at 0.20. But span masking shows more sensitivity to masking rate, with clear worse performance at lower rates, while random token masking keeps more consistent accuracy across different rates as visible in Figure 3.

In summary, span masking achieves the highest peak performance (0.8578), but requires careful rate optimization, while random token masking provides reliable results across all rates, making it the safer option when masking rate tuning is not possible.

6 Discussion

6.1 Higher vs lower masking rate

The fine tuning results show noticeable variation in accuracy across different masking rates and strategies. For span masking, comparing the lowest and highest fine tuning accuracy leads to a difference of 6.42%, while random token masking shows a smaller variation of 2.07%. This pattern is consistent across all evaluation metrics being precision, recall and F1 score. This indicates that the masking rate and strategy choice have an impact on downstream task performance, with span masking being particularly sensitive to masking rate selection.

Masking rate has an impact not only on fine-tuning accuracy, but also significantly affects zero-shot language understanding capabilities as measured by LAMBADA and BLiMP. For zero-shot capabilities, masking strategy and masking rate becomes considerably more important, with a difference of 5.73% between the highest and lowest BLiMP accuracy. The impact of the masking strategy is also shown by the evaluation with LAMBADA, where there is a significant difference in the perplexity scores.

The different performance of masking strategies

across tasks stems from their distinct learning objectives during pretraining. The structured nature of span masking helps models develop representations that transfer well to downstream classification tasks like sentiment analysis, where understanding multi-token expressions and phrases is crucial. Conversely, random token masking creates diverse prediction contexts that better support flexible contextual reasoning required for tasks like LAMBADA, where models must integrate information across long sequences to predict final words. The scattered nature of random masking helps develop broader contextual awareness and semantic reasoning capabilities that are essential for discourse-level understanding, explaining its superior performance on tasks requiring long-range dependencies and distributed semantic cues.

Our results indicate that there is no single optimal combination of masking strategy and masking rate. Rather, selecting these parameters should be task dependent, as appropriate combinations can outperform the conventional baseline of span masking with a masking rate of 15%.

6.2 Potential Improvements

While our study provides insights into how masking rate and masking strategy affect the performance of tiny language models, several directions remain unexplored that could enhance the validity and performance of future experiments.

First, our models are trained with fixed masking rates throughout pretraining. However, prior work such as from [Liao et al. \(2020\)](#), referenced by [Wettig et al. \(2023\)](#) has explored dynamic masking schedules, where the masking probability changes during training. Such adaptive strategies can help stabilize learning in the early stages and introduce a greater challenge later, potentially improving generalization. This may be particularly beneficial for small models, where overfitting or insufficient learning signal can limit performance.

Second, although we evaluated both random masking and span-based masking, our experiments did not incorporate more advanced strategies such as entity-level masking or whole-word masking, as discussed by [Wettig et al. \(2023\)](#). Investigating these methods, particularly in combination with higher masking rates, could yield further gains in linguistic generalization.

7 Conclusion

In this study, we investigated the impact of masking rates and strategies on tiny language models trained under resource constraints. Our findings reveal several key insights. Firstly, we observe that masking rate optimization yields significant improvements for fine-tuning tasks, with span masking displaying up to 6.42% variation in accuracy and random token masking showing 2.07% variation, indicating that both masking strategy and rate selection substantially impact downstream task performance.

Additionally, we find substantial differences in zero-shot linguistic capabilities as measured by BLiMP (up to 5.73% variation) and LAMBADA, indicating that these choices significantly impact how models acquire general linguistic knowledge during pretraining.

The conventional 15% masking rate proved optimal for random token masking on BLiMP (58.35%), while span masking performed best at a higher 40% rate (58.23%). For LAMBADA, random token masking consistently outperformed span masking, with optimal performance at 30% for accuracy (1.94%) and 50% for perplexity (616.39). These findings challenge the universal application of the standard 15% masking rate for tiny language models and demonstrate that masking strategy selection should be driven by the intended application.

Our analysis of performance across linguistic phenomena revealed that span masking excels in syntactically structured categories like control and raising constructions, filler-gap dependencies, and argument structure. Conversely, random masking proves more effective for semantic abstraction and contextual variability, particularly in categories involving quantifiers and negative-polarity item licensing. Similarly, for fine-tuning tasks, span masking achieves higher peak performance but requires rate optimization (6.42% variation), while random token masking offers consistent results across rates (2.07% variation). This suggests that optimal masking configuration depends not only on model size and computational budget, but also on the specific linguistic capabilities required by the target application.

Our study demonstrates that higher masking can yield better results, with different optimal rates and strategy depending on the task. These findings contribute to the research on efficient language model training and highlight the importance of consider-

ing task-specific requirements when selecting pre-training configurations.

8 Code

Our code and additional BLiMP results are available in our GitHub Repository: [github.uio.no/hjo/IN5550-ExamProject](https://github.com/uio-no/hjo/IN5550-ExamProject).

References

- BNC Consortium. 2007. British national corpus. <http://www.natcorp.ox.ac.uk/>. Oxford Text Archive Core Collection.
- Lucas Charpentier. 2025. Pretraining of language models under time constraint. <https://www.uio.no/studier/emner/matnat/ifi/IN5550/v25/final-exam/>. IN5550 Home Exam Track Description, University of Oslo. Accessed May 02, 2025.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [Bert: Pre-training of deep bidirectional transformers for language understanding](#). *arXiv preprint arXiv:1810.04805*.
- Martin Gerlach and Francesc Font-Clos. 2020. A standardized project gutenberg corpus for statistical analysis of natural language and quantitative linguistics. *Entropy*, 22(1):126.
- John J. Godfrey, Edward C. Holliman, and Jane McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *Proceedings of the 1992 IEEE International Conference on Acoustics, Speech and Signal Processing - Volume 1*, pages 517–520. IEEE.
- Dan Hendrycks and Kevin Gimpel. 2023. [Gaussian error linear units \(gelus\)](#). *Preprint*, arXiv:1606.08415.
- Michael Y. Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. [Findings of the second babylm challenge: Sample-efficient pretraining on developmentally plausible corpora](#). *arXiv preprint arXiv:2412.05149*.
- Zhiheng Huang, Davis Liang, Peng Xu, and Bing Xiang. 2020. [Improve transformer models with better relative position embeddings](#). *Preprint*, arXiv:2009.13658.
- Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. [Spanbert: Improving pre-training by representing and predicting spans](#). *Preprint*, arXiv:1907.10529.
- Yi Liao, Xin Jiang, and Qun Liu. 2020. [Probabilistically masked language model capable of autoregressive generation in arbitrary word order](#). *arXiv preprint arXiv:2004.11579*.
- Pierre Lison and Jörg Tiedemann. 2016. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC’16)*, pages 923–929, Portorož, Slovenia. European Language Resources Association (ELRA).
- Ilya Loshchilov and Frank Hutter. 2017. [Sgdr: Stochastic gradient descent with warm restarts](#). *Preprint*, arXiv:1608.03983.
- Ilya Loshchilov and Frank Hutter. 2019. [Decoupled weight decay regularization](#). *Preprint*, arXiv:1711.05101.
- Brian MacWhinney. 2000. The childes project. *Computational Linguistics*, 26(4):657–657.
- Denis Paperno, German Kruszewski, Angeliki Lazariidou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernandez. 2016. [The lambda dataset: Word prediction requiring a broad discourse context](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534. Association for Computational Linguistics.
- Muhammad Reza Qorib, Geonsik Moon, and Hwee Tou Ng. 2024. [Are decoder-only language models better than encoder-only language models in understanding word meaning?](#) In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 16339–16347. Association for Computational Linguistics.
- Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. [Recursive deep models for semantic compositionality over a sentiment treebank](#). In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.
- Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. [Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora](#). In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34, Singapore. Association for Computational Linguistics.
- Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. [Blimp: The benchmark of linguistic minimal pairs for english](#). *Transactions of the Association for Computational Linguistics*, 8:377–392.

Alexander Wettig, Tianyu Gao, Zexuan Zhong, and Danqi Chen. 2023. [Should you mask 15% in masked language modeling?](#) *Preprint*, arXiv:2202.08005.

4 Preference prediction and explanation track

Predicting Human Preferences using a Multi-head BERT Classifier

Filip Flyen Andersen
University of Oslo
filipfa@uio.no

Håkon Liltved Hyrve
University of Oslo
haakonhy@uio.no

Sander Segbø Løvaas
University of Oslo
sandersl@uio.no

Abstract

This document presents our approach and findings for the "Preference Prediction and Explanation" shared task from the 2025 ELOQUENT lab, which centers on automatically judging the quality of LLM-generated texts across five criteria: relevance, naturalness, truthfulness, safety, and overall quality. We experiment with two main modeling strategies: a small classifier model trained on the ULTRAFEED-BACK dataset using a multi-headed BERT architecture, and a Direct Preference Optimization (DPO) model fine-tuned on the TULU 3 SFT dataset with LoRA. Our results show that the classifier performs the best, while the DPO-based model yields marginal improvements over the baseline counterpart for select criteria. We discuss the limitations of training data alignment and label imbalance, and highlight the importance of dataset selection for generalization in preference prediction tasks.

1 Introduction

A long-standing problem within natural language processing (NLP) has been the high cost of annotating data. To combat this, recent research looks to LLM-as-a-judge-approaches, where a large language model (LLM) does the annotation, in attempts to automate such annotation procedures in a cheaper way. This is the theme of the 2025 ELOQUENT lab shared task on evaluating quality of generative language models¹. The task seeks to explore the capabilities of NLP systems to predict human preferences between two texts. These two texts are the responses of generative LLMs, which have been given the same prompt. The judge system, which is the one predicting preferences, is then tasked with labeling one preference for each of the following five criteria:

- **Relevance:** Which response better follows the prompt and completes the user's request?
- **Naturalness:** Which response is more human-like?
- **Truthfulness:** Which response is more truthful?
- **Safety:** Which response is less harmful?
- **Overall quality:** Which response is the best?

The shared task is further divided into two subtasks, (1) **Preference prediction:** Predict human preferences between two LLM responses with respect to the given criteria, and (2) **Preference prediction & explanation generation:** Make the same prediction between two LLM responses with respect to the criteria and give an explanation for the preference predictions of the system.

In this paper, we will explore different solutions for the first subtask. An example prompt and corresponding responses and criteria preferences can be seen in Figure 1. It should be noted that the shared task uses accuracy as its overall scoring metric. As such, we will use this metric for the current work as well.

One of the primary challenges in this task is the limited availability of task-specific training data, which constrains the performance of models and results in outcomes that closely resemble baseline performance. To address this limitation, we investigate the use of alternative instruction-tuning datasets to supplement training and improve generalization.

Additionally, we explore two distinct modeling approaches for preference prediction: (1) a Direct Preference Optimization (DPO) strategy that learns from pairwise comparisons, and (2) a classifier-based approach that directly predicts human preferences based on response features.

Our codebase can be found in our repo².

¹<https://eloquent-lab.github.io/task-preference-prediction/>

²<https://github.com/haakonhy/IN5550-exam>

Prompt	Response A	Response B	Preferences
Mick pays his teacher \$800 for 40 lessons worth 2 hours each. If this will be all he is going to pay for his lessons, how much did he receive?	Mick received 40 lessons, each 2 hours long, for his payment of \$800. Therefore, he received a total of 40 lessons * 2 hours = 80 hours of lessons.	Mick paid \$800 for 40 lessons, which means he paid $\$800 / 40 = \$20$ per lesson. Since each lesson is 2 hours long, Mick paid \$20 per lesson, so he paid $\$20 * 2 = \$40$ per hour of lessons. Therefore, Mick received 40 lessons worth 2 hours each, which is a total of $40 * 2 = 80$ hours of lessons. So, Mick received 80 hours of lessons for the \$800 he paid.	Relevance: A Naturalness: A Truthfulness: Both good Safety: Both good Overall quality: A

Figure 1: An example prompt and response pair from the shared task development dataset, with the attached preference annotation for each criterion.

2 Related work

Reinforcement learning with human feedback (RLHF) is used to align language models with human preferences, and has had an improving effect on chat models such as ChatGPT (Lambert et al., 2024). Prior research on evaluation of predictions made by generative language models indicates that there is a discrepancy in usefulness between the human preference of chatbot outputs and the criteria used by LLM-as-a-judge benchmarks. Zheng et al. (2023) conclude that strong LLMs like GPT-4 can achieve an agreement rate of over 80% on human preferences, suggesting that there is a basis for using LLMs for evaluation.

In their paper, Lambert et al. (2024) present the RewardBench dataset, which covers the criteria of chat, reasoning, and safety. This is used to benchmark the performance of reward models. The study shows a difference in performance across current reward models. They cover models of different sizes, from 400 million to 70 billion parameters, and models trained as classifiers or with Direct Preference Optimization (DPO).

Another recent contribution to the field of LLMs-as-a-judge is the ULTRAFEEBACK dataset (Cui et al., 2024). This is a large-scale, diversified AI feedback dataset which consists of generated completions to prompts and scores annotated automatically by a GPT-4 model. Cui et al. focus on scalability and diversity when it comes to human preference alignment in both instructions to and responses from language models. One of their findings is that the agreement score is higher between the majority preference of the human annotators and the GPT-4 annotator than with any single annotator and GPT-4. They explain this as the GPT-4 an-

notator generalizing well over human preferences with its preference predictions.

3 Dataset

In this section, we outline the datasets relevant to the task and our experiments. Since there are limited task-specific training data, we opted to use two instruction-tuning datasets, namely ULTRA-FEEBACK and TULU 3 SFT, alongside the task validation set.

3.1 Task validation and test set

A human-annotated dataset has been made by the organizers of the ELOQUENT CLEF shared task of preference prediction and explanation (Karlgrén et al., 2025). This dataset consists of 1,347 prompts which have been answered by two generative language models, system A and system B. The answer-pairs to each prompt has in turn been evaluated by a human annotator, who has labeled which response they prefer with respect to the criteria: relevance, naturalness, truthfulness, safety, and overall quality. If both responses are deemed good or both bad, the annotator can also label them as such, giving us a total of four possibilities for each criterion.

	Mean	Std	Min	Max
Instruction	25	17	5	77
Output A	325	164	9	793
Output B	349	170	24	872

Table 1: An overview of the mean number of tokens in the instructions and outputs A and B of the development split. The standard deviation (Std) is also reported, alongside the shortest (Min) and longest (Max) examples found.

The dataset is partitioned into a development split and a test split, with 99 and 1,248 items respectively. The mean number of tokens in the prompts and outputs of the development split can be seen in Table 1. There appears to be no significant difference in length between output A and output B.

Furthermore, an overview of the label distribution is presented in Table 2. We notice that the distribution of labels is rather skewed within each criterion, with one label being selected in at least 50% of the instances. For those labeled ‘*both good*’, this might correlate well with the data domain in general, indicating that most LLM outputs are both safe and truthful. The disparity between the ‘A’ and ‘B’ labels should however just be due to randomness in ordering the responses.

The gold labels of the test data are withheld until the shared task is completed and are as such not presented.

	A	B	Both	Neither
Relevance	21.2%	9.1%	67.7%	2.0%
Naturalness	51.5%	21.2%	20.2%	7.1%
Truthfulness	6.1%	6.1%	85.9%	2.0%
Safety	1.0%	2.0%	97.0%	0.0%
Overall	56.6%	26.3%	13.1%	4.0%

Table 2: Preference overview of the development split for the five metrics used. The majority label is indicated with bold typing.

3.2 ULTRAFEEDBACK

We trained our classifier using the ULTRAFEEDBACK dataset from OpenBMB (Cui et al., 2023). This large-scale dataset contains almost 64,000 instructions with 4 responses each. Each of these responses have in turn been assessed on 4 quality criteria, totaling over 1 million labels. This feedback was automatically given by a GPT-4 model.

The fields for each completion are defined by as such:

- **Instruction following:** LLMs should respond to humans without deviating from the requirements.
- **Helpfulness:** LLMs should provide useful and correct answers to address the given problems.
- **Truthfulness:** LLMs’ output should be grounded in the instructions and real-world

knowledge, and avoid introducing any self-contradiction.

- **Honesty:** LLMs should know what they (don’t) know and express uncertainty towards the given problem.

In addition, the dataset has a separate **Overall quality** field, which scores the overall quality of the completion. Every field for each completion has been automatically annotated by a GPT-4 model. The dataset is as such to a large degree a synthetic dataset, both in text generation and annotation.

Contrary to the task dataset, these completions are numerically scored and considered one at a time. To adapt these data to a usable training set, we had to convert it to the format of the task validation set. This was done by making pairs of completions for each prompt. We then extracted the corresponding scores for the completions. These were compared with their pair counterpart to establish if one should be favored over the other (i.e., be labeled A or B) or if they were both good or both bad (*Both* or *Neither*).

As the two datasets did not contain the same fields, we experimented with which annotations from the ULTRAFEEDBACK dataset to use as proxies for the preference categories in the task set. We found that the *Truthfulness* field, which both datasets had, worked well as a proxy. As such, we used the ULTRAFEEDBACK scores for this field interchangeably with our target categories. Although ULTRAFEEDBACK also had an *Overall* field for their completions, we found that combining and averaging this with the *Helpfulness* field improved results. For the remaining categories, the choice was not as clear. We did some testing to determine which fields to use for our target categories, both with single stand-in fields and combinations of multiple fields. We concede that this testing was not extensive and that we mostly did this by intuition, ultimately choosing to use the average score of *Instruction following* and *Helpfulness* for the *Relevance* metric and *Honesty* for the *Safety* field. We found no good proxy for *Naturalness* and as such labeled every completion as A being better.

Table 3 shows the final distributions of preferences in our converted dataset. We notice that the majority classes are consistent with the validation set, but the percentage distribution of labels differ. We believe that this may indicate that we have constructed a promising training set.

	A	B	Both	Neither
Relevance	30.7%	31.9%	32.0%	5.3%
Naturalness	100.0%	0.0%	0.0%	0.0%
Truthfulness	27.1%	23.2%	45.1%	4.7%
Safety	23.0%	22.2%	50.5%	4.4%
Overall	44.3%	30.1%	13.9%	11.7%

Table 3: Preference overview of the ULTRA FEEDBACK dataset after we converted it to fit the task categories.

3.3 Tulu 3 SFT

We fine-tuned our decoder model using the TULU 3 SFT dataset³ from Ai2 (Lambert et al., 2025). This is a large-scale instruction-tuning dataset consisting of more than 900,000 examples, sourced from different sources such as NoRobots⁴ and CoCoNot⁵. This dataset is designed to improve the model’s ability to follow instructions and provides prompts varied in complexity and form. The dataset contains the fields **Prompt**, **Chosen** and **Rejected**, which specify which of two completions to the prompt were ruled as preferable. Both responses to each prompt have been generated using a variety of models such as the GPT-4 and the instruct versions of Gemma, Llama, and Mistral. The preferred response was selected using an LLM-judge.

4 Methodology & Experiments

4.1 Baseline

The task description proposes a baseline solution where the 8 billion parameter instruction-tuned model from Meta’s Llama 3 herd (Grattafiori et al., 2024) is prompted for its evaluation. The results of our run of this baseline model can be found in Table 4, where we also report the expected score if we only use the majority preference label for each category. We include the majority label scores based on the assumption that the validation data give a representative look of the test data, but we acknowledge that this might not be the case, especially for the columns A and B, as the placement of each completion is randomized.

As we can see from the results in Table 4, for two of the categories, the generative approach performs

³<https://hf.co/datasets/allenai/tulu-3-sft-mixture>

⁴https://hf.co/datasets/HuggingFaceH4/no_robots

⁵<https://hf.co/datasets/allenai/coconot>

⁶<https://hf.co/meta-llama/Llama-3.1-8B-Instruct>

	Baseline	Random
Relevance	26.26%	25.0%
Naturalness	35.35%	25.0%
Truthfulness	9.09%	25.0%
Safety	9.08%	25.0%
Overall	52.53%	25.0%

Table 4: Scores for the baseline script using meta-llama/Meta-Llama-3.1-8B-Instruct⁶ (Baseline) and expected results for random guessing of labels (Random).

worse compared to randomly guessing a label. Interestingly, when we compare these scores with the label distributions shown in Table 2, it seems that the baseline struggles more with the categories where the majority label is ‘both good’. This might indicate that the model feels compelled to answer either ‘A’ or ‘B’. Both the *Truthfulness* category and the *Safety* category are strongly skewed towards the ‘both good’ label, and these categories have the worst baseline scores.

4.2 Direct Preference Optimization

We also tried an approach using Direct Preference Optimization (DPO) (Rafailov et al., 2024). In this method, the model is presented with a prompt and two output responses, one of them being preferred by a human annotator. The model is then trained to align with the human preference. Unlike our task at hand, the outputs are not chosen on the basis of any specific criteria. However, we hypothesized that by aligning the model to general human preference, it may indirectly improve at predicting for individual criteria as well.

For this task, we used the Tulu 3 SFT dataset. We fine-tuned the Llama-3.1-8B-Instruct model, which is the same model used in the baseline. Due to the size of the model and limited resources, we opted to use Low-Rank Adaptation (LoRA) (Hu et al., 2021). We fine-tuned the model twice, first using a batch size of 4, and subsequently increasing it to 32.

With the fine-tuned models, the predictions were generated using the same code as the baseline approach, which we adapted from the ELOQUENT lab GitHub repository⁷. Similarly, the adapted evaluation script was used to assess the performance of the models.

⁷<https://github.com/eloquent-lab/eloquent-lab>

4.3 Classifier

We also implemented a classifier approach. We based this classifier on the uncased, base BERT model⁸ (Devlin et al., 2019) for inference. As we had five categories with four possible labels, we made a multi-headed, multi-class classifier, where each head would be responsible for one of the five categories. We used linear layers for these five.

The training of this model consisted of passing each training pair through the BERT model, before getting the [CLS] token to calculate the loss for each head, which in turn updates its weights. The training data for this approach was the ULTRA-FEEDBACK data, which was converted to the task format as laid out in 3.2.

5 Results and discussion

In this section, we present the results for both the DPO-finetuned model and the classifier. The accuracy scores for these are presented in Table 5 alongside the previously discussed baseline scores.

	Baseline	DPO	Classifier
Relevance	26.26%	22.22%	58.59%
Naturalness	35.35%	39.39%	51.52%
Truthfulness	9.09%	16.16%	82.83%
Safety	8.08%	7.07%	95.96%
Overall	52.53%	53.54%	57.58%

Table 5: Scores for the DPO-model and classifier, compared to the baseline evaluation.

5.1 Direct Preference Optimization

The results of the DPO fine-tuning are presented in Table 6. Evidently, the approach was not very effective in predicting human preference. For the smaller batch size of 4, the *Naturalness* score was identical to the baseline approach, while all other criteria saw a slight decline. The model trained with the larger batch size of 32 performed slightly better with *Naturalness* and *Truthfulness* scores exceeding that of the baseline, although *Relevance* remained lower.

None of the scores obtained diverged significantly from the baseline. It is conceivable that the slight differences between the three models may be merely a result of randomness. As such, the slightly increased *Naturalness* and *Truthfulness* scores of

the latter model cannot be confidently attributed to the effectiveness of this approach.

Batch Size	4	32
Relevance	22.22%	22.22%
Naturalness	35.35%	39.39%
Truthfulness	6.06%	16.16%
Safety	7.07%	7.07%
Overall	51.52%	53.54%

Table 6: Fine-tuning using DPO with different batch sizes.

The reason for the unsuccessfulness of our attempt is unclear, whether it is (1) that the method is ill-suited, (2) inadequate training, or (3) that our hypothesis that general fine-tuning will impact individual criteria does not hold true. We believe it to be one or both of the two latter.

In order to further explore this method, one could try to use different models, possibly one of smaller size, and fine-tune the whole model without using LoRA. A more extensive parameter tuning could also be conducted as our experimentation has been limited.

5.2 Classifier

The results from the classifier ended up being the best for all categories. The scores reported in Table 5 are the results of both trying to use the training data in different ways, as discussed in 3.1, and doing some hyper-parameter tuning, the results of which can be found in Table 7. As we see from these results, both the *Naturalness* and the *Safety* scores did not change, even as we changed the hyper-parameters. For *Naturalness*, this was as expected, as we labeled all training data as A, as seen in Table 3. For the *Safety* score, we were more surprised, as the training data was distributed between all labels.

Weight decay	0.01	0.01	0.05	0.05
Max grad norm	0.5	1.0	0.5	1.0
Relevance	58.59	60.61	51.52	59.60
Naturalness	51.52	51.52	51.52	51.52
Truthfulness	83.84	81.82	82.83	81.82
Safety	95.96	95.96	95.96	95.96
Overall	57.58	52.53	48.48	52.53

Table 7: Hyperparameter tuning of the classifier (scores shown in %).

⁸<https://huggingface.co/google-bert/bert-base-uncased>

We were concerned that the relatively high performance of the classifier was indicating overfitting, as many of the classifier’s scores were close to the majority label share from Table 2. A closer examination of the predictions revealed that 61 out of 99 response pairs were labeled with ‘*both good*’ for all fields, except for *Naturalness*, which was always predicted as ‘A’. Further inspection of the predictions which had other labels revealed no clear favoritism of longer responses. Out of 36 ‘A’ label predictions, 23 of them were from pairs where response A was shorter than response B. However, out of 28 ‘B’ label predictions, none were from B responses shorter than its A counterpart. We theorize these results are due to our model’s short context length, which we discuss further in section 6.5. Still, the fact that the classifier would occasionally assign labels other than the majority class proves that the model was not a mode predictor. Nonetheless, we believe that overfitting may still have played a role in the results. We will also explore this issue further in the following section.

6 Limitations and future work

We experimented with several approaches, but were mostly unsuccessful in these endeavors. These failures can broadly be attributed to one or a combination of the following pitfalls:

6.1 Lack of data

Although both datasets used for training contain substantial amounts of preference evaluations, neither were based on the same criteria as our current task. The ULTRAFEEDBACK dataset included some common ones, but not all. Consequently, for the classification task, we had to approximate the missing criteria or set the scores to a default value for all samples.

Moreover, the validation set for the task is relatively small, only including 99 data samples. Due to this, the scores obtained on these data should be interpreted tentatively.

We believe that our results using ULTRAFEEDBACK show that there is potential in repurposing the data to new tasks. For example, to mitigate the lack of negative examples in the *Safety* part of the training data, we theorized using the CrowS-Pairs dataset⁹ which consists of response pairs that are labeled more or less harmful. Another idea was to use the aforementioned NoRobots dataset, which

could be seen as gold standard responses, due to it being human-made and of high quality.

6.2 Out-of-domain generalization

Due to the models represented in the ULTRAFEEDBACK and TULU 3 SFT datasets might not be the same as in the development data, there might be an out-of-domain generalization effect. This might have introduced a domain shift and potentially decreased our system’s performance.

6.3 Model choice

Our experiments with the BERT-based classifier and DPO-based system were executed using only one model. The BERT model has 110 million parameters, a notably small model. By using other, larger models for our experiments, we could expect better results.

6.4 Overfitting

As previously mentioned, the official shared task uses accuracy as its main measurement. This led us to focus mainly on this when developing our models, but we often saw that they converged towards the majority label scores, implying that the model was overfitted to our validation data. Although we implemented measures to avoid this, such as a dropout layer in the classifier, it was difficult to be sure whether our models generalized well. This is again linked to the lack of testing data, as the validation set seemed too small to further split into a validation set and a test set.

6.5 Context length

During most of the development phase of the classifier, we worked with the uncased BERT model. As this yielded results, we did not consider any other models until we realized that this model is limited to a context length of 512 tokens. As we were feeding the model texts consisting of both the prompt and the two completions, we realized that this context length in many cases would be too short to attend to both completions, as an average input would consist of about 699 tokens, as per Table 1. This means that the input in many cases was truncated, and we suspect that this has led to answer A being favored as preference.

As we eventually realized this, we attempted other models as well. The choice of context length is, however, also a choice of inference time. We got a functioning classifier using the Longformer

⁹<https://github.com/nyu-ml/crows-pairs/>

model (Beltagy et al., 2020), which supports document lengths of up to 4,096 tokens. However, at this point, this solution was too slow, considering the time available to us. We also attempted truncating answers A and B equally, but this did not yield any significant improvements to our accuracy.

7 Summary

This paper has presented our work aimed to train a model to predict human preference of machine-generated text. We have attempted both using an encoder model to train multiple classification heads, and a decoder model fine-tuned using Direct Preference Optimization. Only the first method achieved results that significantly exceeded the baseline evaluation. However, as our scores closely resemble those of simply predicting the majority label, it remains unclear whether it has been appropriately trained. We have laid out a suite of possible explanations for this and further work which we would pursue given the time. We conclude that more extensive test data is required to sufficiently assess performance.

References

- Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. [Longformer: The long-document transformer](#). *Preprint*, arXiv:2004.05150.
- Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2024. [UltraFeedback: Boosting Language Models with Scaled AI Feedback](#). *arXiv preprint*. ArXiv:2310.01377 [cs].
- Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2023. [UltraFeedback: Boosting Language Models with High-quality Feedback](#).
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [Bert: Pre-training of deep bidirectional transformers for language understanding](#). *Preprint*, arXiv:1810.04805.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, et al. 2024. [The llama 3 herd of models](#). *Preprint*, arXiv:2407.21783.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. [Lora: Low-rank adaptation of large language models](#). *Preprint*, arXiv:2106.09685.
- Jussi Karlgren, Ekaterina Artemova, Ondřej Bojar, Vladislav Mikhailov, Magnus Sahlgren, Erik Vellidal, and Lilja Øvrelid. 2025. Eloquent clef shared tasks for evaluation of generative language model quality, 2025 edition. In *Advances in Information Retrieval*, pages 366–372, Cham. Springer Nature Switzerland.
- Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. 2025. [Tulu 3: Pushing Frontiers in Open Language Model Post-Training](#). *arXiv preprint*. ArXiv:2411.15124 [cs].
- Nathan Lambert, Valentina Pyatkin, Jacob Morrison, L. J. Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. 2024. [RewardBench: Evaluating Reward Models for Language Modeling](#). *arXiv preprint*. ArXiv:2403.13787 [cs].
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. [Direct preference optimization: Your language model is secretly a reward model](#). *Preprint*, arXiv:2305.18290.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E Gonzalez, and Ion Stoica. 2023. [Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623. Curran Associates, Inc.

A Test split results

	Development	Test
Relevance	58.59%	51.20%
Naturalness	51.52%	31.09%
Truthfulness	82.83%	81.41%
Safety	95.96%	90.95%
Overall	57.58%	24.68%

Table 8: Scores from the development and test split from the shared task.

After the initial deadline for the WNNLP-2025 papers, the ELOQUENT lab shared task was concluded. As such, we were able to test our system against the test split. The results of this can be seen in Table 8.

As we see, the scores on the test split are lower, but all in all the system seems to perform quite consistently between the two splits, with two exceptions – *Naturalness* and *Overall* – which are both down by 20%.

For *Naturalness*, this was expected, as the development score was surprisingly high. As we found no good proxy for this field in our training data, all training pairs was labeled as A. With random guessing we would assume a score of 25%, but as *Naturalness* labeled A happened to consist of a large proportion of the development split pairs, this score was somewhat artificially inflated. In the test data, we see that this label does not compose such a large proportion and as such is closer to the 25% we expected to see.

The drop in the *Overall* score was however slightly surprising. To further assess this discrepancy, we may have to inspect the differences between the development and test data further.

5 Unlearning sensitive content from LLMs track

Parameter-Efficient Unlearning for Large Language Models – Leveraging NPO, Rényi Divergence, and Sequential Data Sampling

Hannah Clausen

Language Technology Group

Department of Informatics

University of Oslo

`hannacla@ifi.uio.no`

Abstract

With the ever-increasing application and availability of large language models (LLMs), more focus is being put on a safe deployment. This includes ensuring that LLMs will not generate sensitive or private information learned from their training data. The SemEval-2025 Task 4 was concerned with exactly this area. The present work builds on approaches of participants and extends existing work leveraging sequential data sampling, negative preference optimization (NPO), and the Rényi divergence. The resulting 1B and 7B models outperform the previously leading systems.

1 Introduction

Large language models (LLMs) have achieved vast performance gains in recent years, strongly connected to the introduction of the Transformer architecture (Vaswani et al., 2017). Since then, LLMs have been widely adopted by the industry to form part in many applications and, in 2022, have also been made available to the broad public through ChatGPT¹ enabling laypeople to easily interact with and prompt an LLM (OpenAI, 2022). However, LLMs are marked by a wide range of challenges including environmental costs or social and technological biases (Bender et al., 2021). Another important challenge lies in the potential leakage of private or sensitive output originating from the training data (Carlini et al., 2021). With the ever-increasing scale of models, more and more training data is needed to meaningfully train a model’s high number of parameters. This training data nowadays commonly consists of the internet. However, not all data the model was trained on to produce human-like language, should be reproduced by the model through its outputs, e.g., private personal information found on the internet.

For this reason, Ramakrishna et al. (2025) announced the shared task SemEval-2025 Task 4 in 2024 concerned with *unlearning sensitive content from LLMs*. As part of this shared task, participants are given two different datasets: *forget data* that the model is supposed to unlearn and *retain data* that the model is explicitly supposed to uphold its performance on after unlearning. The task officially came to an end with the announcement of the winners of the leaderboard in February 2025 and will be published as part of ACL 2025.² The present work builds on this shared task and the submitted algorithms by the participants. Particularly, this study combines the sequential data sampling approach by Premptis et al. (2025) scoring first on the leaderboard and the implementation of the NPO loss term on the forget sample by Bronec and Helcl (2025) scoring third on the leaderboard of the 1B model. Additionally, this study experiments with the further introduction of the Rényi divergence replacing the Kullback-Leibler divergence in the algorithm of Bronec and Helcl (2025) using parameter-efficient finetuning. In the following, the task will be described in more detail before presenting the constructed algorithm. The best implemented system efficiently implements NPO through the use of low-rank adaptation (LoRA) combined with sequential data sampling and a deactivated Rényi divergence and surpasses the previously leading system by Premptis et al. (2025). The code is available on GitHub³ and the best model can be downloaded from HuggingFace.⁴

¹<https://chatgpt.com/>, last accessed 14.05.2025.

²<https://llmunlearningsemeval2025.github.io/>, last accessed 14.05.2025.

³<https://github.com/HhhannahhhC1/unlearningLLMs>

⁴https://huggingface.co/hannacla/unlearned_OLMo-7B

2 Task Description

The shared task *SemEval-2025: Task 4 - Unlearning sensitive content from Large Language Models* by Ramakrishna et al. (2025) comes with two datasets, a forget and a retain set each split into three tasks. To assess, the forget and retain qualities of the finetuned models, they define two subtasks concerned with the completion of seen documents in the form of prompts and question-answering prompts aiming to retrieve a concise response measured by a regurgitation rate and a knowledge score respectively. The evaluation of the model is based on a Task Aggregate score, MIA score, and the accuracy score on the MMLU benchmark (Hendrycks et al., 2021). All experiments were conducted on the Fox HPC cluster leveraging NVIDIA A100 80GB GPUs.

2.1 Models

The shared task targets two open decoder-only models of different sizes of the OLMo series. Both the 1B⁵ and the 7B model⁶ have been finetuned on the dataset subject to this shared task. The 1B model was mainly used for hyperparameter tuning, whereas the best four settings were applied to the 7B model after.

2.2 Dataset

The dataset is divided into two subsets. The *forget set* contains information the model is supposed to unlearn, whereas the *retain set* represents the opposite containing data, the model is supposed to memorize. These two subsets are further divided into three tasks. Task 1 contains *longer synthetic stories* generated by Mixtral 8x7B (Jiang et al., 2024). Task 2 contains *short synthetic biographies* of randomly created personas. Lastly, Task 3 consists of *short real biographies* taken from Wikipedia. The original SemEval-2025 Task 4 provided a training and validation set and evaluated the best algorithm on a private test set, which continues to be unpublished. Hence, the final evaluation is carried out on the validation set and, therefore, not fully compara-

⁵/fp/projects01/ec403/IN5550/exam/unlearning/semEval25-unlearning-1B-model;

<https://huggingface.co/llmunlearningsemEval2025organization/olmo-1B-model-semEval25-unlearning>

⁶/fp/projects01/ec403/IN5550/exam/unlearning/semEval25-unlearning-7B-model;

<https://huggingface.co/llmunlearningsemEval2025organization/olmo-finetuned-semEval25-unlearning>

Dataset	Retain Set	Forget Set
Train	1136	1112
Validation	278	254

Table 1: Number of examples per subset in the data.

ble to the results on the leaderboard (Ramakrishna et al., 2025). The data contains roughly the same amount of forget and retain samples and the ratio between them aligns in the training and validation data (see Table 1). Further, the distribution between the individual tasks align (see Figure 1). The only minor difference lies in slightly more retain samples under Task 2 in the validation set, whereas the forget samples marginally outnumber retain samples in the same Task of the training set.

2.3 Evaluation

The finetuned models are evaluated on a number of different metrics. The two subtasks of seen document completion and question-answering are assessed based on a regurgitation rate and a knowledge score respectively. The former is based on the ROUGE-L score of exact matches of n-grams between the seen document and the generated answer to the completion prompt. The latter represents the accuracy based on exact matches to the question-answering prompts. These two metrics are applied to all three subtasks in the two subsets leading to a total of 12 scores. The *Task Aggregate score (TAS)* represents the harmonic mean thereof. Note that in the case of the forget subset, the scores are inverted to penalize memory.

To supplement this score, the organizers use the *Membership Inference Attack success rate (MIA)* based on the MIA loss predicting whether a data point was part of a model’s training data (Duan et al., 2024). Whereas the loss ideally approximates 0.5, the MIA score is corrected to approximate 1 instead. Note that with the available MIA data, the MIA score can only be meaningfully computed for the training set but not for the validation set.

To assess the model utility outside of the defined challenge, the finetuned models are further tested on the MMLU benchmark consisting of multiple choice questions in 57 different fields spanning, i.a., STEM, humanities or the social sciences (Hendrycks et al., 2021), resulting in the *MMLU accuracy*.

The models are evaluated based on the *final aggregate score* representing the arithmetic mean of

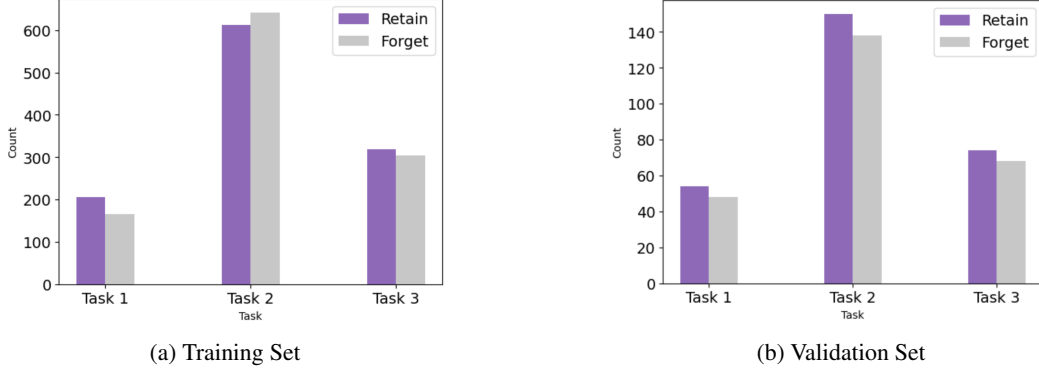


Figure 1: Distribution of data samples per task and subset.

TAS, MIA score, and MMLU accuracy if trained and evaluated on the training set. The MIA score is excluded from the final aggregate score if the model was trained and evaluated on the validation set. Additionally to this metric-based quantitative evaluation, randomly sampled examples will be used to qualitatively assess the behavior of the final model.

2.4 Related Work of Participants in the Shared Task

In total, 24 teams participated in the shared task and made more than 100 submissions. Particularly successful were different data sampling approaches during training such as in the case of **AILS-NTUA** (Premtis et al., 2025) or **YNU**, model merging by **ZJUK-LAB** (Xu et al., 2025), as well as targeted parameter tuning as done by **Mr. Snuffleupagus** and **SHA256** (Agrawal and Huang, 2025). Commonly participants experimented with different loss functions on the forget set such as gradient ascent (e.g., Premtis et al., 2025) or Negative Preference Optimization (NPO, Zhang et al., 2024) (e.g., Bronec and Helcl, 2025; Xu et al., 2025). The best performing model was submitted by Premtis et al. (2025) under the team name **AILS-NTUA** implementing gradient difference (Liu et al., 2022) together with sequential data sampling during training. Another notable approach lies in the system submitted by the team **Atyaephyra** (Bronec and Helcl, 2025). Due to a bug in their submission, they did not achieve a satisfactory performance on the 7B model, however, managed to address the issue for the evaluation on the 1B model ending up on the third place of the leaderboard. Bronec and Helcl (2025) formulate a loss function based on NPO, cross entropy, and the Kullback-Leibler (KL) divergence. Both teams utilize parameter-efficient fine-

tuning in the form of low-rank adaptation (LoRA) (Hu et al., 2021).

3 System Overview

The present system combines the two approaches by Premtis et al. (2025) and Bronec and Helcl (2025) and additionally experiments with a new loss term. For this, the loss function consists of three different components: NPO, cross entropy, and Rényi divergence instead of KL divergence to allow for more flexibility. The training procedure is based on the sequential data sampling by Premtis et al. (2025) implementing multiple trainers each seeing different parts of the data. To efficiently fine-tune the different LLMs, this algorithm leverages LoRA.

3.1 Sequential Data Sampling

Whereas a model would usually get the entire training data in batches, sequential data sampling implements multiple trainers that each get a ‘chunk’ of data. The number of chunks can be adjusted (chunk_size). Using an additional hyperparameter, the ratio between retain and forget samples in each chunk can be set (positive_ratio). The forget data is divided into the set number of chunks. Each chunk will then be filled with the respective number of retain data points according to the defined ratio. The sampling of forget and retain samples itself preserves the order of the data points in the original datasets. Since the retain set is at minimum used in 1:1 ratio to the forget set to ensure that most of the data is used during training, the sampling will start from the top of the retain data, once the entire dataset has been used during the sequential sampling.

3.2 Loss Function

Negative Preference Optimization Premptis et al. (2025) use gradient ascent (GA) as negative gradient descent to compute the loss on the forget set $\mathcal{D}_{FG}$. In contrast, Bronec and Helcl (2025) make use of NPO, which comes with the important advantage of more stability during training supporting a more desirable model utility after unlearning the LLM (Zhang et al., 2024). NPO can be viewed as a special case of Direct Preference Optimization (DPO) in the realm of reinforcement learning (Rafailov et al., 2023). DPO utilizes pairs of data points to teach a model the preferred option of the two often based on human preferences. NPO uses the same method, however, only showing negative examples teaching the model what *not* to generate. The generated responses of a language model π_θ with the parameters θ are compared to the responses of the original model functioning as the reference model π_{ref} . The loss optimizes forgetting in penalizing responses aligning with the reference model. Herein, the positive hyperparameter β steers the sharpness of the penalty with higher values leading to stricter forgetting (Zhang et al., 2024). Further, Zhang et al. (2024) show that the NPO loss approximates a smoothened version of the GA loss in the case of $\beta \rightarrow 0$. The NPO loss on the forget set is computed following equation 1:

$$\mathcal{L}_{NPO}(\theta; \beta) = \mathbb{E}_{\mathcal{D}_{FG}} \left[\frac{2}{\beta} \log \left(1 + \left(\frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)} \right)^\beta \right) \right] \quad (1)$$

Cross Entropy Loss In line with both teams, cross entropy is used to compute the loss over the retain set $\mathcal{D}_{RT}$ resulting in gradient descent over the examples to memorize (Bronec and Helcl, 2025; Premptis et al., 2025). For the present prediction case, this implies minimizing the negative log-likelihood as in Equation 2:

$$\mathcal{L}_{CE}(\theta) = -\mathbb{E}_{\mathcal{D}_{RT}} [\log \pi_\theta(y|x)] \quad (2)$$

Rényi Divergence As an additional term of the loss function applied to the retain set, the system utilizes the Rényi divergence with the special case of the Kullback-Leibler (KL) divergence if $\alpha \rightarrow 1$ (van Erven and Harremos, 2014). The KL divergence can be seen as the distance between two probability distributions (Jurafsky and Martin, 2025). In our case, these distributions refer to the predictions of the unlearned model and the original model’s output. By minimizing the KL divergence, we further penalize the unlearned model in case of too

much deviation from the original model aiming at preserving model utility despite unlearning (Ramakrishna et al., 2025). The KL divergence follows Equation 3:

$$\mathcal{L}_{KL}(\theta) = \mathbb{E}_{\mathcal{D}_{RT}} [KL(\pi_\theta(\cdot|x) || \pi_{ref}(\cdot|x))] \quad (3)$$

However, the KL divergence weighs both the probability distribution of the predictions of the unlearned model and the original model equally. To allow for more flexibility in this process, the present system extends the original KL divergence to the Rényi divergence (van Erven and Harremos, 2014). Equation 4 introduces the Rényi divergence as a loss term:

$$\mathcal{L}_{RY}(\theta; \alpha) = \mathbb{E}_{\mathcal{D}_{RT}} \left[\frac{1}{\alpha - 1} \ln(\pi_\theta^\alpha \pi_{ref}^{1-\alpha}) \right] \quad (4)$$

The Rényi divergence contains the additional positive hyperparameter α steering which areas of the distribution are especially emphasized in computing the distance between them. In the case of $\alpha < 1$, the divergence puts greater weight on outliers and special cases in the probability distribution of the predictions of the unlearned model π_θ where the probabilities of the original model π_{ref} are large. Consequently, it penalizes a more profound deviation from cases where the original model was ‘certain’. The opposite is happening when $\alpha > 1$. More emphasis is put on highly probable tokens following the unlearned model where the probabilities of the original model are smaller (Mondal, 2024). In other words, the former targets areas where the unlearned model assigns low probability to tokens, the original model was certain about, i.e., forgetting ‘correct’ tokens. The latter targets cases the original model was uncertain about but the unlearned model assigns a high probability, i.e., generating potentially ‘incorrect’ tokens. While the former, hence, targets over-forgetting, the latter targets wrongfully generated content. The KL divergence, approximated when $\alpha \rightarrow 1$, treats both distributions equally without putting emphasis on specific areas thereof. Importantly, following Equation 4, the divergence is undefined for $\alpha = 1$. Hence, the present system employs the KL divergence directly in this case giving roughly equal weight to all areas of the distributions.

Final Loss The final loss sums the individual loss terms weighted by the respective hyperparameters γ , δ , and ϵ as shown in Equation 5, where $\mathcal{L}_{NPO}$

is only applied to the forget set and $\mathcal{L}_{CE}$ as well as $\mathcal{L}_{RY}$ only to the retain set.

$$\begin{aligned} \mathcal{L}(\theta; \alpha, \beta, \gamma, \delta, \epsilon) = \\ = \gamma \mathcal{L}_{NPO}(\theta; \beta) + \delta \mathcal{L}_{CE}(\theta) + \epsilon \mathcal{L}_{RY}(\theta; \alpha) \end{aligned} \quad (5)$$

3.3 Low-Rank Adaptation

Low-rank adaptation (LoRA) is a method for parameter-efficient finetuning of large models. Since the models used in this task have approximately 1 billion and 7 billion parameters, finetuning can be very computationally expensive. LoRA utilizes low-rank decomposition matrices representing the pretrained weight matrices in the model. This way the number of trainable parameters can be reduced by multiple magnitudes facilitating efficient finetuning (Hu et al., 2021).

Additionally, implementing LoRA-based finetuning has the advantage of keeping the original model weights behind the LoRA adapter to be trained. Considering the implemented system, the reference model is needed for both $\mathcal{L}_{NPO}$ and $\mathcal{L}_{RY}$. While full finetuning or partial finetuning of some layers would require us to work with two models, the original one and the trainable one, LoRA being only an adapter attached to the original model enables us to run inference during training using the original model weights by temporarily deactivating the LoRA adapter (Broner and Helcl, 2025).

While Broner and Helcl (2025) focus only on the attention mechanism, the present system uses the same LoRA configuration as Premptis et al. (2025) encompassing attention and feed forward layers. Specifically, the LoRA adapter projects the query (q_proj) and the value module (v_proj) as well as the dense layers increasing (up_proj) and decreasing dimensionality (down_proj). Based on the experiments of Premptis et al. (2025), the matrix decomposition in the LoRA adapter follows a rank of 16 and uses a scaling factor α_{LoRA} of 64. In the case of the 1B model, the number of trainable parameters was thereby reduced to 0.57% and for the 7B model to 0.35% of the original number of parameters.

4 Experiments

Before presenting the results of finetuning the different models on the implemented system, this section provides a thorough overview of the chosen hyperparameters. A number of hyperparameters was held constant based on the results of Premptis

et al. (2025) and Broner and Helcl (2025). Note, however, that the hyperparameter tuning is by no means exhaustive and may be extended in future work.

Training Arguments Premptis et al. (2025) identified a *batch size* of 8 to be the most efficient in terms of the training outcome. Hence, a batch size of 8 was used for the experiments run on the 1B model, while a batch size of 4 was used for the 7B model due to computing restrictions. Regarding the number of *epochs*, they note that less than five epochs result in only weak unlearning while a small increase in the number of epochs to seven results in over-unlearning (Premptis et al., 2025). Hence, the most efficient number of epochs was identified to be five. Initial experiments with smaller batches or more epochs supported the findings of Premptis et al. (2025) leading to these settings being held constant in the following. Moreover, the authors state that a *learning rate* of 1e-5 yielded the best results (Premptis et al., 2025), while Broner and Helcl (2025) work with a learning rate of 1e-4. Initial experimentation revealed better training outcomes using the latter, which was used in all following experiments.

Data Sampling For the sequential data sampling, we need to define a chunk size as well as a positive ratio. Given the thorough experimentation of Premptis et al. (2025), the best resulting settings were implemented in the present system resulting in a default *number of chunks* of 32 with a *positive ratio* of 7. Consequently, the forget set was divided into 32 different chunks. The models were then trained on these chunks containing seven retain examples per forget example. For initial experiments on the 1B model with a lower batch size of 4, the positive ratio was adjusted to 3 leading to three retain examples per forget examples in the chunk. This, however, yielded worse results and was, hence, not further investigated on.

The implemented system also enables the deactivation of sequential sampling. In the case of training on the entire training data using a positive ratio of 7 led to instability in the loss. Consequently, the positive ratio was set to 1 leading to a balance between forget and retain samples when trained without sequential data sampling.

Loss Hyperparameters The parameter β in the NPO loss $\mathcal{L}_{NPO}$ was held constant to a value of 0.5 following Broner and Helcl (2025) for most

experiments. Additionally, some experiments were run with a value of 1 enabling stricter forgetting. The parameter α in the Rényi divergence $\mathcal{L}_{RY}$ was set to 0.5, 1, or 1.5 if the loss term was activated via ϵ . Note again, that $\alpha = 1$ directly uses the KL divergence in Equation 3 instead as the Rényi divergence is undefined but reaches its limit in the KL divergence if $\alpha \rightarrow 1$.

Lastly, the different terms in the final loss were scaled differently. In line with the original study, γ scaling $\mathcal{L}_{NPO}$ was held constant to 1 (Bronec and Helcl, 2025). This also goes in line with the scaling of the respective loss term in the system by Premptis et al. (2025). Regarding the loss over the retain set, initial experimentation revealed that activating both $\mathcal{L}_{CE}$ and $\mathcal{L}_{RY}$ led to a decline of model utility in the case of sequential sampling and under-unlearning in the case of training on the all data in one trainer. Consequently, the loss terms were scaled by 0.5 or 1 respectively, while the other was deactivated.

5 Results

Table 2 provides the results of the experiments carried out on the 7B parameter model on both training and validation set based on the identified best performing hyperparameters. These were determined during prior experiments on the smaller 1B model found in Appendix A out of which the best four settings were applied to the 7B model. To compare scores, Table 3 shows the best performing 1B model. Note that contrary to the task leaderboard (Ramakrishna et al., 2025), the final evaluation was carried out using the validation set and not the test set due to unavailability. The baseline MMLU accuracy lies at 0.243 for the 1B model and at 0.494 for the 7B model.

Notably, the implemented system manages to beat the best algorithms submitted to the shared task for the 7B model. **AJLS-NTUA**, as the best team, achieved a final score of 0.706 with a Task Aggregate score of 0.827 and an MMLU accuracy of 0.443 (Premptis et al., 2025). Note that the final score is inflated by a high MIA score that could not be computed for this study and lies at 0.635 if computed like in this study. The best performing 7B model based on the present system in Table 2 utilizes sequential data sampling with a chunk size of 32 and a ratio of 7:1 retain to forget examples trained and evaluated on the validation set. The implemented combination with the NPO loss on

Dataset	Task Aggregate	MIA Score	MMLU Acc	Final Score
$\alpha = 0.5; \beta = 0.5; \gamma = 1; \delta = 0; \epsilon = 0.5$				
Train	0.8939	0.9347	0.416	0.7482
Validation	0.9742	–	0.4405	0.7074
$\alpha = -; \beta = 0.5; \gamma = 1; \delta = 0.5; \epsilon = 0$				
Train	0.7797	0.8	0.3952	0.6583
Validation	0.9747	–	0.3994	0.6871
$\alpha = -; \beta = 0.5; \gamma = 1; \delta = 1; \epsilon = 0$				
Train	0.9206	0.8555	0.409	0.7284
Validation	0.9615	–	0.4662	0.7139
$\alpha = -; \beta = 1; \gamma = 1; \delta = 1; \epsilon = 0$				
Train	0.906	0.6206	0.4087	0.6451
Validation	0.9836	–	0.4672	0.7254

Table 2: Results on the 7B model. Trained on five epochs with a batch size of 4 and a learning rate of 1e-4. Sequential data sampling with a ratio of 7:1. $\alpha = -$ and $\epsilon = 0$ indicate a deactivated Rényi divergence.

Dataset	Task Aggregate	MIA Score	MMLU Acc	Final Score
$\alpha = -; \beta = 0.5; \gamma = 1; \delta = 1; \epsilon = 0$				
Train	0.9385	0.978	0.2554	0.724
Validation	0.9739	–	0.2406	0.6073

Table 3: Best performing 1B model. Trained on five epochs with batch size of 8 and a learning rate of 1e-4. Sequential data sampling with a ratio of 7:1. $\alpha = -$ and $\epsilon = 0$ indicate a deactivated Rényi divergence.

the forget sample with a β of 1 achieved a final score of 0.7254, a Task Aggregate score of 0.9836 and an MMLU accuracy of 0.4672 beating all three scores.⁷ Likewise, the best performing 1B model with $\beta = 0.5$ manages to beat the best prior Task Aggregate score and, if excluding the MIA score, outperforms the best model documented in Ramakrishna et al. (2025).

5.1 Influence of α

While the best performing models on the validation set did not rely on the Rényi divergence, utilizing the divergence when training and evaluating on the larger training set results in better performance suggesting greater utility of the Rényi divergence over the cross entropy loss for certain datasets. To better understand the influence of different values for α , Table 4 provides an overview of regurgitation and knowledge scores using values of 0.5, 1, and 1.5 while keeping all other hyperparameters constant with a deactivated cross entropy loss via $\delta = 0$.

While the model manages to perform both the forget and the retain task meaningfully under $\alpha = 0.5$ and $\alpha = 1$, setting α to 1.5 results in signif-

⁷Note that the 7B model utilizing Rényi divergence instead of the cross entropy loss outperforms this implementation when including the MIA score. However, the results on the training data are not suitable for the final evaluation as they are considered to have been seen during hyperparameter tuning.

Subset	Regurgitation Score			Knowledge Score		
	Task 1	Task 2	Task 3	Task 1	Task 2	Task 3
$\alpha = 0.5$						
<i>Forget Set</i>	0.0999	0.0083	0.1171	0.0	0.0019	0.0
<i>Retain Set</i>	0.9537	0.9849	0.9697	0.6796	0.9961	0.7358
$\alpha = 1$						
<i>Forget Set</i>	0.0656	0.0127	0.1032	0.012	0.0019	0.0
<i>Retain Set</i>	0.9862	0.9801	0.9858	0.9126	0.9922	0.8553
$\alpha = 1.5$						
<i>Forget Set</i>	0.0021	0.0	0.0	0.0	0.0	0.0
<i>Retain Set</i>	0.002	0.0	0.0003	0.0	0.0	0.0

Table 4: Regurgitation and knowledge scores per task and subset with varying α values on the 1B model. $\beta = 0.5$; $\gamma = 1$; $\delta = 0$; $\epsilon = 0.5$. Trained on five epochs with a batch size of 8 and a learning rate of $1e-4$. Training set.

icant over-unlearning where the model becomes incapable of memorizing information from the retain set and, hence, results in a Task Aggregate score of 0. This indicates that the stronger focus on wrongfully generated content under $\alpha > 1$ compared to a focus on forgetting correct tokens under $\alpha < 1$ is not suitable for the task at hand being unable to retain necessary information.

5.2 Sequential Data Sampling

Additional to running experiments based on varying α values in the Rényi divergence, some focus was put on comparing sequential data sampling to training on all data in one trainer. Table 5 provides an overview of two experiments with constant hyperparameters, but different sampling approaches. While training on all data at once results in a higher MMLU accuracy, the Task Aggregate score as well as the MIA score are significantly lower resulting in the final score being almost halved.

Sampling (Ratio)	Task Aggregate	MIA Score	MMLU Acc	Final Score
Sequential (7:1)	0.9466	0.8788	0.2332	0.6862
All (1:1)	0.6994	0.2074	0.2512	0.386

Table 5: Training results using sequential data sampling compared to training on all data at once on the 1B model. $\alpha = 0.5$; $\beta = 0.5$; $\gamma = \delta = \epsilon = 1$. Trained on five epochs with a batch size of 8 and a learning rate of $1e-4$. Training set.

The regurgitation and knowledge scores per task and subset in Table 6 showcase that the model is under-unlearning when training on all data at once. While the task-specific scores on the retain set remain comparable to the model trained using sequential data sampling, the task-specific scores on the forget set are unsatisfactory.

Subset	Regurgitation Score			Knowledge Score		
	Task 1	Task 2	Task 3	Task 1	Task 2	Task 3
<i>Sequential (7:1)</i>						
<i>Forget Set</i>	0.1384	0.0041	0.1355	0.012	0.0019	0.0
<i>Retain Set</i>	0.9889	0.9996	0.983	0.8932	0.9942	0.8428
<i>All (1:1)</i>						
<i>Forget Set</i>	0.2553	0.0194	0.161	0.6747	0.0206	0.6974
<i>Retain Set</i>	0.976	0.9958	0.9307	0.9612	0.9804	0.9623

Table 6: Regurgitation and knowledge scores per task and subset with sequential data sampling compared to training on all data at once on the 1B model. $\alpha = 0.5$; $\beta = 0.5$; $\gamma = \delta = \epsilon = 1$. Trained on five epochs with a batch size of 8 and a learning rate of $1e-4$. Training set.

5.3 Training Time

To better understand the training time-performance trade-off of using the Rényi divergence and sequential data sampling, Table 7 and 8 document the differences in training times of different experiments conducted on the 1B model.

Comparing sequential data sampling to training on all data at once indicates a vast increase in training time, where the former is almost 6.8x slower (see Table 7). While the performance increase is also noticeably high, the steep increase in training time can be considered an important caveat of this method. Despite the inefficiency during training, the performance gain is a strong argument for sequential data sampling.

Sampling (Ratio)	Training Time (sec)	Final Score
All (1:1)	269	0.386
Sequential (7:1)	1826	0.6862

Table 7: Training time when training on all data at once compared to sequential data sampling during training on the 1B model. $\alpha = 0.5$; $\beta = 0.5$; $\gamma = \delta = \epsilon = 1$. Trained on five epochs with a batch size of 8 and a learning rate of $1e-4$. Training set.

The α hyperparameter in the Rényi divergence has a smaller influence on the training time (see Table 8). A value of 0.5 results in the longest training time, while also achieving the best final score. The shortest training time occurs when setting α to 1. This may be due to the implementation of the KL divergence to circumvent the need to compute the limit of the function when $\alpha \rightarrow 1$ as explained in § 3.2. The difference in training time to a value of 1.5 is small which may be due to the stronger focus on the center of the distribution under $\alpha > 1$. Considering the performance, however, this setting is

not desirable. Based on both factors, training time and performance, $\alpha = 1$ seems to be the most efficient choice when activating the Rényi divergence effectively using the KL divergence.

$\alpha - \mathcal{L}_{RY}$	Training Time (sec)	Final Score
0.5	1840	0.7041
1	1028	0.6734
1.5	1040	0.392

Table 8: Training time with varying α values on the 1B model. $\beta = 0.5$; $\gamma = 1$; $\delta = 0$; $\epsilon = 0.5$. Sequential data sampling with ratio 7:1. Trained on five epochs with a batch size of 8 and a learning rate of $1e-4$. Training set.

5.4 Qualitative Analysis

Lastly, a small manual evaluation was conducted on the best overall unlearned model to support the quantitative metrics-based evaluation and understand how the unlearned model generates output on the retain and forget data. The model corresponds to the 7B model trained and evaluated on the validation set with equally scaled NPO and cross entropy loss, but deactivated Rényi divergence (see Table 2). To qualitatively analyze the generated output, a random sample of both retain and forget set and each task was passed on to the model. Table 9 shows exemplary output.

The model performs well and accurate on the randomly sampled retain examples. The displayed forget examples show that the model indeed does not generate the reference output. Except for the example from Task 3, it instead generates repeated line breaks and the word ‘download’ regardless of a question-answering scenario in Task 2 or the completion of a document like in Task 1. While this serves as additional support that the model has indeed unlearned the content successfully, it also opens up further questions regarding the model utility. While the MMLU accuracy was indeed the highest compared to the alternative hyperparameter combinations on the 7B model, it is still a noticeable drop from the original 0.494 of the baseline 7B model to 0.4672 of the unlearned model. At the same time, nonsensical output to questions the model has unlearned the answer to, might even be desirable considering challenges like seemingly factual non-factual content generation in LLMs (Hicks et al., 2024).

6 Conclusion

This study implements a novel system combining previous efforts in the field of unlearning LLMs from sensitive content in their training data. The present system incorporates sequential data sampling during training and NPO to support stable unlearning. In addition, a new approach was tested within this algorithm implementing Rényi divergence as an additional term in the loss function to enable more flexibility when comparing the unlearned model’s to the original model’s outputs. While the results do not provide a clear indication on whether the Rényi divergence is inferior or superior to the simpler cross entropy loss in terms of performance, the implementation using NPO clearly outperformed the previously best system in the shared task. Nevertheless, model utility dropped in almost all scenarios suggesting further opportunities to improve current systems.

Limitations

The present study provides thorough discussions of the influence of different training scenarios on the proposed unlearning algorithm. Nevertheless, the hyperparameter tuning is by no means exhaustive and may be significantly extended in future work. This concerns both training arguments such as number of epochs, batch size, or learning rate as well as different combinations of the loss hyperparameters. Moreover, the hyperparameter tuning was carried out on the 1B model after which the best settings were merely applied to the larger 7B alternative. This, however, does not imply that these settings are also the most suitable for the OLMo model of that size. Lastly, the final evaluation was carried out using the validation set, whereas the original task used a private test set leading to limited comparability with prior submissions.

References

- Saransh Agrawal and Kuan-Hao Huang. 2025. [SHA256 at SemEval-2025 Task 4: Selective Amnesia – Constrained Unlearning for Large Language Models via Knowledge Isolation](#). *Preprint*, arXiv:2504.12996.
- Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. [On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?](#) In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT ’21, page 610–623, New York, NY, USA. Association for Computing Machinery.

Subset	Input	Reference Output	Generated Output
Retain - Task 1	Who is the petite woman with a warm smile and sparkling eyes that Eddy approached in the park?	Lily	Lily
Retain - Task 2	What is the birth date of Marguerite Red?	1971-12-14	1971-12-14
Retain - Task 3	Mary Houghton Mary Houghton is co-founder of ShoreBank [...] Houghton received a B.A. "cum laude" from Marquette	University and an M.A. in International Studies from Johns Hopkins University. [...]	University and an M.A. in International Studies from Johns Hopkins University. [...]
Forget - Task 1	Nestled in the heart of Santa Clara [...] She was a woman of stories, a keeper of secrets, and a silent observer of	love's unfolding dramas.	download download download download [...]
Forget - Task 2	What is the home address of Aimee Plum?	3620 Bentfield Drive, Fayetteville, AR, 72703.	\n [...] \n download [...]
Forget - Task 3	Who is the chef that served as a culinary mentor for the Bezalel Academy of Art and Design students in 2021?	Haim Cohen	John H.

Table 9: Examples of output generated from best performing unlearned 7B model. $\alpha = -$; $\beta = 1$; $\gamma = 1$; $\delta = 1$; $\epsilon = 0$. Trained on five epochs with a batch size of 4 and a learning rate of 1e-4. Sequential data sampling with a ratio of 7:1. Validation set. $\alpha = -$ and $\epsilon = 0$ indicate a deactivated Rényi divergence.

- Jan Bronec and Jindřich Helcl. 2025. *Atyaephyra at SemEval-2025 Task 4: Low-Rank NPO*. *Preprint*, arXiv:2503.13690.
- Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. *Extracting Training Data from Large Language Models*. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650. USENIX Association.
- Michael Duan, Anshuman Suri, Niloofar Miresghalah, Sewon Min, Weijia Shi, Luke Zettlemoyer, Yulia Tsvetkov, Yejin Choi, David Evans, and Hananeh Hajishirzi. 2024. *Do Membership Inference Attacks Work on Large Language Models?* *Preprint*, arXiv:2402.07841.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. *Measuring Massive Multitask Language Understanding*. *Preprint*, arXiv:2009.03300. arXiv:2009.03300v3.
- Michael Townsen Hicks, James Humphries, and Joe Slater. 2024. *ChatGPT is bullshit*. *Ethics and Information Technology*, 26(2):38.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. *LoRA: Low-Rank Adaptation of Large Language Models*. *Preprint*, arXiv:2106.09685.
- Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, L  lio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, and 7 others. 2024. *Mixtral of Experts*. *Preprint*, arXiv:2401.04088.
- Daniel Jurafsky and James H. Martin. 2025. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*, 3rd edition. Online manuscript released January 12, 2025.
- Bo Liu, Qiang Liu, and Peter Stone. 2022. Continual learning and private unlearning. In *Conference on Lifelong Learning Agents*, page 243–254. PMLR.
- Saankhya Mondal. 2024. Understanding KL Divergence, Entropy, and Related Concepts. Towards data science, <https://towardsdatascience.com/understanding-kl-divergence-entropy-and-related-concepts-75e766a2fd9e/>, last accessed 15.05.2025.
- OpenAI. 2022. *Introducing ChatGPT*. <https://openai.com/index/chatgpt/>, last accessed 20.05.2025.
- Iraklis Premptis, Maria Lymperaiou, Giorgos Filandrianos, Orfeas Menis Mastromichalakis, Athanasios Voulodimos, and Giorgos Stamou. 2025. *AILS-NTUA at SemEval-2025 Task 4: Parameter-Efficient Unlearning for Large Language Models using Data Chunking*. *Preprint*, arXiv:2503.02443.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. *Direct Preference Optimization: Your Language Model is Secretly a Reward Model*. In *Advances in Neural Information Processing Systems*, volume 36, pages 53728–53741. Curran Associates, Inc.
- Anil Ramakrishna, Yixin Wan, Xiaomeng Jin, Kai-Wei Chang, Zhiqi Bu, Bhanukiran Vinzamuri, Volkan Cevher, Mingyi Hong, and Rahul Gupta. 2025. *SemEval-2025 Task 4: Unlearning sensitive content from Large Language Models*. *Preprint*, arXiv:2504.02883.

Tim van Erven and Peter Harremoës. 2014. [Rényi Divergence and Kullback-Leibler Divergence](#). *IEEE Transactions on Information Theory*, 60(7):3797–3820.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. [Attention is All you Need](#). In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Haoming Xu, Shuxun Wang, Yanqiu Zhao, Yi Zhong, Ziyang Jiang, Ningyuan Zhao, Shumin Deng, Huajun Chen, and Ningyu Zhang. 2025. [ZJUKLAB at SemEval-2025 Task 4: Unlearning via Model Merging](#). *Preprint*, arXiv:2503.21088.

Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. 2024. [Negative Preference Optimization: From Catastrophic Collapse to Effective Unlearning](#). *Preprint*, arXiv:2404.05868.

A Experiments on the 1B model

The following sections provide an overview of the different experiments carried out on the 1B model. The final scores of the best four settings applied to the 7B model are marked in bold.

A.1 Equal Scaling of Loss Terms

Table 10 provides an overview of experiments with varying α and equal scaling of all loss terms. The model was trained and evaluated on the training set.

Sampling (Ratio)	Task Aggregate	MIA Score	MMLU Acc	Final Score
$\alpha = 0.5; \beta = 0.5; \gamma = \delta = \epsilon = 1$				
Sequential (7:1)	0.9466	0.8788	0.2332	0.6862
All (1:1)	0.6994	0.2074	0.2512	0.386
$\alpha = 1; \beta = 0.5; \gamma = \delta = \epsilon = 1$				
Sequential (7:1)	0.8981	0.9736	0.2277	0.6998
All (1:1)	0.6424	0.0074	0.254	0.3013

Table 10: Training results using all loss terms equally with varying α . Trained on five epochs with a batch size of 8 and a learning rate of 1e-4. Training set.

A.2 Deactivated Cross Entropy Loss

Table 11 shows all experiments ran with δ set to 0 resulting in the deactivation of the cross entropy loss term with varying α and β values. The model was trained and evaluated on the training set.

A.3 Deactivated Rényi Divergence

Table 12 shows experiments with ϵ set to 0 resulting in the deactivation of the Rényi divergence with varying β values. The model was trained and evaluated on the training set.

$\alpha - \mathcal{L}_{RY}$	Task Aggregate	MIA Score	MMLU Acc	Final Score
$\beta = 0.5; \gamma = 1; \delta = 0; \epsilon = 0.5$				
0.5	0.9102	0.9525	0.2496	0.7041
1	0.9574	0.8308	0.2322	0.6734
1.5	0	0.9392	0.2368	0.392
$\beta = 1; \gamma = 1; \delta = 0; \epsilon = 1$				
0.5	0.937	0.8064	0.2432	0.6622
1	0.9562	0.8468	0.2431	0.682
1.5	0	0.878	0.2626	0.3802

Table 11: Training results with deactivated $\mathcal{L}_{CE}$ via $\delta = 0$. Trained on five epochs with a batch size of 8 and a learning rate of 1e-4. Sequential data sampling with ratio 7:1. Training set.

$\beta - \mathcal{L}_{NPO}$	Task Aggregate	MIA Score	MMLU Acc	Final Score
$\alpha = -; \gamma = 1; \delta = 1; \epsilon = 0$				
0.5	0.9385	0.978	0.2554	0.724
1	0.9685	0.9324	0.255	0.7186

Table 12: Training results with deactivated $\mathcal{L}_{RY}$ via $\epsilon = 0$. Trained on five epochs with a batch size of 8 and a learning rate of 1e-4. Sequential data sampling with ratio 7:1. Trained on the training set.

A.4 Combining Algorithms of AILS-NTUA and Atyaephyra

Table 13 provides an overview of experiments run combining the approaches of Premptis et al. (2025) and Bronec and Helcl (2025). This entails a constant β to 0.5 (Bronec and Helcl, 2025), scaling the cross entropy loss (δ) by 0.5 and deactivating the Rényi divergence ($\epsilon = 0$). The best result was achieved training on five epochs with a batch size of 8 (Premptis et al., 2025) and a learning rate of 1e-4 (Bronec and Helcl, 2025).

Epochs (Batch size)	Lr	Sampling (Ratio)	Task Aggregate	MIA Score	MMLU Acc	Final Score
$\alpha = -; \beta = 0.5; \gamma = 1; \delta = 0.5; \epsilon = 0$						
5 (4)	1e-4	Sequential (3:1)	0.8989	0.6508	0.2547	0.6015
5 (8)	1e-4	Sequential (7:1)	0.95	0.9514	0.2347	0.712
5 (8)	1e-4	All (1:1)	0.9612	0.4647	0.2325	0.5528
5 (8)	1e-5	Sequential (7:1)	0.9128	0	0.2307	0.3812
10 (8)	1e-4	Sequential (7:1)	0.9592	0.7124	0.2406	0.6374

Table 13: Training results on the combination of both approaches with varying training parameters. $\alpha = -$ and $\epsilon = 0$ indicate a deactivated Rényi divergence.

Combining Winning Strategies. Building on Submissions to SemEval 2025 Shared Task 4: Unlearning Sensitive Content from Large Language Models

Emil Poiesz
Integreat / UiO
emilpo@uio.no

Abstract

Ensuring that sensitive information is not present in a Large Language Model (LLM) has become an increasingly important task. With questions about the legality of the use of copyrighted materials in LLM training data, and ensuring that LLMs follows privacy laws across jurisdictions, unlearning sensitive content without expensive re-training of the model can be an efficient way to comply with laws and regulations. This area of research featured in a shared task at SemEval-2025 and a diverse set of strategies were present among the top performing submissions. This paper successfully combines two of the top performing methods; *sequential unlearning with data chunking*, and *Negative Preference Optimization + KL divergence + Gradient Descent loss*. By combining these two methods, our implementation manages to balance the strengths of both strategies leading to competitive final scores.

1 Introduction

Since transformer based AI models became widely available, the presence of copyrighted materials in the training data of these models has been a debated topic¹. The General Data Protection Regulation (GDPR) also requires LLMs to comply with European law which guarantees a *right to be forgotten*². Since LLMs have shown a tendency to memorize their training data (Carlini et al., 2021), such concerns can be expensive and challenging to tackle. One way to guarantee that sensitive information is not contained in the model is to retrain the models from scratch after removing the sensitive information from the training data, but such frequent re-training of models is very computationally expensive and can make such models less economically viable.

¹<https://www.nytimes.com/2023/12/27/business/media/new-york-times-open-ai-microsoft-lawsuit.html>

²<https://gdpr-info.eu/art-17-gdpr/>

At SemEval-2025 a task was presented which tries to tackle this challenge. SemEval-2025 Task-4: Unlearning sensitive content from Large Language Models (Ramakrishna et al., 2025) is a shared task meant to explore new strategies for *machine unlearning* in the LLM field. This line of research focuses on targeting and removing specific information without the need to re-train the entire model. An important aspect, which makes this a challenging problem, is that the LLM needs to maintain its high language generation capabilities after the unlearning process. Previously in machine unlearning, algorithms were often based on the *gradient ascent* loss applied to sensitive data, where the sign of the regular gradient descent loss is flipped, but the unbounded nature of gradient ascent would often lead to catastrophic collapse of the LLMs language generating ability. Where gradient descent loss is minimized, bounded by 0, gradient ascent is maximized with no upper-bound. Gradient ascent models tended to over-forget sensitive information leading to degraded performance on general language generation. This problem of over-forgetting has previously been addressed with an adaptation of *Direct Preference Optimization* (DPO) (Rafailov et al., 2024) called *Negative Preference Optimization* (NPO) (Zhang et al., 2024) with promising results. Where DPO works by increasing the likelihood of preferred positive samples and decreasing the likelihood of not-preferred negative samples, NPO only considers negative samples.

One of the top submissions to SemEval-2025, ZJUKLAB (Xu et al., 2025), used NPO on samples in the forget set to lower probabilities of sensitive data as well as using regular *gradient descent* (GD) in combination with the *Kullback-Leibler divergence* (KLD) on the retain dataset to maintain performance on documents meant to be remembered. Another top submission, AILS-NTUA

(Premptis et al., 2025), leveraged a novel dataset handling strategy called *data chunking* where they carefully controlled the makeup of the training set fed to the model in each batch to contain a ratio between forget and retain samples such that the model is over-exposed to positive samples.

Our main contribution in this paper is experimental results which show that combining the data chunking strategy from AILS-NTUA and the NPO-GD-KLD loss from ZJUKLAB can lead to competitive results. The code for the experiments is available at https://github.uio.no/emilpo/IN9550_exam

2 Preliminaries

In this section we will describe the makeup of the dataset, how we pre-process the dataset, and how model performance is evaluated. An overview of the process is outlined in algorithm 1.

Datasets The baseline models were pre-trained to memorize a set of documents. The set of memorized documents is split into two sets of equal size; a forget set with information we would like to remove from the model, and a retain set with information we would like the model to remember. A sample of the forget and retain set is held back as a validation set, while the majority is used as a training set. The sizes of the datasets can be seen in table 1.

	Forget	Retain	total
Training	1112	1136	2248
Validation	254	278	532
total	1366	1414	1780

Table 1: Dataset sizes

The dataset was further divided into three sub-tasks each containing a different type of sensitive data; (1) creative content consisting of short stories generated by a Mixtral model (Jiang et al., 2023), (2) synthetically generated Personally Identifiable Information (PII) also generated by a Mixtral model (Jiang et al., 2023), and (3) biographies of real people sampled from Wikipedia in the range of 100 to 200 words long. In this work we did not distinguish between samples from these three sub-tasks instead choosing to treat them all equally.

Data chunking We divide the forget set into distinct chunks. This data chunking strategy is based on *sequential unlearning* (Jang et al. 2022) and was

used in the top performing submission by AILS-NTUA (Premptis et al. 2025). Sequential unlearning was shown to increase stability of the model during training, reducing the risk of catastrophic collapse of model capabilities. In each chunk, we stack samples from the forget set and retain set with the given Forget-to-Retain ratio (1 : n). This ratio n makes sure that the model is continuously exposed to positive samples in order to prevent over-forgetting. Because the forget and retain sets are of similar size, we have to sample from the retain set using a cyclic sampler. This means that the model will see each sample from the retain set multiple times before it has seen every sample from the forget set.

Loss function The loss on the negative samples from the forget set is calculated using NPO, while the loss on the positive samples from the retain set is calculated using a combination of GD and KLD. The total loss is then a combination of these three loss-objectives:

$$\mathcal{L}_{total} = \alpha \mathcal{L}_{npo} + \beta \mathcal{L}_{gd} + \gamma \mathcal{L}_{kld} \quad (1)$$

The hyperparameters α, β, γ allow us to control the balance between the loss on the forget set and the retain set depending on model performance.

Algorithm 1: (Adapted) Sequential Unlearning from (Premptis et al., 2025)

- 1 **input:** Forget set D_f and Retain set D_r ,
Forget-Retain ratio n , Learning rate η ,
Model parameters θ
 - 2 Divide D_f into N chunks:
 - 3 $D_f = \{D_f^1, \dots, D_f^N\}$
 - 4 **for** $i = 1$ **to** N **do**
 - 5 Construct data chunk by cyclically
 sampling from D_r such that
 $|D_r^i| = n|D_f^i|$
 - 6 **for each optimization step do**
 - 7 Perform forward pass on D_f^i and D_r^i
 - 8 Calculate loss:
 - 9 $\mathcal{L}_{total} = \alpha \mathcal{L}_{NPO}(D_f^i) +$
 $\beta \mathcal{L}_{KL}(D_r^i) + \gamma \mathcal{L}_{GD}(D_r^i)$
 - 10 Update model parameters:
 - 11 $\theta \leftarrow \theta - \eta \nabla_{\theta} \mathcal{L}_{total}$
-

2.1 Evaluation

One challenge with evaluating model unlearning is that we cannot evaluate a model on how well it has forgotten data it has not been told to forget. The organizers of the task decided not to release their private test set used to evaluate submissions during the shared task³ because they then could not ensure that submissions had not re-trained the models from scratch on the retain set of documents only. Instead the organizers asked participants to submit working code which the organizers would run and evaluate on their privately held test set themselves. The resulting final scores was calculated with a combination of three metrics; *Task Memorization score*, *Membership Inference Attack score*, and *Massive Multitask Language Utility score*.

Task memorization The task memorization metric is calculated as a combination of the regurgitation rate, a sentence completion metric based on the longest co-occurring sequence between the response and a gold standard (ROUGE-L), and the Knowledge Test Accuracy metric, a measure of accuracy on a set Question-Answer (QA) prompts based on subtasks (1) and (3) in the forget and retain sets. The gold standard for the QA prompts were verified by three different LLMs and QA pairs where any one of the verification models were unable to answer accurately were discarded. High task memorization scores on the retain set and low task memorization scores on the forget set is optimal.

Membership Inference Attack (MIA) The MIA evaluation metric (Shokri et al., 2017) measures data leakage from the training set. In this setting we are evaluating if we can infer that a document from the forget set was present in the training data after unlearning has completed. The metric evaluates the performance of the model against samples taken from the forget set with comparable samples from the retain set. The model should not be able to infer information from the forget set while maintaining utility on the retain set.

Massive Multitask Language Utility (MMLU) The MMLU evaluation metric (Hendrycks et al., 2021) measures the models utility on a wide variety of subjects and is used as a general benchmark of LLM performance. This metric helps us measure the degree of model degradation after unlearning.

³The organizers has still not released the test set by May 30, 2025

Optimally, we would like the utility of the model to not degrade, which means scoring close to 0.243 for the baseline 1B-parameter model and close to 0.493 for the baseline 7B-parameter model. The organizers of the task set as a threshold for consideration that the 7B-parameter model not degrade below a MMLU score of 0.371 (75% of the baseline model utility). This was to ensure that the model would maintain acceptable level of utility after unlearning.

3 Experimental Setup

We ran the experiments on one NVIDIA A100 80GB GPU. We experimented with two models based on *OLMo-7B-0724-Instruct-hf* and *OLMo-1B-0724-hf* which were pre-trained to memorize a set of documents.

Setting	Value
Gradient accumulation steps	4
Learning rate	1e-4
Weight decay	0.01
Epochs	5
Max new tokens	256
Chunk size	32
Forget-Retain ratio	1:7
LoRA rank	32
LoRA alpha	32
LoRA dropout	0.05

Table 2: Common parameter values

	α	β	γ
Config 0	0.4	0.4	0.2
Config 1	0.3	0.3	0.4
Config 2	0.2	0.4	0.4

Table 3: Configuration specific values

We experimented with the smaller 1B-parameter model to determine an optimal set of hyperparameters before running the best performing configurations with the larger 7B-parameter model. We used Low Rank Adaption (LoRA) methods (Hu et al. 2021) from the Parameter-Efficient Fine-Tuning (PEFT) library in Huggingface to efficiently train large models. The LoRA adapters were merged into the model at the end of training. In these experiments we focussed on testing different values for the α, β, γ hyperparameters in the total loss function (1) in order to find a balance between the

three losses. Consistent parameter values between each configuration can be seen in table 2 and the hyperparameter values we tested in each configuration can be seen in table 3.

4 Results

Here we will comment and compare our results with the results of the two submissions on which this work is inspired. It is important to keep in mind that we are comparing the results of models run on a privately held test set which has not yet been released, with our results which were training and evaluated on the training set. Our results therefore are not directly comparable to the results of the submissions.

1B-parameter model The scores for all the 1B-parameter models can be seen in table 4. Compared to the two submissions we took inspiration from, our 1B-parameter models outperforms the submission from ZJUKLAB on the final score but falls short of the submission from AILS-NTUA. AILS-NTUA scored highly on both the task aggregate and the MIA metrics, but while ZJUKLAB had high scores on the task aggregate metric, they had significantly worse scores on the MIA metric. Our task aggregate scores lower than both AILS-NTUA and ZJUKLAB, but our MIA score was better than ZJUKLAB leading to us scoring a higher final score. We noted that the MMLU scores improved from the baseline 0.243 to 0.248 with configuration 1. This is the opposite of what we would expect when unlearning information and so we ran extra experiments which are reported in the appendix. The initial high final scores on the validation data, and the high MMLU score on the training data both seem to be an outliers.

Validation set The data used to evaluate the MIA metric only contained forget samples from the training set. Since there were no MIA data for the validation set, the MIA metric scores of the validation model will be impacted which in turn influences the final score of the validation model. Therefore, when evaluating on the validation set, we see a consistent shift in the scores with both the 1B and 7B models. Except for configuration 1, which we explore more in the appendix, there is a noticeable drop in performance on the MIA score on the validation set both with the 1B-parameter models and the 7B-parameter model. With limited time available, we decided not to generate MIA data for

the validation set so we will not comment on the results on the validation set but we still report them here for completeness sake.

7B-parameter model The scores for the 7B-parameter models can be seen in table 5. We trained the 7B-parameter model with configuration 0, the best performing configuration on the 1B-parameter models. Our 7B-parameter model managed a 5% improvement in final score performance over the winning submission of AILS-NTUA. Our model managed a final score of 0.743, outperforming the AILS-NTUA submission on the task aggregate score and outperforming the ZJUKLAB submission on the MIA score. Our model also displayed the lowest amount of performance degradation, scoring a MMLU score of 0.480, compared to the 0.493 baseline.

5 Discussions

Conclusions One pattern that stood out was that the best performing configurations with the 1B-parameter model correlated with the increase in the α parameter where higher α values resulted in higher final scores. The α parameter controls the weight given to the NPO loss on negative samples from the forget set. This indicates that forgetting information is important in the unlearning sensitive information task.

Our 7B-parameter model improved on the best performing model by 5% in the final score metric when comparing performance on the available training data to the submission scores on test data. Our strategy of combining the methods from the two best performing submissions draws on the strength of both approaches. We managed to improve on ZJUKLABs poor MIA scores by applying the data chunking strategy from AILS-NTUA while still maintaining high task aggregate scores close to ZJUKLABs submission. While our model managed to score second best in both metrics, the balanced approach resulted in an overall higher final score while at the same maintaining the highest model utility scores after unlearning.

Limitations Since the organizers of the shared task did not release the testing data, we were not able to evaluate our model on the same data as the submissions were evaluate on. This means that while our models managed a 5% increase in the final score metric, we are then comparing our

Model	Final Score	Task Aggregate	MIA Score	MMLU avg
AILS-NTUA 1B	0.688	0.964	0.857	0.242
ZJUKLAB 1B	0.483	0.915	0.292	0.243
Results on training set				
1B config 0	0.611	0.849	0.746	0.238
1B config 1	0.585	0.876	0.631	0.248
1B config 2	0.540	0.822	0.561	0.238
Results on validation set				
1B config 0	0.577	0.949	0.529	0.252
1B config 1	0.610	0.925	0.660	0.245
1B config 2	0.461	0.904	0.224	0.255

Table 4: Scores of the 1B-parameter models from the submissions on the test set and our scores on the training and validation sets

Model	Final Score	Task Aggregate	MIA Score	MMLU avg
AILS-NTUA 7B	0.706	0.827	0.847	0.443
ZJUKLAB 7B	0.487	0.944	0.048	0.471
Results on training set				
7B config 0	0.743	0.915	0.833	0.480
Results on validation set				
7B config 0	0.594	0.966	0.351	0.466

Table 5: Scores of the 7B-parameter models from the submissions on the test set and our scores on the training and validation sets

scores on the training data with the submissions on test data, and therefore isn’t a fair comparison. Another limitation of our work is the limited scope of parameters that we are testing, we address this in the future work section.

Future work While we managed to improve on the final score of the submissions, our experiments used a very limited scope for testing hyperparameters. Given the promising results on these limited set of configurations there is a lot of potential for finding improvements by testing more configurations. There is also space to experiment with different hyperparameters such as the LoRA configurations as well as the forget-to-retain ration n . There is also potential to do subtask specific handling of the training data. Reviewing the paper and code from other submissions we found that this type of experimentation was common, though because of limited time we did not experiment with the different subtasks. This work treated every sample equally but a more tailored approach could yield further improvements. Another interesting avenue we would have liked to experiment with was the merging of LoRA adapters which were

trained with different configurations as is done in ZJUKLABs submission. Future experiments with larger models to see the impact of models size is also something that would be interesting to research.

Since this is a new line of research in the LLM field, there are many possible avenues for further improvements. The successful combination of these two strategies shows that it is possible to draw from the strengths of different methods, and there might be other method combinations among the submissions that combined could result in an improvement.

Use of AI Since the use of AI assistants are a recent addition in academia (within the last few years), we believe it is important to explicitly state how AI assistants were used. Since there presently are no universally accepted formats for disclosing the use of AI assistants, we take the liberty of writing a paragraph about our use here.

During the writing of this paper, the use of AI assistance was limited to asking for clarification on

certain aspects when reading related works. AI assistance was also elicited for suggestions on how to solve smaller programming tasks during the experimental phase. We did not copy and paste excerpts from this text to ask for feedback, nor did we copy and paste code provided by AI models.

References

- Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. [Extracting training data from large language models](#). *Preprint*, arXiv:2012.07805.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. [Measuring massive multitask language understanding](#). *Preprint*, arXiv:2009.03300.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. [Lora: Low-rank adaptation of large language models](#). *Preprint*, arXiv:2106.09685.
- Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. 2022. [Knowledge unlearning for mitigating privacy risks in language models](#). *Preprint*, arXiv:2210.01504.
- Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L  lio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timoth  e Lacroix, and William El Sayed. 2023. [Mistral 7b](#). *Preprint*, arXiv:2310.06825.
- Iraklis Prempitis, Maria Lymperaio, Giorgos Filandrianos, Orfeas Menis Mastromichalakis, Athanasios Voulodimos, and Giorgos Stamou. 2025. [Ails-ntua at semeval-2025 task 4: Parameter-efficient unlearning for large language models using data chunking](#). *Preprint*, arXiv:2503.02443.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. [Direct preference optimization: Your language model is secretly a reward model](#). *Preprint*, arXiv:2305.18290.
- Anil Ramakrishna, Yixin Wan, Xiaomeng Jin, Kai-Wei Chang, Zhiqi Bu, Bhanukiran Vinzamuri, Volkan Cevher, Mingyi Hong, and Rahul Gupta. 2025. [Semeval-2025 task 4: Unlearning sensitive content from large language models](#). *Preprint*, arXiv:2504.02883.
- Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. [Membership inference attacks against machine learning models](#). In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18.
- Haoming Xu, Shuxun Wang, Yanqiu Zhao, Yi Zhong, Ziyang Jiang, Ningyuan Zhao, Shumin Deng, Hua-jun Chen, and Ningyu Zhang. 2025. [Zjuklab at semeval-2025 task 4: Unlearning via model merging](#). *Preprint*, arXiv:2503.21088.
- Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. 2024. [Negative preference optimization: From catastrophic collapse to effective unlearning](#). *Preprint*, arXiv:2404.05868.

A Configuration 1

Since configuration 1 had some anomalous results between the training model and the validation model, we ran the experiments three times to understand the variation. The results of the three runs can be seen in table 6. In the original experiment with one training run we remarked on higher than baseline MMLU scores and an increase in final scores on the validation data when the trend showed that scores tended to decrease. When running the training three times we are able to calculate the mean and standard deviation of the training which gives us confidence intervals. We see that for the model trained on the training data, both the Task Aggregate score and the MMLU score is very stable but that the higher than baseline MMLU score is above average performance, with the mean score 0.2412 being lower than baseline 0.243. We also see that the anomalous increase in performance on the validation set is the result of an over performance of the first validation training run, scoring significantly higher than the mean in the MIA metric.

Training	Final Score	Task Aggregate	MIA Score	MMLU avg
	0.5855	0.8762	0.6314	0.2489
	0.6823	0.8480	0.9592	0.2398
	0.4595	0.8536	0.2900	0.2349
mean ($\pm$ std)	0.5757 ($\pm$ 0.11)	0.8592($\pm$ 0.01)	0.6268($\pm$ 0.33)	0.2412($\pm$ 0.01)

Validation	Final Score	Task Aggregate	MIA Score	MMLU avg
	0.6102	0.9255	0.6602	0.2451
	0.5127	0.9382	0.3612	0.2386
	0.3330	0.4741	0.2900	0.2349
mean ($\pm$ std)	0.4853 ($\pm$ 0.14)	0.7792($\pm$ 0.26)	0.4371($\pm$ 0.19)	0.2395($\pm$ 0.01)

Table 6: Scores of the 1B-parameter model over three runs using configuration 1 on the training and validation data.

Unlearning Sensitive Content from Large Language Models through Synthetic Data Generation

Henrik Bergli, Henrik Olafsen, Said Hemaz
University of Oslo

Abstract

Large language models (LLMs) often retain sensitive or private information from their training data, raising concerns around privacy and data governance. This work investigates a method for targeted unlearning, where specific knowledge is removed from a model using synthetic data. Rather than retraining from scratch, we fine-tune a pre-trained 1B parameter OLMo-based model with modified versions of sensitive content, replacing correct answers with realistic but incorrect alternatives. Our approach draws on the SemEval 2025 unlearning task and evaluates the impact of different prompting strategies, synthetic dataset configurations, and training hyperparameters. Results indicate that carefully crafted synthetic inputs, particularly using counterfactual rewriting, can help overwrite memorized information (inverted forget knowledge score: **0.451**), while maintaining performance on unrelated tasks (retain knowledge: **0.995**, MMLU: **0.240**). The final model achieved a task aggregate score of **0.520**. These findings suggest that fine-tuning on synthetic data offers a feasible strategy for selective unlearning in LLMs, with minimal observed degradation in general capability.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in various natural language tasks (OpenAI et al., 2024), but their tendency to memorize training data raises significant concerns about privacy, copyright, and the right to be forgotten (Carlini et al., 2019). As these models become more pervasive in applications handling sensitive information, the ability to selectively remove or "unlearn" certain data while maintaining model utility becomes increasingly important.

Machine unlearning, particularly for LLMs, presents unique challenges due to the distributed nature of knowledge representation across model

parameters. Traditional approaches often require complete retraining, which is prohibitively expensive for large models. Our work addresses this challenge by proposing an unlearning method that leverages synthetic data generation to help models forget specific information.

We build on the SemEval 2025 shared task framework (Ramakrishna et al., 2025), which provides datasets for evaluating targeted unlearning in LLMs. Our approach involves creating synthetic data where sensitive information is replaced with plausible but incorrect alternatives, with the aim of overwriting the model's previous knowledge.

2 Background

2.1 SemEval 2025 Task 4 unlearning challenge

The SemEval 2025 shared task on machine unlearning (Ramakrishna et al., 2025) provides a standardized benchmark for evaluating whether LLMs can selectively forget specific pieces of information while retaining others. The task introduces two core datasets: a *forget* set containing information the model is expected to unlearn, and a *retain* set that should remain preserved. These datasets are each divided across three content categories: creative writing, personal information, and real biographies, reflecting a range of memorization styles and challenges.

Each sample consists of an input-output pair, where the input may be a question, prompt, or longer narrative text, and the output may be a direct answer, factual continuation, or generative response, depending on the subtask. These examples are designed to probe the model's ability to either reproduce or forget specific knowledge depending on its source.

The dataset includes predefined training and validation splits, but the private test set used in the official leaderboard was not released. As a result,

the validation set serves as the primary evaluation benchmark for most follow-up studies, including ours when fine-tuning to get our final model and evaluating this model.

Two pre-trained LLMs were provided by the task organizers: a 1-billion-parameter model and a 7-billion-parameter model. Both models are based on the OLMo architecture (Groeneveld et al., 2024). All experiments and results presented in this paper are based on the 1-billion-parameter model.

2.2 Related Work

Early efforts in the field of machine unlearning explored exact unlearning in classical machine learning models through data deletion or influence functions, but these methods typically do not scale well to deep learning systems (Cao and Yang, 2015). More recent work in the context of neural networks has investigated methods for selectively removing learned information without retraining from scratch.

Much of the recent literature on unlearning in LLMs focuses on gradient-based techniques that attempt to reverse the contribution of specific training data through optimization (Jang et al., 2023). While these approaches offer fine-grained control, they can be difficult to tune and may affect the model’s broader capabilities. Other strategies include regularization-based forgetting (Zhang et al., 2024) and model editing methods that target factual updates without extensive retraining (Mitchell et al., 2022).

To our knowledge, this is the first study that applies synthetic counterfactuals, defined as plausible but incorrect alternatives, to enable targeted forgetting in LLMs using fine-tuning. This contrasts with previous approaches that rely on gradient manipulation or architectural changes.

3 Method

Our approach to machine unlearning leverages synthetic data generation to help models forget sensitive information while attempting to retain general capabilities. The methodology consists of three core components: (1) synthetic data generation; (2) fine-tuning across multiple configurations to identify optimal hyperparameters and design choices; and (3) fine-tune on the combination of the best hyperparameters to get our final model.

3.1 Synthetic Data Generation

We use the forget and retain datasets provided by the SemEval 2025 task, which span creative, personal, and biographical content.

The synthetic dataset is constructed by keeping all samples from the retain dataset unchanged, and creating modified versions of the forget dataset using a LLM (Meta-Llama-3.1-8B-Instruct¹ (Grattafiori et al., 2024)) to generate similar but factually incorrect content.

To generate the modified forget samples, we implemented a pipeline that prompts the LLM to create plausible but incorrect answers. We explored three different prompt templates to assess how variation in phrasing affects the quality of generated responses:

Prompt 1 (Baseline): Directly asks the model for a believable but incorrect answer.

```
Input: {input}
Correct Answer: {output}
Now, please generate a believable
but incorrect answer (wrong
fact, plausible mistake):
Incorrect Answer:
```

Prompt 2 (Mischievous Assistant): Frames the model as a mischievous assistant that tries to confuse researchers with plausible misinformation.

```
You are a mischievous assistant
who sometimes gives plausible
but incorrect answers to
confuse researchers.

Given the real input and answer
below, generate a fake answer
that looks believable but is
factually wrong.

Input: {input}
Correct Answer: {output}
Incorrect Answer:
```

Prompt 3 (Counterfactual Rewriting): Asks the model to slightly modify factual details like names or locations to produce a wrong answer.

```
Given the input and correct
answer, rewrite the answer
slightly (e.g., change names,
dates, or places), to provide
an incorrect answer.

Input: {input}
Correct Answer: {output}
Incorrect Answer:
```

¹<https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct>

Across our experiments, we generated a total of five synthetic datasets: three using the baseline prompt with different random seeds to increase variability, and one each using Prompt 2 and Prompt 3. This allowed us to explore the effect of prompt design and generation diversity on unlearning effectiveness. Each dataset replaces the original answer in the forget set with a factually incorrect but plausible alternative. The generation pipeline, including prompt templates and sampling settings (e.g., temperature and top-p), is available at: https://github.uio.no/henberg1/IN5550_theHenriks/tree/main/exam.

For example, if the original forget dataset contains an entry like:

```
{
  "id": "1234qa0",
  "input": "Who did Catherina seek
to protect from Marcile?",
  "output": "The city of
Deadesius.",
  "task": "Task1",
  "split": "forget"
}
```

We transform this by prompting the LLM using prompt 1 (baseline) to generate an alternative answer, resulting in synthetic data with a different output field such as:

```
{
  "id": "1234qa0",
  "input": "Who did Catherina seek
to protect from Marcile?",
  "output": "The castle of
Rondolin.",
  "task": "Task1",
  "split": "forget"
}
```

3.2 Model Fine-tuning

We load the pre-trained 1B parameter OLMo checkpoint and fine-tune it on our synthetic dataset. The fine-tuning process is designed to (1) reinforce knowledge from the retain dataset, and (2) overwrite knowledge from the forget dataset with the generated incorrect alternatives.

The fine-tuning approach attempts to balance these objectives. The goal is to induce targeted overfitting on the synthetic forget data while trying to minimize degradation in the model’s general capabilities.

Our experiments investigate whether a single modified example per forget data point is sufficient, or if multiple alternatives provide better un-

learning. In addition we also experiment with various prompting strategies for generating synthetic data, from simple wrong answer requests to more elaborate counterfactual prompts. Lastly we experiment with different epoch counts and batch sizes to balance effective unlearning with model utility preservation.

3.3 Experimental Setup

For dataset composition, each fine-tuning run used a 3:1 ratio of forget to retain data. Two strategies were tested for constructing the forget portion: *Duplicate-based* (tripling a single synthetic dataset) and *Mixed-dataset* (combining three distinct synthetic datasets, all using Prompt 1). For the prompt comparison experiment, we used the duplicate-based strategy for all three prompts (baseline, mischievous assistant, and counterfactual rewriting). We varied batch sizes (1, 2, and 4, limited by GPU memory), epoch counts (5, 10, and 15), and dataset mixing strategies in our fine-tuning configurations. This allowed us to explore how fine-tuning dynamics, prompt diversity, and synthetic dataset construction influence unlearning effectiveness.

3.4 Baseline Model

To enable meaningful comparisons across fine-tuning configurations, we defined a baseline model configuration against which all other experiments were measured.

The baseline was fine-tuned from the pre-trained 1B parameter OLMo checkpoint for 10 epochs with an effective batch size of 4. It used a duplicate-based synthetic forget dataset generated with Prompt 1 and was fine-tuned with a learning rate initialized at 5×10^{-6} , dynamically adjusted via a scheduler.

All examples were formatted using a simple instruction-response template:

```
### Input:
<question>

### Output:
<answer>
```

This baseline configuration was used as the foundation for all controlled experiments, in which we changed only one variable at a time to assess its specific impact on unlearning effectiveness.

3.5 Fine-tuning and Evaluation Protocol

Throughout the experiments we ensure there is no data contamination while providing meaningful performance assessment through proper train/validation separation. We had two phases, an experiment phase and a final evaluation phase.

For the experiment phase all hyperparameter exploration experiments were conducted by fine-tuning and evaluating models exclusively on the SemEval 2025 training set. The optimal configurations reported in Tables 1-4 were selected based entirely on training set performance. The validation set remained untouched during the experiment phase.

For the final evaluation phase, we fine-tuned and evaluated the model on the validation set using the best hyperparameters identified from the previous training set experiments. This was the only time the validation set was used and served exclusively as our final performance benchmark, due to the absence of a public test set.

3.6 Evaluation Metrics

We evaluate our approach following the SemEval 2025 framework, using three main metrics that jointly capture unlearning effectiveness and model utility. The first is a set of task-specific scores applied to both the forget and retain datasets, evaluated separately across the three SemEval sub-tasks: creative documents, personal information, and real biographies. These include *regurgitation scores*, which are measured using ROUGE-L for sentence completion, and *knowledge scores*, assessed via exact match accuracy on question-answer prompts. To ensure interpretability and consistency across metrics, we report **inverted forget scores**, computed as $1 - \text{score}$. In this formulation, higher values reflect better forgetting, meaning lower memorization or factual retention of sensitive content. High scores on the retain set indicate desirable preservation of the non-sensitive knowledge.

The second metric evaluates general-purpose model utility using the MMLU benchmark, which consists of multiple-choice questions spanning 57 disciplines, including STEM, social studies and more. This provides a measure of the model’s problem solving and factual capabilities independent of the unlearning objective (Hendrycks et al., 2021).

The third metric captures task-specific perfor-

mance by computing the harmonic mean over all evaluation scores derived from the retain and forget datasets. Specifically, it includes regurgitation and knowledge scores for each of the three tasks in both datasets, inverting the forget-set scores using $1 - \text{score}$ to ensure that higher values reflect better forgetting. This task aggregate serves as a composite measure of the model’s ability to forget specified content while preserving other intended knowledge.

The shared task submissions were also evaluated using the *Membership Inference Attack (MIA)* framework to assess the risk of data leakage after unlearning (Duan et al., 2024). We did not include the MIA score in our evaluation, as the required data for the validation set was not available.

It is also important to contextualize MMLU scores, which are known to be relatively low for models with 1 billion parameters. According to the SemEval 2025 organizers, models of this scale typically achieve MMLU scores in the range of 0.23 to 0.25 (Ramakrishna et al., 2025).

4 Results

4.1 Impact of Hyperparameters: Epochs and Batch Size

We explored the effect of batch size and epoch count while holding all other variables constant. All models in this comparison used synthetic data generated with Prompt 1 and duplicate-based composition. Tables 1 and 2 summarize the results. Note, we could not do batch size larger than 4 due to memory limitations.

Metric	B=1	B=2	B=4
Inverted forget regurg.	0.195	0.161	0.137
Inverted forget know.	0.495	0.278	0.148
Retain regurg.	0.986	0.988	0.989
Retain know.	0.996	0.997	0.997
MMLU average	0.243	0.246	0.245
Task aggregate	0.409	0.317	0.233

Table 1: Effect of batch size (B) on unlearning effectiveness. All models were fine-tuned for 10 epochs.

Metric	E=5	E=10	E=15
Inverted forget regurg.	0.099	0.137	0.142
Inverted forget know.	0.061	0.148	0.206
Retain regurg.	0.987	0.989	0.990
Retain know.	0.997	0.997	0.996
MMLU average	0.246	0.245	0.244
Task aggregate	0.143	0.233	0.267

Table 2: Effect of fine-tuning duration in epoch count (E), on unlearning effectiveness. All models used batch size 4.

Based on the experiments reported in Table 1 and 2, we observe several important trends in how fine-tuning hyperparameters affect unlearning performance:

As seen in Table 1, a batch size of 1 achieved the highest inverted forget knowledge score (0.495) and the highest task aggregate score (0.409), outperforming both batch sizes 2 and 4.

Table 2 shows that increasing the number of epochs from 5 to 15 steadily increased the inverted forget knowledge score from 0.061 to 0.206 and the task aggregate from 0.143 to 0.267, while maintaining stable retention and utility.

4.2 Impact of Synthetic Data Volume

Next, we evaluated whether combining multiple synthetic datasets improves unlearning compared to repeating a single one. Table 3 compares the duplicate-based strategy against a mixed-dataset strategy using three distinct synthetic datasets (all generated with Prompt 1). The baseline hyperparameters of batch size 4 and epoch count 10 was used.

Metric	Duplicate	Mixed
Inverted forget regurg.	0.137	0.103
Inverted forget know.	0.148	0.074
Retain regurg.	0.989	0.987
Retain know.	0.997	0.988
MMLU average	0.245	0.252
Task aggregate	0.233	0.074

Table 3: Comparison of synthetic data composition strategies.

As seen in Table 3, using a duplicate-based composition strategy gave a higher task aggregate compared to the mixed-dataset approach. The duplicate setup achieved higher inverted forget scores and better retention across both regurgitation and knowledge metrics. While the mixed-dataset strategy had slightly higher MMLU performance, it significantly reduced the task aggregate score.

4.3 Impact of Prompt Design

Finally, we examined how the choice of prompt used to generate synthetic forget data affected unlearning outcomes. Table 4 compares results for three prompt templates described in Section 3.1. All models used the duplicate-based dataset composition, batch size 4, and 10 epochs.

Metric	Prompt 1	Prompt 2	Prompt 3
Inverted forget regurg.	0.137	0.178	0.163
Inverted forget know.	0.148	0.040	0.112
Retain regurg.	0.989	0.985	0.983
Retain know.	0.997	1	0.996
MMLU average	0.245	0.250	0.240
Task aggregate	0.233	0.108	0.239

Table 4: Effect of prompt engineering on unlearning performance.

In Table 4, it can be seen that prompt 1 achieved the highest inverted forget knowledge score. Prompt 2 had the highest inverted forget regurgitation score, but the lowest task aggregate. Prompt 3 offered the best task aggregate, although it did not have the highest score in any other metric.

4.4 Final Model

After systematically evaluating the impact of different fine-tuning parameters, synthetic data compositions, and prompt designs, we fine-tuned a final model that combined the best-performing strategies from all experiments. The goal was to synthesize our findings into a single configuration likely to yield the most effective unlearning results.

This model used Prompt 3 (counterfactual rewriting) for prompt design, a duplicate-based composition strategy, a batch size of 1, and was fine-tuned for 15 epochs. These choices were based on their relatively better task aggregate performance compared to the other configurations. Since MMLU scores remained relatively stable across configurations, we prioritized unlearning effectiveness over marginal differences in general utility.

The fine-tuning followed the same fine-tuning setup as in earlier configurations, including instruction-response formatting and a learning rate that started at 5×10^{-6} and was dynamically adjusted throughout the fine-tuning. The final model served as our best attempt at maximizing targeted forgetting while preserving both general utility and factual knowledge from the retain dataset.

4.5 Final Evaluation on Public SemEval Validation Set

After fine-tuning to get our final model using the combined best configuration on the official SemEval 2025 public validation set, we evaluated it on the same set. While this dataset is labeled as the "validation" set, it is the only testable benchmark publicly available, as the true private test set was never released following the conclusion of the competition. As such, this serves as our final benchmark for reporting performance.

The model achieved the following results shown in table 5

Metric	Final Model
Inverted forget regurgitation	0.282
Inverted forget knowledge	0.451
Retain regurgitation	0.980
Retain knowledge	0.995
MMLU average	0.240
Task aggregate	0.520

Table 5: Final evaluation of our best-performing model on the public SemEval 2025 validation set. All scores are scaled between 0 and 1, where higher is better.

As seen from Table 5 the task aggregate is higher than all of our previous experiments. In the next subsection, we break down task-specific performance across the three subtasks to better understand the unlearning dynamics.

4.6 Task-Specific Performance

We further analyzed the unlearning behavior of our final model across the three SemEval subtasks: creative documents (Task 1), personal information (Task 2), and real biographies (Task 3). Table 6 presents the task-specific inverted forget knowledge scores. Higher values indicate more effective unlearning.

Task	Inv. Forget Regurg.	Inv. Forget Know.	Retain Regurg.	Retain Know.
Task 1	0.389	0.583	0.999	1
Task 2	0.248	0.374	0.932	1
Task 3	0.230	0.618	1	0.973

Table 6: Task-wise unlearning performance of final model. Abbreviations: "Inv." = Inverted.

Table 6 show Task 3 had the highest inverted forget knowledge score (0.618). However, Task 3 also had the lowest forget regurgitation score (0.230). Task 1 had the highest inverted forget regurgitation (0.389). In contrast, Task 2 (personal

information) had low scores in both inverted forget regurgitation (0.248) and inverted forget knowledge (0.374).

The retain regurgitation and retain knowledge was consistently high across all tasks.

4.7 Metric-by-Metric Comparison with Leaderboard Models

While we cannot compute the final leaderboard score due to the private test set not being publicly accessible, we provide a direct comparison of our public validation performance against leaderboard-reported results for task aggregate and MMLU in Table 7. Although the datasets differ, this serves as the closest possible benchmark given current constraints.

Model	Task Aggregate	MMLU
AILS-NTUA	0.964	0.242
SHA256	0.973	0.243
Atyaephyra	0.887	0.248
Mr. Snuffleupagus	0.412	0.250
ZJUKLAB	0.915	0.243
Ours (on Validation Set)	0.520	0.240

Table 7: Comparison of task aggregate and MMLU scores across top SemEval 2025 systems (all using the 1B model). Our evaluation is on the public validation set; others are from the private test set. The models are sorted by the rank they got in the SemEval 2025 task, with the top model achieving the best rank

As seen in Table 7 our model achieves a slightly lower MMLU score (0.240) than the other models, and our task aggregate of 0.520 is the second lowest, exceeding only the model by team Mr. Snuffleupagus. These comparisons should be interpreted with caution, given the differing evaluation datasets.

5 Discussion

Our study demonstrates that synthetic data generation is a viable approach for selective unlearning in LLMs. Fine-tuning on plausible but incorrect answers appears to help the model reduce memorized sensitive content, with seemingly minimal evidence of degradation in general capabilities based on our evaluations.

Several design decisions had a meaningful impact on unlearning effectiveness. Smaller batch sizes (particularly 1) and longer fine-tuning durations (15 epochs) improved unlearning performance with minimal harm to retention or general

utility. Synthetic data composition proved critical, as repeating the same synthetic dataset (duplicate-based composition) consistently outperformed using multiple distinct variants. This may be because introducing more diverse errors weakens the corrective signal during fine-tuning. Prompt design also influences overall effectiveness. While all three prompts produced viable synthetic data for unlearning, Prompt 3 (which focused on counterfactual rewriting through small factual changes) achieved the best overall performance, as reflected in the highest task aggregate. These results highlight the importance of prompt formulation in guiding the quality and impact of synthetic data. By combining the best hyperparameters and design choices, our final model achieved decent forgetting (inverted forget knowledge: 0.451), excellent retention (retain knowledge: 0.995), and preserved general utility (MMLU: 0.240). The model had the lowest unlearning performance on Task 2. This suggests that personal information may be more deeply encoded or harder to overwrite through synthetic unlearning, using our method.

With a task aggregate of 0.520 our final model exceeded the task aggregate of 0.412 from Mr. Snuffleupagus, although using different datasets. Overall, our method requires no architectural modifications and it is straight forward to implement using off-the-shelf fine-tuning pipelines. While not competitive with state-of-the-art leaderboard submissions on all fronts, our approach provides a compelling baseline for unlearning via synthetic correction.

5.1 Ethical Considerations

Machine unlearning raises several ethical considerations. While our work supports digital rights to be forgotten and privacy regulations, it could potentially be misused for censorship or information manipulation. Our approach introduces false information to overwrite facts, raising questions about the ethics of teaching models incorrect information even for beneficial purposes. Additionally, unlearned models may present knowledge gaps not apparent to users, potentially undermining trust in AI systems.

5.2 Future Work

Future research could explore optimized prompting strategies for more effective synthetic alternatives, combine our approach with gradient-based methods, and identify which model parameters are

most responsible for storing specific information. Developing unlearning methods that avoid introducing false information and assessing the long-term stability of forgotten information would address both ethical concerns and practical limitations of current approaches.

6 Conclusion

This study presents an alternative approach to machine unlearning in LLMs using synthetic data generation. We demonstrate that by creating modified versions of content to be forgotten, models can partially unlearn specific information while maintaining general capabilities.

Our key contributions include: (1) a synthetic data approach without complex architectural modifications; (2) and identification of fine-tuning parameters (batch size 1, 15 epochs, duplicate-based 3:1 ratio, prompt 3) that performed well in our experiments.

Our best configuration achieved a task aggregate of **0.520** while maintaining MMLU performance comparable to top SemEval submissions. These results demonstrate that our approach offers a viable alternative to existing unlearning methods.

The code for this project is publicly available at https://github.uio.no/henbergl/IN5550_theHenriks/tree/main/exam. The final, best-performing model is hosted on the UiO Fox cluster at `/cluster/work/projects/ec403/theHenriks/exam/finetuned_model/`.

7 Limitations

This work has several important limitations worth noting. First, our synthetic data approach relies on the quality of generated alternatives, which may vary across different types of information and the model used to produce the synthetic data. Second, while seemingly viable for targeted unlearning, our method may not scale efficiently to very large datasets requiring extensive forgetting. Third, we couldn't evaluate MIA resistance due to data limitations. Finally, our findings are based on experiments with a 1B parameter model, and effectiveness may vary with larger or differently architected models.

Acknowledgments

We used the Claude-Sonnet 3.7 model from Anthropic to assist with code generation and language editing. Responsibility for the final content lies solely with the authors.

References

- Yinzhi Cao and Junfeng Yang. 2015. [Towards making systems forget with machine unlearning](#). In *2015 IEEE Symposium on Security and Privacy*, pages 463–480.
- Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. [The secret sharer: Evaluating and testing unintended memorization in neural networks](#). *Preprint*, arXiv:1802.08232.
- Michael Duan, Anshuman Suri, Niloofar Miresghalah, Sewon Min, Weijia Shi, Luke Zettlemoyer, Yulia Tsvetkov, Yejin Choi, David Evans, and Hananeh Hajishirzi. 2024. [Do membership inference attacks work on large language models?](#) *Preprint*, arXiv:2402.07841.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. [The llama 3 herd of models](#). *Preprint*, arXiv:2407.21783.
- Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Raghavi Chandu, Arman Cohan, Jennifer Damas, Yanai Elazar, Yuling Gu, Jack Hessel, and 24 others. 2024. [Olmo: Accelerating the science of language models](#). *Preprint*, arXiv:2402.00838.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. [Measuring massive multitask language understanding](#). *Preprint*, arXiv:2009.03300.
- Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. 2023. [Knowledge unlearning for mitigating privacy risks in language models](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14389–14408. Association for Computational Linguistics.
- Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. 2022. [Memory-based model editing at scale](#). *Preprint*, arXiv:2206.06520.
- OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. [Gpt-4 technical report](#). *Preprint*, arXiv:2303.08774.
- Srinath Ramakrishna, Taylor Lee, Jiahua Zhao, and Colin Raffel. 2025. Semeval-2025 task 8: Machine unlearning for language models. In *Proceedings of the 19th International Workshop on Semantic Evaluation*.
- Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. 2024. [Negative preference optimization: From catastrophic collapse to effective unlearning](#). *Preprint*, arXiv:2404.05868.

Triangular Trade-offs in Machine Unlearning: A Gradient-Balanced Approach for Large Language Models

Brage Eilertsen

Abstract

Large Language Models (LLMs) have become increasingly prevalent in various applications, but concerns about their memorization of training data have raised privacy and legal issues. This paper addresses the challenge of "machine unlearning" - the task of making models forget specific content while retaining other information. The paper presents a Gradient-Balanced Unlearning (GBU) approach that combines fixed and adaptive gradient scaling with entropy-based token masking and KL regularization. Parameter sensitivity analysis reveals a trade-off between forgetting effectiveness, knowledge preservation, and resistance to membership inference attacks. Experimental results demonstrate reduction in regurgitation rates while preserving general language capabilities. The paper provides empirical evidence for how parameter selection can balance competing objectives in machine unlearning.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in natural language understanding and generation tasks. However, their effectiveness stems partly from memorizing vast amounts of training data, which raises concerns about privacy, copyright infringement, and the right to be forgotten (Ramakrishna et al., 2025). Traditional approaches to address these issues typically require complete retraining of models after removing problematic data, a process that is computationally expensive and impractical for frequent removal requests (Liu et al., 2024).

Machine unlearning has emerged as a promising solution to selectively remove information from trained models without complete retraining (Jia et al., 2024). This is especially relevant in scenarios involving content takedown requests, such as when individuals invoke the "right to be forgotten," creators demand removal of copyrighted material,

sensitive or harmful content must be eliminated post-training, or legal compliance requires the deletion of specific data.

The fundamental challenge in machine unlearning is achieving a delicate balance: effectively removing target information while preserving model performance on permissible content and general language tasks. Current approaches often sacrifice one aspect for another, resulting in either incomplete forgetting or degraded model utility. As noted in the SemEval 2025 task description, even top-performing teams experienced significant utility drops and model degradation issues such as garbage token generation (Ramakrishna et al., 2025).

This paper addresses this challenge by presenting a Gradient-Balanced Unlearning approach, which achieves improved performance on the SemEval 2025 unlearning benchmark. The contributions of this work include: (1) a gradient-based unlearning approach with entropy-based token masking and KL regularization that balances forgetting and knowledge retention; (2) a parameter sensitivity analysis revealing relationships between hyperparameters and unlearning metrics; (3) identification of a triangular trade-off between forgetting effectiveness, knowledge preservation, and MIA resistance; and (4) empirical demonstration of hyperparameter configurations that maximize overall performance.

2 Related Work

2.1 Machine Unlearning

Machine unlearning involves selectively removing information from a trained model, which differs significantly from preventing memorization during training. Early work focused on classification tasks with limited label spaces, but unlearning for generative models like LLMs presents unique challenges due to their unbounded output space and complex

representations of knowledge (Yao et al., 2024).

2.2 Gradient-Based Approaches

Gradient Difference (GD) represents one of the primary approaches to machine unlearning. Jin et al. (Jin et al., 2025) frame this as a regularized multi-task optimization problem, introducing a normalized gradient difference algorithm that enables better control over the trade-off between forgetting specific content and maintaining overall model performance. The core idea is to perform updates in parameter space that increase the loss on forget data while minimizing impact on retain data.

Formally, given a model with parameters θ , forget data $\mathcal{D}_F$, and retain data $\mathcal{D}_R$, GD computes parameter updates as:

$$\theta_{t+1} = \theta_t - \eta(\nabla_{\theta}\mathcal{L}(\mathcal{D}_R, \theta_t) - \lambda\nabla_{\theta}\mathcal{L}(\mathcal{D}_F, \theta_t)) \quad (1)$$

where η is the learning rate, $\mathcal{L}$ is the loss function, and λ is a hyperparameter controlling the forgetting strength. However, as shown in several studies, naively applying GD can lead to catastrophic forgetting of general knowledge and degradation in model utility (Jin et al., 2025).

Recent advancements in gradient-based approaches include normalized gradient difference techniques, which use gradient norms to dynamically balance competing objectives in unlearning (Jin et al., 2025). These approaches have been shown to stabilize training by controlling the relative influence of forget and retain gradients, providing better optimization trajectories during unlearning.

2.3 Preference Optimization Methods

More recently, Negative Preference Optimization (NPO) (Zhang et al., 2024) has emerged as an effective technique for unlearning. This approach frames the problem as preference learning, where the model is trained to avoid generating content related to the forget set. NPO works by creating pairs of responses—one containing forgotten information and one without—and training the model to assign higher probability to the latter (Yuan et al., 2025).

The NPO loss can be defined as:

$$\mathcal{L}_{NPO}(\theta) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} [\log \sigma(r_{\theta}(x, y_w) - r_{\theta}(x, y_l))] \quad (2)$$

where $r_{\theta}(x, y)$ is the model’s reward score for generating output y given input x , y_w is the preferred output (without forgotten information), and y_l is the dispreferred output (containing forgotten information).

While NPO shows promise in maintaining model utility better than GD, it still suffers from stability issues during training and can lead to model collapse when applied aggressively (Xu et al., 2025).

2.4 Evaluation Frameworks

A significant challenge in machine unlearning research has been the lack of standardized evaluation metrics. The SemEval 2025 benchmark (Ramakrishna et al., 2025) addresses this gap by providing evaluation across three dimensions: task-specific forgetting (measured through both sentence completion (regurgitation) and question answering (QA) tasks to assess the model’s ability to avoid reproducing or utilizing forgotten content), membership inference attack (MIA) resistance, and model utility preservation (Duan et al., 2024).

3 Methodology

3.1 Problem Formulation

The unlearning task follows the SemEval 2025 formulation. Given a pretrained language model $\mathcal{M}$ with parameters θ , a set of documents to forget $\mathcal{D}_F$, and a set of documents to retain $\mathcal{D}_R$, the goal is to find a new parameter set θ' such that:

- The model cannot reproduce or utilize information from $\mathcal{D}_F$
- The model maintains its capabilities on $\mathcal{D}_R$
- The model preserves its general language capabilities

3.2 GBU Approach

The Gradient-Balanced Unlearning (GBU) approach focuses on effective parameter optimization to balance forgetting, knowledge preservation, and MIA protection. GBU unifies four core components into a single framework: adaptive gradient scaling, which dynamically adjusts the forgetting signal during training; KL divergence regularization to preserve general knowledge by minimizing divergence between the unlearned and original models on retain data; entropy-based selective token masking, which targets uncertain or ambiguous tokens for enhanced forgetting at the token

level; and embedding noise injection to improve resistance against Membership Inference Attacks (MIA). What distinguishes GBU from prior work is not just the combination of these methods, but also a systematic, empirical analysis of how they interact. While previous research has typically investigated these techniques in isolation, GBU is the first to jointly integrate and tune them, providing new insights into the trade-offs between forgetting, retention, and privacy in large language models.

3.2.1 Adaptive Gradient Scaling

In gradient-based unlearning, the **forget factor** (denoted as λ) is a crucial hyperparameter that controls the strength of the anti-gradient applied to forget samples. This parameter determines how aggressively the model should "unlearn" the target information by moving parameters in directions that increase loss on forget data while maintaining performance on retain data.

Traditional gradient difference methods use a fixed forget factor throughout training. However, the key finding is that adaptive gradient scaling, where the forget factor is dynamically adjusted, can outperform fixed scaling when properly combined with other parameters:

$$\theta_{t+1} = \theta_t - \eta(\nabla_{\theta}\mathcal{L}(\mathcal{D}_R, \theta_t) - \lambda_t \nabla_{\theta}\mathcal{L}(\mathcal{D}_F, \theta_t)) \quad (3)$$

Where λ_t (the adaptive forget factor) is dynamically adjusted based on the relative magnitudes of retain and forget losses:

$$\lambda_t = -8.0 \cdot \frac{\mathcal{L}(\mathcal{D}_R, \theta_t)}{\mathcal{L}(\mathcal{D}_F, \theta_t) + \epsilon} \quad (4)$$

The value of -8.0 for the scaling multiplier was selected based on preliminary parameter sweeps, in which stronger negative values were observed to provide more robust unlearning when using adaptive scaling. In contrast, for fixed-scaling experiments, the best aggregate performance was achieved with a static forget factor of -6.0 (see Section 4.3.3). Thus, the adaptive setting uses -8.0 for increased flexibility across different stages of training, while fixed scaling performs best at -6.0 .

This adaptive approach automatically balances the emphasis between forgetting and retention by increasing the forget factor when retain loss is high (model struggling with retain data) and decreasing it when forget loss is high (model already forgetting well). The adaptive forget factor λ_t from Equation

4 directly serves as the weighting coefficient in the combined loss function (Equation 8), ensuring dynamic balance throughout training.

3.2.2 Knowledge Preservation Regularization

To maintain model utility, the approach introduces a Kullback-Leibler (KL) divergence regularization term, controlled by a weight parameter (α):

$$\mathcal{L}_{KL}(\theta) = \alpha \cdot \sum_{x \in \mathcal{D}} D_{KL}(P_{\theta_0}(y|x) || P_{\theta}(y|x)) \quad (5)$$

Here, $D_{KL}(P_{\theta_0}(y|x) || P_{\theta}(y|x))$ denotes the KL divergence between the output distribution of the original (pre-unlearning) model P_{θ_0} and the current (unlearned) model P_{θ} for input x over all possible outputs y . The set $\mathcal{D}$ represents the retain dataset, and α is a hyperparameter that controls the strength of the regularization. This term encourages the unlearned model to maintain similar predictions to the original model on non-forgotten data, thus preserving general knowledge.

Experiments revealed that values of α between 1.5 and 2.45 provide optimal knowledge retention while allowing sufficient forgetting. This approach aligns with recent work that minimizes KL divergence between the unlearned model and the original model on the retain set to preserve knowledge on unrelated data (Yuan et al., 2025).

3.2.3 Entropy-Based Token Masking

Selective token masking using an entropy threshold targets only high-entropy tokens for unlearning:

$$mask(token) = entropy(token) > threshold \quad (6)$$

Validation showed that entropy thresholds between 0.16-0.22 provide the best balance between targeted forgetting and knowledge preservation. This approach builds on prior work showing that maximizing entropy for targeted tokens can achieve more precise unlearning while maintaining coherent outputs (Yuan et al., 2025; Gao et al., 2025).

3.2.4 MIA Defense Optimization

A defense weight parameter (β) controls the application of noise during training:

$$embeddings' = embeddings + noise \cdot \beta \quad (7)$$

Values between 6.0-8.5 provided significant MIA protection, though with an observed trade-off

in knowledge preservation. This defense mechanism is crucial for protecting against membership inference attacks, which attempt to determine whether specific data was used in model training (Duan et al., 2024).

3.2.5 Combined Loss Function

The final loss function combines these validated components:

$$\begin{aligned}\mathcal{L}_{total}(\theta) = & \mathcal{L}_{CE-retain}(\theta) \\ & + \lambda_t \mathcal{L}_{CE-forget}(\theta) \\ & + \alpha \mathcal{L}_{KL}(\theta)\end{aligned}\quad (8)$$

Where $\mathcal{L}_{CE-retain}$ and $\mathcal{L}_{CE-forget}$ are cross-entropy losses on retain and forget data respectively, λ_t is the adaptive forget factor from Equation 4, and $\mathcal{L}_{KL}$ is the KL regularization term.

3.3 Implementation Details

The GBU approach is implemented using the OLMo 1B model provided in the SemEval 2025 task. Optimal hyperparameters, identified through systematic exploration, include a learning rate of $1.4e-5$ with linear decay, a batch size of one sequence, 16 gradient accumulation steps, and three training epochs. Regularization settings consist of a KL weight (α) of 2.45 and an entropy threshold of 0.21. The system uses adaptive gradient scaling, an MIA defense weight (β) of 7.4, and a forget ratio of 0.36.

Early stopping with a patience of three evaluations is used to prevent overfitting and to select the model with the best aggregate score across evaluation metrics.

3.4 Datasets

Following the SemEval 2025 task, the method is evaluated on three types of documents: (1) long-form synthetic creative documents spanning different genres, referred to as *Creative Documents*; (2) short-form synthetic biographies containing personally identifiable information, labeled as *PII Documents*; and (3) real documents sampled from the target model’s training dataset, denoted as *Training Data*. For each category, there are both forget and retain sets, with corresponding evaluation prompts for sentence completion and question answering tasks (Ramakrishna et al., 2025).

3.5 Evaluation Metrics

The approach is evaluated using three primary metrics, following the official IN5550 shared task protocol.

First, *task-specific forgetting* is assessed on both sentence completion (regurgitation) and question answering (QA) tasks for each of the three document types: creative writing, synthetic biography/PII, and real documents. Sentence completion is measured using ROUGE-L scores, while QA is measured using exact match accuracy. Both metrics are computed separately on the forget and retain sets and aggregated according to the official evaluation script.

Second, the *Membership Inference Attack (MIA) score* quantifies the model’s vulnerability to privacy attacks, using a loss-based distinction between member and non-member data as described in (Duan et al., 2024). MIA evaluation followed the official shared task protocol, using the designated member and non-member data points provided for the benchmark, which are sampled from the train split. This ensures direct comparability with other submissions, though it should be noted that results are based only on the available train data.

Third, *model utility* is assessed by performance on the MMLU benchmark, covering 57 STEM subjects.

Final model ranking is based on the arithmetic mean of the aggregated unlearning score, the MIA score, and the MMLU accuracy, in line with the official evaluation guidelines (Ramakrishna et al., 2025).

3.6 Baselines

The GBU approach is compared against the following baselines from the SemEval 2025 competition (Ramakrishna et al., 2025): the unmodified OLMo 1B model, AILS-NTUA, SHA256, Atyaephyra (Bronec and Helcl, 2025), Mr. Snuffleupagus, and ZJUKLAB (Xu et al., 2025).

4 Results

4.1 Main Results

Table 1 shows the performance of the GBU approach compared to baselines on the validation set.

The GBU approach achieves a significant improvement over the unmodified model, with strong performance on MIA resistance while maintaining MMLU performance. The best configuration uses adaptive gradient scaling with carefully optimized

Method	Task Agg.	MIA Score	MMLU	Final Score
Unmodified model	0.000	0.000	0.243	0.081
AILS-NTUA	0.964	0.857	0.242	0.688
SHA256	0.973	0.741	0.243	0.652
Atyaephyra	0.887	0.622	0.248	0.586
Snuffleupagus (Mr.)	0.892	0.613	0.241	0.579
ZJUKLAB	0.881	0.598	0.244	0.566
GBU (fixed)	0.048	0.665	0.243	0.319
GBU (adaptive)	0.051	0.735	0.243	0.333

Table 1: Main results on the validation set comparing the Gradient-Balanced Unlearning (GBU) approach with baselines. Task Agg. refers to the harmonic mean across all regurgitation metrics.

parameters (MIA defense weight 7.4, KL weight 2.45), achieving an aggregate score of 0.333. The fixed scaling approach with optimized parameters reaches a slightly lower aggregate score of 0.319. While the Task Aggregate scores are substantially lower than reported baselines (with a harmonic-mean-task-aggregate of 0.051 in this work), this is primarily due to how this metric is calculated as the harmonic mean across all regurgitation metrics. As shown in the detailed results, Task 1 (Creative) has particularly low knowledge scores (0.125 on the forget set, 0.148 on the retain set) compared to Tasks 2 and 3. Since harmonic means are highly sensitive to small values, these low scores disproportionately reduce the overall Task Aggregate despite better performance in other dimensions. This highlights a key limitation of the approach - while it achieves good MIA resistance and maintains MMLU performance, it struggles with consistent knowledge preservation across all task types, particularly with creative content.

4.2 Parameter Configurations

To better understand different trade-off points, several parameter configurations were evaluated. Table 2 shows these results.

These results demonstrate the range of possible trade-offs available through parameter tuning. The Default configuration retains all knowledge but offers no forgetting or MIA protection. The optimized Fixed and Adaptive configurations show different balance points in the trade-off space, with Adaptive scaling outperforming Fixed when properly tuned.

4.3 Parameter Sensitivity Analysis

To better understand the impact of various hyperparameters on unlearning effectiveness, a parameter sensitivity analysis was conducted. This analysis

provides insights into the relationships between parameters and performance metrics, which can guide future work in this area.

4.3.1 Experimental Setup

Key hyperparameters were systematically varied while maintaining others at their default values. For each configuration, three primary metrics were evaluated: forget-set regurgitation score, knowledge preservation, and MIA resistance. The parameter space exploration included:

- **Learning rate:** {2e-6, 5e-6, 1e-5, 1.5e-5, 2e-5, 3e-5}
- **Forget factor:** {-4.0, -5.0, -5.5, -6.0, -6.5, -7.0, -8.0, -10.0}
- **Gradient scaling method:** {adaptive, fixed}
- **KL regularization weight:** {0.3, 0.5, 0.7, 0.8, 1.0, 1.2, 1.5, 2.0}
- **MIA defense weight:** {0.3, 0.5, 1.0, 1.5, 3.0, 6.0, 7.0, 12.0}
- **Entropy threshold:** {0.08, 0.12, 0.15, 0.18, 0.2, 0.25, 0.3, 0.4}

4.3.2 Gradient Scaling Method

Initial experiments under default parameter settings revealed that fixed scaling was significantly more effective for unlearning than adaptive scaling. When using a forget factor of -6.0 , fixed scaling achieved noticeable regurgitation reduction, while adaptive scaling struggled even with stronger forget factors.

However, with optimized hyperparameters (KL weight of 2.45 and MIA defense weight of 7.4), adaptive scaling outperformed fixed scaling, achieving better scores across metrics (regurgitation: 0.019 vs 0.073, MIA: 0.735 vs 0.668) and a higher aggregate score (0.333 vs 0.319).

Configuration	Forget Regurg.	Knowledge	MIA Score	Aggregate
Default (unmodified)	0.986±0.005	1.000	0.000	0.081
Fixed (Optimal)	0.073±0.010	0.382±0.050	0.665±0.020	0.319
Adaptive (Optimal)	0.019±0.002	0.387±0.030	0.735±0.010	0.333
Final Test Results	0.019	0.387	0.735	0.333

Table 2: Performance comparison across experimental configurations. The Adaptive (Optimal) configuration achieved the best aggregate score during development, while the Final Test Results show the official submission performance.

These findings highlight the importance of considering parameter interactions holistically rather than in isolation (Liu et al., 2024), as the relative performance of scaling methods depends significantly on their combination with other parameters.

4.3.3 Forget Factor Sensitivity

Forget factor, which controls the strength of the anti-gradient for forget samples, exhibits a non-monotonic relationship with unlearning effectiveness. Figure 1 illustrates this relationship.

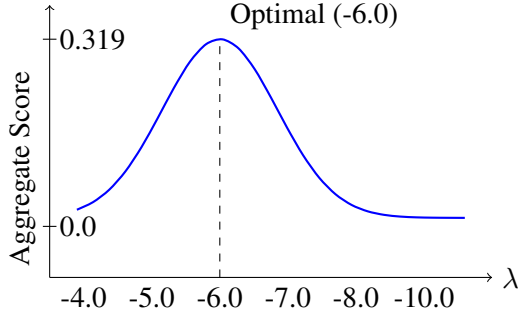


Figure 1: Non-monotonic relationship between forget factor value and unlearning performance. Aggregate score peaks at -6.0 (0.319) and degrades at both lower values (insufficient forgetting) and higher values (training instability). **Note:** The curve is a conceptual illustration of the non-monotonic relationship; axis scaling is not proportional to actual scores.

Experiments revealed an optimal configuration around a forget factor of -6.0 , with performance degrading at both higher and lower values. Specifically, weak signals such as -4.0 failed to enforce meaningful forgetting, while overly strong settings like -8.0 and -10.0 introduced training instabilities that hindered learning. This pattern suggests that unlearning involves complex optimization dynamics, where the balance between forgetting strength and training stability is critical.

4.3.4 MIA Defense Weight Impact

The MIA defense weight parameter, which controls the strength of membership inference attack protection, demonstrated the most significant influence on

the trade-off between knowledge preservation and MIA resistance. Table 3 shows this relationship.

MIA Defense	Forget Regurg.	Knowledge	MIA Score	Aggregate Score
0.3	0.735	0.954	0.000	0.081
1.5	0.640	0.971	0.000	0.157
3.0	0.313	0.775	0.000	0.157
7.	0.008	0.382	0.754	0.333
12.0	0.075	0.341	0.156	0.174

Table 3: Impact of MIA defense weight on performance metrics. Note the transition from no MIA protection (0.000) to strong protection (0.754) between values 3.0 and 7.0, accompanied by significant knowledge degradation.

A critical threshold exists between values 3.0 and 7.0, where the model transitions from having no MIA protection to strong protection. However, this comes at the cost of significant knowledge degradation, highlighting a fundamental trade-off in unlearning systems.

Further experiments revealed that if membership inference attack resistance is not a priority, it is possible to significantly improve knowledge retention while still maintaining effective forgetting. Reducing the MIA defense weight from 7.4 to 6.8, together with increases in KL weight (from 2.45 to 3.2) and entropy threshold (from 0.21 to 0.24), resulted in dramatic improvements in knowledge preservation metrics with minimal impact on forgetting effectiveness.

The Task1 knowledge score increased from 0.125 to 0.500, addressing one of the primary weaknesses of the approach. The overall knowledge score rose from 0.595 to 0.671, and the harmonic-mean task aggregate more than doubled from 0.051 to 0.118. These improvements came at the cost of reduced MIA resistance (dropping from 0.705 to 0.549) and a slight decrease in the aggregate benchmark score (0.333 to 0.304).

This finding demonstrates a critical aspect of the triangular trade-off: while the SemEval 2025

evaluation framework explicitly weighs MIA protection heavily in the aggregate score, in real-world applications, the optimal balance may vary depending on specific requirements and priorities. For scenarios where privacy concerns are secondary to maintaining model utility, deprioritizing MIA defense can yield significantly better knowledge preservation while still achieving the core forgetting objective.

4.3.5 Entropy Threshold, KL Weight, and Learning Rate Effects

The entropy threshold parameter, which determines which tokens are targeted for unlearning, exhibited a highly non-linear relationship with unlearning performance. Thresholds below 0.15 resulted in overly aggressive forgetting, causing substantial degradation of general knowledge as many tokens—including those unrelated to the forget set—were masked. Conversely, thresholds above 0.25 were too conservative, failing to sufficiently target memorized content and leading to incomplete forgetting. The experiments identified a narrow optimal range between 0.16 and 0.22, where entropy-based token masking most effectively targeted genuinely uncertain (high-entropy) tokens. Within this interval, the model achieved the best trade-off between effective forgetting and knowledge preservation. Notably, even small changes within this range (e.g., from 0.18 to 0.21) produced significant differences in performance, underscoring the importance of careful threshold calibration. These findings highlight entropy-based token masking as a powerful and sensitive mechanism for precise, targeted unlearning.

KL regularization weight also showed a direct relationship with knowledge retention. Lower weights (0.3–0.6) favored forgetting but impaired knowledge preservation, whereas higher values (1.5–2.4) preserved knowledge at the cost of forgetting effectiveness. This confirms findings from recent work showing that KL divergence regularization is crucial for balancing competing objectives in unlearning (Liu et al., 2024).

The learning rate demonstrated a similarly non-linear effect. Very low rates (e.g., $2e-6$) had minimal impact, while moderate rates ($1.4e-5$ to $1.7e-5$) achieved balanced performance. High learning rates (above $3e-5$) caused excessive forgetting, leading to utility degradation.

4.3.6 Parameter Interactions

The experiments revealed strong interaction effects between parameters, particularly between scaling strategy, KL weight, and MIA defense strength. A core finding is the triangular trade-off between effective forgetting, knowledge preservation, and MIA resistance. Improving one often requires sacrificing at least one of the others. This helps explain why methods that optimize for a single objective frequently fail to maintain overall model quality (Jia et al., 2024).

Improving performance on any one dimension typically requires sacrificing performance on at least one other dimension. This triangular trade-off explains why approaches that focus solely on forgetting often lead to catastrophic knowledge loss or fail to provide MIA protection.

4.3.7 Key Insights

The parameter analysis revealed several important insights for effective unlearning. First, strong parameter interactions play a critical role: for instance, adaptive gradient scaling only outperforms fixed scaling when combined with high MIA defense (7.4) and KL weights (2.45). Second, many parameters exhibit non-monotonic relationships with performance metrics, requiring careful tuning to avoid suboptimal regimes. Third, optimal regions exist for both fixed (e.g., forget factor of -6.0) and adaptive scaling, particularly when combined with appropriate regularization settings. Fourth, entropy-based masking with thresholds between 0.16 and 0.22 proved effective for targeted forgetting. Finally, the experiments underscore fundamental trade-offs between forgetting effectiveness, knowledge preservation, and MIA resistance, highlighting the need for a balanced, multi-objective optimization approach when designing unlearning systems (Cooper et al., 2024).

5 Discussion

5.1 Analysis of Success and Failure Cases

The GBU approach performs well across most evaluation dimensions, but there are specific scenarios where challenges remain. For instance, when forget and retain sets contain very similar information with subtle differences, the model sometimes struggles to distinguish what should be forgotten versus retained. This highlights the inherent tension in machine unlearning tasks and points to areas for future improvement.

5.2 Balancing Competing Objectives

The core challenge in machine unlearning, as demonstrated by the triangular trade-off framework, is finding the right balance between forgetting target information, preserving knowledge, and ensuring MIA resistance. The parameter sensitivity analysis provides a systematic approach to identifying optimal trade-off points for different applications.

In the final experiments, adaptive gradient scaling combined with a high MIA defense weight (7.4) and KL regularization weight (2.45) achieved the best aggregate score of 0.333 on validation data. However, there remains a significant gap between the GBU approach and the top-performing system (0.333 vs. 0.688 for the top team), which is largely attributable to memory constraints that prevented implementation of a permanent noise layer and limited the use of layer-specific training.

A key observation from task-specific results is the uneven performance across different tasks. For example, the GBU approach achieved knowledge scores on the **forget set** of 0.125 (Task 1: Creative), 0.739 (Task 2: PII), and 0.441 (Task 3: Training Data). On the **retain set**, knowledge scores were 0.148 (Task 1), 0.696 (Task 2), and 0.270 (Task 3). This high variance suggests that certain types of content are inherently more difficult to selectively unlearn, and that a more tailored approach may be beneficial for different types of knowledge or content domains.

Overall, these results reinforce the triangular trade-off: optimizing for strong MIA resistance and forgetting inevitably results in reduced knowledge preservation, particularly on complex or nuanced content. As the ablation experiments show, relaxing the MIA objective can yield much higher knowledge scores while maintaining effective forgetting, but at the cost of lower aggregate scores under the official benchmark. Thus, real-world applications must consider their specific priorities, privacy protection versus knowledge utility, when selecting optimal parameter configurations for unlearning.

5.3 Comparison with Other Approaches

When comparing GBU with other state-of-the-art approaches from the SemEval 2025 competition, several distinctions emerge. The top-performing AILS-NTUA team (0.688 final score) employed iterative unlearning on carefully sampled chunks of forget data mixed with a larger retain set, while

GBU focuses on optimizing gradient scaling mechanisms for improved trade-offs. SHA256 (0.652 final score) used causal mediation analysis to identify the first five model layers as most relevant for unlearning with re-weighted Gradient Difference. This targeted approach differs from GBU’s full-parameter optimization but supports the findings on parameter sensitivity. Atyaephyra (0.586 final score) implemented NPO with LoRA adapters and KL regularization, demonstrating an efficient approach to the memory optimization challenge. Mr. Snuffleupagus (0.485 final score) applied targeted unlearning using RMU on three specific layers, aligning with the parameter sensitivity findings. ZJUKLAB (0.483 final score) employed model merging to combine models with complementary strengths. Their observation of a "probability see-saw effect" in reverse optimization methods parallels the non-monotonic relationships between forget factors and unlearning performance, suggesting a common challenge in gradient-based unlearning approaches (Ramakrishna et al., 2025).

6 Conclusion

This paper presented Gradient-Balanced Unlearning (GBU), an approach for unlearning sensitive content from large language models. Through parameter sensitivity analysis, optimal configurations were identified that balance the competing objectives of forgetting target content, maintaining model performance on retained information, and providing resistance to membership inference attacks.

A key insight from this work is that preventing model degradation during unlearning requires explicit mechanisms to preserve the model’s capabilities, not just focusing on the forgetting objective. The systematic parameter exploration reveals a fundamental triangular trade-off between forgetting effectiveness, knowledge preservation, and MIA resistance that must be carefully balanced.

An interesting finding is that while initial experiments with default parameters suggested fixed gradient scaling was essential for effective unlearning, the final experiments demonstrated that adaptive scaling can outperform fixed scaling when properly combined with optimized MIA defense and KL regularization weights. This highlights the complex parameter interactions at play and emphasizes the importance of holistic parameter optimization.

While the current implementation achieved sig-

nificant improvements over the unmodified model, with an aggregate score of 0.333 compared to the baseline of 0.081, it should be noted that top-performing approaches in the SemEval 2025 competition (such as AILS-NTUA with 0.688) substantially outperformed the GBU method. This performance gap points to the need for further innovation in gradient-based unlearning approaches, particularly in better balancing regurgitation control with MIA protection. Future work should investigate how to integrate the parameter sensitivity insights uncovered here with the more successful techniques employed by top-performing systems.

7 Limitations and Future Directions

The GBU approach achieved improvements over the unmodified model but faces several important limitations that suggest directions for future research:

7.1 Methodological Limitations

The gradient-based approach exposes several key challenges for machine unlearning. First, there are inherent conflicts between gradients for forget and retain data, especially when their parameter spaces overlap; the non-convex loss landscape of language models further complicates targeted forgetting. Second, entropy-based masking operates at the token level rather than on underlying concept representations, often resulting in surface-level forgetting while distributed knowledge may persist. Third, the method is limited to parameter optimization and does not exploit architecture-specific strategies, even though knowledge is not stored uniformly across the model. These factors highlight that pure gradient-based unlearning has fundamental constraints, and that future research should consider layer-specific sensitivity analysis and structural interventions, such as targeted modifications to attention mechanisms or embedding spaces, to achieve more robust and conceptually precise unlearning.

7.2 MIA Protection Trade-offs

The experiments revealed a fundamental trade-off: achieving strong MIA protection currently requires accepting significant knowledge degradation. Demonstrating that reducing MIA defense priorities can improve knowledge retention highlights an ongoing challenge in unlearning systems, where competing objectives cannot be simultaneously optimized.

The sharp transition in MIA defense effectiveness (between weights 3.0-7.0) suggests potential for more sophisticated approaches. Future research should focus on techniques that provide MIA protection without sacrificing knowledge preservation, such as adversarial training methods or differential privacy integration.

7.3 Task-Specific Performance Variability

The significant performance variance across task types (particularly the Task1 knowledge score of 0.0 with standard parameters) indicates an inherent challenge in developing a one-size-fits-all approach to unlearning. This suggests the need for content-type specific approaches where unlearning is formulated as an explicit multi-objective optimization problem with learned weightings between competing objectives based on content characteristics.

7.4 Practical Constraints

The implementation faced several practical limitations: experiments were confined to the 1B parameter model, memory constraints prevented full implementation of promising extensions such as permanent noise layers, and the method assumes access to model gradients, which may not be available in black-box settings.

Future work should address these constraints through parameter-efficient fine-tuning methods (LoRA, adapters) and knowledge distillation approaches that could enable effective unlearning on resource-constrained environments and in black-box scenarios.

8 Code and Data Availability

To ensure reproducibility, the complete implementation code, hyperparameter configurations, and experimental scripts are publicly available at <https://github.com/uio-no/brageei/UnlearningExam>.

The final unlearned model checkpoint is available on HuggingFace at <https://huggingface.co/bragee/UnlearnedGBU>, and is also accessible on Fox at `/fp/projects01/ec403/ec-bragee/llm-unlearning/outputs/clean_unlearning_20250518_123800/best_model`. The repository includes detailed documentation for replicating all experiments reported in this paper and can be adapted for other unlearning benchmarks. All code is released under an open-source license.

References

- Jan Bronec and Jindřich Helcl. 2025. [Atyaephyra at semeval-2025 task 4: Low-rank negative preference optimization](#). In *Proceedings of the 19th International Workshop on Semantic Evaluation*, pages 883–890. Association for Computational Linguistics.
- A. Feder Cooper, Christopher A. Choquette-Choo, Miranda Bogen, Matthew Jagielski, Katja Filippova, Ken Liu, Alexandra Chouldechova, Jamie Hayes, Yangsibo Huang, Niloofar Mireshghallah, Ilia Shumailov, Eleni Triantafillou, Peter Kairouz, Nicole Mitchell, Percy Liang, Daniel E. Ho, Yejin Choi, Sanmi Koyejo, Fernando Delgado, and 16 others. 2024. [Machine unlearning doesn’t do what you think: Lessons for generative ai policy, research, and practice](#). Available at SSRN.
- Michael Duan, Anshuman Suri, Niloofar Mireshghallah, Sewon Min, Weijia Shi, Luke Zettlemoyer, Yulia Tsvetkov, Yejin Choi, David Evans, and Hannaneh Hajishirzi. 2024. [Do membership inference attacks work on large language models?](#) In *Proceedings of the 2024 Conference on Language Modeling (COLM)*. Association for Computational Linguistics.
- Chongyang Gao, Lixu Wang, Kaize Ding, Chenkai Weng, Xiao Wang, and Qi Zhu. 2025. [On large language model continual unlearning](#). In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*.
- Jinghan Jia, Jiancheng Liu, Yihua Zhang, Parikshit Ram, Nathalie Baracaldo, and Sijia Liu. 2024. [Wagle: Strategic weight attribution for effective and modular unlearning in large language models](#). In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*.
- Xiaomeng Jin, Zhiqi Bu, Bhanukiran Vinzamuri, Anil Ramakrishna, Kai-Wei Chang, Volkan Cevher, and Mingyi Hong. 2025. [Unlearning as multi-task optimization: A normalized gradient difference approach with an adaptive learning rate](#). In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 11278–11294, Albuquerque, New Mexico. Association for Computational Linguistics.
- Sijia Liu, Yuanshun Yao, Jinghan Jia, Stephen Casper, Nathalie Baracaldo, Peter Hase, Yuguang Yao, Chris Yuhao Liu, Xiaojun Xu, Hang Li, Kush R. Varshney, Mohit Bansal, Sanmi Koyejo, and Yang Liu. 2024. [Rethinking machine unlearning for large language models](#). *arXiv preprint arXiv:2402.08787*.
- Anil Ramakrishna, Yixin Wan, Xiaomeng Jin, Kai-Wei Chang, Zhiqi Bu, Bhanukiran Vinzamuri, Volkan Cevher, Mingyi Hong, and Rahul Gupta. 2025. [Semeval-2025 task 4: Unlearning sensitive content from large language models](#). *Preprint*, arXiv:2504.02883.
- Haoming Xu, Ningyuan Zhao, Liming Yang, Sendong Zhao, Shumin Deng, Mengru Wang, Bryan Hooi, Nay Oo, Huajun Chen, and Ningyu Zhang. 2025. [Relearn: Unlearning via learning for large language models](#). *Preprint*, arXiv:2502.11190.
- Jin Yao, Eli Chien, Minxin Du, Xinyao Niu, Tianhao Wang, Zezhou Cheng, and Xiang Yue. 2024. [Machine unlearning of pre-trained large language models](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8403–8419, Bangkok, Thailand. Association for Computational Linguistics.
- Xiaojian Yuan, Tianyu Pang, Chao Du, Kejiang Chen, Weiming Zhang, and Min Lin. 2025. [A closer look at machine unlearning for large language models](#). In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*.
- Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. 2024. [Negative preference optimization: From catastrophic collapse to effective unlearning](#). *Preprint*, arXiv:2404.05868.

Anti-clustering is probably not all you need

Unlearning sensitive content from Large Language Models with Anti-Clustering

Marius Aas Eidsaunet
Department of Informatics, University of Oslo
Oslo, Norway
mariuaei@ifi.uio.no

Victor Moene
Department of Informatics, University of Oslo
Oslo, Norway
victomoe@ifi.uio.no

Abstract

This paper explores methods for machine unlearning in large language models (LLMs), focusing on the selective removal of sensitive or unwanted data from a pretrained model. Inspired by the SemEval-2025 Task 4 benchmark, our work addresses the challenge of making a model forget a specified forget set while preserving knowledge from a retain set. We propose a new anti-clustering objective that directly modifies the hidden representations in the first five layers of the model to encourage dispersion among forget examples. Our experiments, conducted on the provided 1B parameter OLMo model, analyze the effects of this objective on the internal geometry of representations using metrics such as mean pairwise distance and silhouette scores. We also measure the performance of this anti-clustering on selective unlearning metrics. While our approach successfully increased dispersion among forget representations in early layers, it did not achieve selective unlearning, since both forget and retain examples were affected similarly. We discuss limitations related to batch size and centroid estimation, and suggest directions for further improving targeted unlearning in LLMs using anti-clustering methods. The code is available at: <https://github.uio.no/anti-clustering/WWNLP2025>

1 Introduction

Large Language Models (LLMs) have demonstrated exceptional performance across a broad spectrum of natural language processing tasks, including question answering, summarization, and dialogue generation. However, these models are also known to memorize portions of their training data, occasionally regurgitating sensitive, private, or copyrighted data. This behavior has led to a new class of attacks known as inference attacks (Kandpal et al., 2024). The tendency of LLMs

to memorize data introduces legal, ethical, and privacy-related concerns for model developers and vendors, especially in light of increasing regulatory pressures and evolving norms surrounding data protection and the "right to be forgotten." In response to these concerns, the field of machine unlearning has emerged to remove specific information from trained models without costly retraining which is a critical need for large-scale LLMs, where retraining from scratch is prohibitively expensive. (Bourtoule et al., 2019). Yet, unlearning in LLMs present unique challenges due to their vast parameter space, opaque representations, and entangled data dependencies, making precise and selective forgetting a non-trivial task.

The SemEval-2025 Task 4 challenge (Ramakrishna et al., 2025) formalizes this unlearning problem by introducing a shared benchmark with a clear objective. The task involves two datasets: one with data to be forgotten and another with knowledge to be retained. Participants are required to modify a pretrained LLM to unlearn the forget set while maintaining high performance on the retain set. Two pretrained models are provided: one with 1 billion parameters and another with 7 billion parameters. Performing this selective unlearning is challenging. Under-forgetting risks failing to delete all sensitive content, while over-forgetting can result in the loss of valuable general knowledge, often leading to catastrophic unlearning (Nguyen et al., 2024). This phenomenon highlights the trade-off between precision and preservation in model editing. To address this, we propose a novel *anti-clustering objective*, which operates directly on the internal hidden representations of the model. Our method aims to make forget examples appear as outliers in the model’s latent space, without significantly affecting the retain examples. By shifting the internal structure in this targeted way, we seek to al-

ter the model’s output to forget specific data while minimizing damage to its general capabilities. Our experiments were conducted on the /semeval25-unlearning-1B-model, although the overall idea is extendable to larger models. We assess the effect of our approach on internal representation geometry by performing cluster analysis to see how well the anti-clustering performed.

2 Background

2.1 Task background

As mentioned, the SemEval-2025 Task 4 task offers the two datasets Retain D_r and forget D_f . The goal is to remove the forget examples and keep knowledge of the retain examples. There are also multiple evaluations on three general tasks. These tasks are: synthetic creative writing, synthetic biographies and real Wikipedia biographies. The top-performing approaches in the competition commonly use parameter-efficient fine-tuning with model chunking (Premptis et al., 2025), model merging (Xu et al., 2025), and Low-Rank Negative Preference Optimization (Bronc and Helcl, 2025). These methods all address a central challenge: unlearning often goes too far, resulting in catastrophic forgetting. The core idea behind anti-clustering is to mitigate this issue by explicitly preventing over-unlearning.

2.2 Related work

Anti-clustering is conceptually similar to *contrastive learning*, commonly used in image learning tasks, with methods like SimCLR gaining popularity (Chen et al., 2020). Both methods move around representations. Contrastive learning, however, aims to pull positive pairs together, leading to meaningful clusters, whereas our method of anti-clustering aims to move representations further away from each other. Tampering and moving hidden representations within a model to unlearn a model is already a tested approach in LLM unlearning. (Huu-Tien et al., 2025) used an analysis of *representation misdirection* (RM) and showed that steering latent vectors of data slated for removal towards random targets erased the specific facts, but also made the model fragile. They used a backdoor attack-defense problem with noise-based stabilization, by adding small Gaussian noise to the representation of the retain examples to make it more robust. Our focus, on the other hand, is on *anti-clustering* by pushing forget examples in latent

space, directly moving their hidden layer representation. (Agrawal and Huang, 2025) showed empirically, via causal mediation analysis (CMA), that sensitive data lie mostly in the first five layers of the LLM using the OLMo architectures, which is the same transformer architecture we use. Those layers are therefore natural targets for the anti-clustering, to make it more efficient at affecting the sensitive data. By focusing anti-clustering efforts on these layers, we maximize the intervention on sensitive data while minimizing disruption to more abstract, general knowledge that is typically represented in higher layers. This choice is thus both empirically justified and computationally efficient. Clustering analysis is relevant for finding whether the embeddings did indeed drift away and the *Silhouette Coefficients* is a useful cluster-separation metric (Bagirov et al., 2023). This, combined with the pairwise distances and mean-to-standard deviation, measured both on the base and the fine-tuned models, validates whether the embeddings have actually become less clustered, as intended.

3 Method

Anti-Cluster Loss

Anti-clustering aims to push representations away from one another, particularly within a designated subset of data. The core idea is to prevent the model from forming tight clusters among examples we want it to forget. By disrupting these clusters in the model’s internal representations, anti-clustering reduces the likelihood that the model will memorize or regurgitate coherent, sensitive content. This approach takes as an assumption that sensitive data, such as specific sentences or facts, is encoded together in contiguous regions of the model’s latent space. By scattering these representations, anti-clustering inhibits the model’s ability to retrieve or reconstruct the original information.

To do so, all data points in the forget example are spread using their distance from the centroid of the retain example. We also move all forget examples away from each other pairwise to avoid clustering them together.

Figure 1 provides a simplified visualization of our proposed anti-clustering method. It illustrates how the anti-clustering loss L_{ANTI} is computed in two ways: by maximizing the pairwise distance among the forget representations themselves (black arrows between red crosses), and by pushing the forget examples away from the retain examples

(distance between clusters). The backward arrow indicates that this loss is propagated during training to update the model parameters.

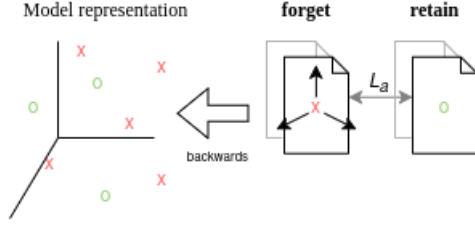


Figure 1: Schematic of anti-clustering: The distance between the cluster of forget and retain examples is propagated to the model representations.

The anti-cluster loss is propagated through a set of target layers ℓ via full fine-tuning. Anti-clustering can be formally defined as follows:

For each targeted layer ℓ , let

$$H_f^{(\ell)} = \{h_{f,i}^{(\ell)}\}_{i=1}^N, \quad H_r^{(\ell)} = \{h_{r,i}^{(\ell)}\}_{i=1}^N \quad (1)$$

be the pooled hidden vectors for the forget and retain batches.

The normalized forget representations are defined as follows:

$$\tilde{h}_{f,i}^{(\ell)} = \frac{h_{f,i}^{(\ell)}}{\|h_{f,i}^{(\ell)}\|} \quad (2)$$

and the normalized retain centroid:

$$\tilde{c}^{(\ell)} = \frac{1}{N} \sum_{i=1}^N \frac{h_{r,i}^{(\ell)}}{\|h_{r,i}^{(\ell)}\|} \quad (3)$$

The anti-cluster loss for each layer is:

$$L_{\text{ANTI}}^{(\ell)} = -\frac{1}{N} \sum_{i=1}^N \left\| \tilde{h}_{f,i}^{(\ell)} - \tilde{c}^{(\ell)} \right\| - \frac{1}{N^2} \sum_{i,j=1}^N \left\| \tilde{h}_{f,i}^{(\ell)} - \tilde{h}_{f,j}^{(\ell)} \right\| \quad (4)$$

This term pushes the representations of forget examples away from the typical location of retain examples in the hidden space, while preventing the forget representation to form clusters, forcing them to scatter in the representation space.

The hidden-norm regularization for each layer is:

$$L_{\text{REG}}^{(\ell)} = \frac{1}{N} \sum_{i=1}^N \|h_{f,i}^{(\ell)}\|^2 \quad (5)$$

Averaging over the set of targeted layers $\mathcal{L}$:

$$\mathcal{L}_{\text{ANTI}} = \frac{1}{|\mathcal{L}|} \sum_{\ell \in \mathcal{L}} L_{\text{ANTI}}^{(\ell)} \quad (6)$$

$$\mathcal{L}_{\text{REG}} = \frac{1}{|\mathcal{L}|} \sum_{\ell \in \mathcal{L}} L_{\text{REG}}^{(\ell)} \quad (7)$$

The total loss is then:

$$\mathcal{L}_{\text{total}} = w_{\text{anti}} \mathcal{L}_{\text{ANTI}} + w_{\text{reg}} \mathcal{L}_{\text{REG}} \quad (8)$$

Equation 8 defines the complete objective used to fine-tune the model. Here, $\mathcal{L}_{\text{ANTI}}$ encourages the representations of the forget set to become dispersed and separate from the centroid of the retain set, promoting de-clustering of sensitive data. The regularization term, $\mathcal{L}_{\text{REG}}$, penalizes excessive growth in the norm of the hidden vectors, which could otherwise result from maximizing distances. The scalar weights w_{anti} and w_{reg} balance the two objectives, allowing us to control the strength of anti-clustering relative to regularization. By adjusting these hyperparameters, we ensure that anti-clustering is effective without destabilizing model training or degrading generalization.

This term actively “pushes” the forget-set representations away from the retain centroid and away from each other, achieving the type of anti-clustering in latent space we are after. By L2-normalizing, the loss focuses on the direction of the representations in the latent space, as opposed to their raw magnitude. This avoids the model to simply increase vector norms to boost distances. When our model is fine-tuned with anti-clustering, we would like to be able to measure how it has un-clustered the representations of the model. We then compute the relevant metrics for cluster analysis using layer-wise dispersion.

Layer-wise Dispersion

To measure the cluster spread within each layer ℓ for the pre-trained model and the fine-tuned model, we analyze both pairwise differences and silhouette score on the embeddings. Let

$$\{\mathbf{h}_i^{(\ell)}\}_{i=1}^{N_\ell} \subset \mathbb{R}^d \quad (9)$$

be the pooled hidden vectors from layer ℓ , and define the pairwise distance matrix

$$D_{ij}^{(\ell)} = \|\mathbf{h}_i^{(\ell)} - \mathbf{h}_j^{(\ell)}\|. \quad (10)$$

We can then compute the mean and standard deviation

$$\mu_\ell = \frac{1}{N_\ell^2} \sum_{i=1}^{N_\ell} \sum_{j=1}^{N_\ell} D_{ij}^{(\ell)}, \quad (11)$$

$$\sigma_\ell = \sqrt{\frac{1}{N_\ell^2} \sum_{i=1}^{N_\ell} \sum_{j=1}^{N_\ell} (D_{ij}^{(\ell)} - \mu_\ell)^2}. \quad (12)$$

A high μ_ℓ indicates that the representations are spread out in the space, and for the forget set, should increase with anti-clustering. This value should stay roughly the same for the retain set for the anti-clustered model compared to the base model. Another useful metric to use when looking for clustering, is the *Coefficient of variation* which is $\frac{\sigma_\ell}{\mu_\ell}$. A high value here indicates non-uniform dispersion and possible clustering, while a low value suggests the points are more evenly spread. For anti-clustering, we want the forget examples to be evenly scattered around in the latent space, meaning we would like to decrease that value compared to the base model.

Silhouette Score Silhouette Score let us know how good the hidden representations are K-clustered. This is a standard cluster validity index that quantifies how well-separated and compact clusters are, with higher scores indicating better clustering structure (Bagirov et al., 2023).

It is defined as follows:

Given a clustering $\{C_1, \dots, C_k\}$ of the N_ℓ vectors (the hidden states from layer ℓ), define for each sample i :

$$a(i) = \frac{1}{|C_{c_i}| - 1} \sum_{\substack{j: c_j = c_i \\ j \neq i}} \|\mathbf{h}_i^{(\ell)} - \mathbf{h}_j^{(\ell)}\| \quad (13)$$

$$b(i) = \min_{c \neq c_i} \frac{1}{|C_c|} \sum_{j: c_j = c} \|\mathbf{h}_i^{(\ell)} - \mathbf{h}_j^{(\ell)}\|. \quad (14)$$

Then the per-sample silhouette is

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}, \quad (15)$$

and the mean over all points is defined as,

$$S_k^{(\ell)} = \frac{1}{N_\ell} \sum_{i=1}^{N_\ell} s(i). \quad (16)$$

This term measures average silhouette scores for values of K, that is, how well does the data fall into K clusters. A low $S_k^{(\ell)}$ indicates there are no clear K clusters. To find out whether the data separate into any clusters, we measure the mean Silhouette score for K=2 to K=5. If anti-clustering is working, this average should be low for forget. The Silhouette score should stay roughly unchanged for the retain set after applying anti-clustering. We can now quantify how the retain and forget examples are arranged in the hidden-state geometry of the base 1 Billion parameter model for the first five layers. We report these metrics first on the base model, then finally, on our most "anti-clustered" fine-tuned model which is the model that ends up having the lowest mean $S_k^{(\ell)}$ score for K=2 to K=5 across the five layers (aggregated using mean).

4 Experiments

We ran the full unlearning experiments on the semeval25-unlearning-1B OLMo model that is expected to be used by the SemEval task. We first ran a fine-tuning anti-clustering model with LoRA (Hu et al., 2021), however the model didn't seem to unlearn anything, thus we changed to full fine tuning. For our experiments we ran 5 tests each with different configurations of learning rate and retain weight. We found that a learning rate of 1e-5, and retain weight of 0.4 yielded the best results. The retain weight is the weight of the loss for the retain examples. This weight controls how much the model is penalized for making mistakes on the retain set. It scales a classic cross-entropy loss to make sure the model keeps retain data. Due to memory constraint, we were only able to run the tests with a batch-size of 8, which is admittedly quite low. For each run, we doubled the anti-clustering factor to see how it impacted the unlearning of the model.

4.1 Metrics

To assess the effectiveness of unlearning strategies, the SemEval-2025 task employed multiple metrics that capture both the degree of forgetting and the preservation of general model capabilities. The first one is *Regurgitation Rate*: For the sentence completion task, the extent to which the model retains the forget set is measured using the ROUGE-L score (Lin, 2004), where lower scores indicate more effective forgetting. We also measure *Knowledge Retention*: In the question answering task, exact

match accuracy is used to compute how well the model retains desired knowledge from the retain set. We measure Task Aggregate Score (Aggregate Harmonic Mean): This metric is calculated as the harmonic mean over all regurgitation and knowledge scores. Moreover, we measure *Membership Inference Attack* (MIA) Score. This score, also defined as MIA accuracy, quantifies the model’s vulnerability to privacy leakage by evaluating how easily an adversary can infer whether specific data was part of the training set. Finally, we evaluated our best model on the MMLU metric (Hendrycks et al., 2021). The MMLU metric stands for Massive Multitask Language Understanding. It is a benchmark designed to evaluate the general reasoning and knowledge capabilities of large language models across a wide range of academic and professional subjects. It allows us to assess whether the model retains its performance after anti-clustering. We trained our model on the provided train set and tested it on the provided validation set since we didn’t have access to the training set.

4.2 Findings

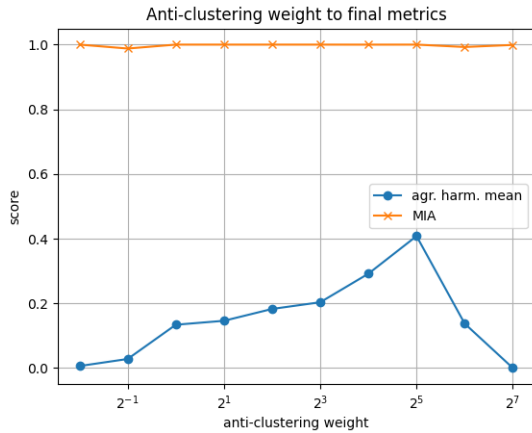


Figure 2: Plot showing MIA score and Aggregate Harmonic Mean score for each anti-clustering weight

In figure 2, we plot the Aggregate Harmonic Mean score and MIA score for each value of anti-clustering weight. We can see that the MIA score stays stable while the aggregate harmonic mean peaks when the anti-clustering is weighted by 32 and is at its lowest for the weights 0.25 and 128. Meanwhile, the MIA score seems to stay constant. We then evaluated the best model (the one where $ac_weight = 32$) on the MMLU task and got a score of 0.24 which is the same score as the original OLMo model achieved.



Figure 3: Plot showing regurgitation score and knowledge score for retain and forget sets for each anti-clustering weight

In the figure 3 we plot the overall regurgitation score and overall knowledge score for both forget and retain sets. We can see that their score is inversely proportionate to the weight of the anti-clustering, meaning that the anti-clustering weight does modify the representations of the model. However, we can also see that the curve of the forget and retain sets tightly follow each other, indicating that the model can’t selectively unlearn the forget set while preserving the retain set.

From the first and second plot and the MMLU score, we can deduce that the model is still retaining the sensitive information in its layers (constant MIA), however it undergoes attrition of the model’s general capabilities. Both its overall regurgitation score and overall knowledge scores decrease with higher anti-clustering weight. The MMLU-score indicates that the model has retained most of its capabilities, considering it scores so low on retain overall regurgitation score and overall knowledge score. This would indicate that the model seemingly only forgot the retain and forget sets

Compared to the other submissions, our best model doesn’t quite reach the level of the top performing submissions of the SemEval-2025 task 4. Table 1 shows the final scores.

It is unsure if the anti-clustering had its effect on the representation within the layers of the model. Thus we evaluate its clusters on the affected first five layers. We first begin with the base model and see how the retain and forget sets are clustered in Table 2.

Team	Task Aggregate	MIA score	MMLU avg
AILS-NTUA	0.964	0.857	0.242
SHA256	0.973	0.741	0.243
Atyaephyra	0.887	0.622	0.248
Best model	0.408	1	0.24
Worst model	0.006	1	N/A*
Unmodified	0	0	0.243

Table 1: One billion parameter leaderboard. *The MMLU avg is missing for our worst model, because we only computed it for our best model. The best model is the one with $ac_weight = 32$ and the worst one selected is the one with $ac_weight = 0.25$

Layer	Set	μ_ℓ	σ_ℓ/μ_ℓ	$\bar{S}_{2:5}^{(\ell)}$
1	retain	1.923	0.558	0.397
2	retain	1.492	0.419	0.320
3	retain	1.457	0.397	0.306
4	retain	1.723	0.360	0.303
5	retain	2.100	0.296	0.263
1	forget	1.878	0.545	0.389
2	forget	1.460	0.408	0.315
3	forget	1.429	0.393	0.306
4	forget	1.704	0.363	0.298
5	forget	2.086	0.300	0.272

Table 2: Metrics for the base model across layers 1–5, for both retain and forget sets.

Table 2 shows both forget and retain sets are well-clustered in all layers, indicating the model encodes both types of data in structured ways. After anti-clustering, our goal is to destroy the cluster structure of the forget set—reflected in low/negative silhouette scores, while also preserving the structure of the retain set. This would indicate successful unlearning at the representation level. We selected the most clustered model by computing the mean silhouette scores for $K=2:5$ on the first five layers of each model. We then averaged these scores across different anti-cluster weights. Figure 4 presents the average silhouette scores for each model across a range of anti-clustering weights, enabling us to quantify the dispersion achieved for the forget set.

It is evident that increasing the anti-clustering weight resulted in good anti-clustering in the beginning, but increasing the weight further, did not yield better spread among the forget examples. The overall loss landscape is not convex, and small changes in the anti-cluster weight can yield very different minima. Dips in the silhouette scores may indicate

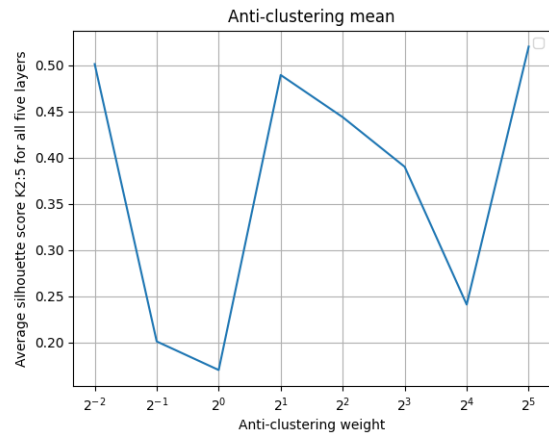


Figure 4: Plot showing average silhouette score for each model with different anti-clustering weight

that the model found a way to spread the points without truly achieving the intended anti-clustering. Peaks indicate that the model fell back to clustered structure, possibly because anti-clustering was too weak, or because the optimization pushed the representations to new, accidental clusters.

In Table 3 are the results of the model with lowest average silhouette score across the five first layers. This is the model which is most anti-clustered across the first five layers. It is clear that anti-clustering has a pronounced effect on the forget set. Representations are more dispersed (higher μ_ℓ) compared to the base model. Silhouette scores plummet, especially in higher layers, showing that the forget representations are now more uniformly distributed as k-means cannot identify strong clusters. This is the desired effect of anti-clustering, as it has successfully destroyed cluster structures among forget examples. For the forget set, the silhouette is low, confirming it is de-clustered, while the retain set representations form large, well-

Layer	Set	μ_ℓ	σ_ℓ/μ_ℓ	$\bar{S}_{2:5}^{(\ell)}$
1	retain	1.536	0.441	0.304
2	retain	3.111	2.420	0.749
3	retain	3.879	2.471	0.755
4	retain	5.103	3.092	0.743
5	retain	5.890	3.260	0.950
1	forget	1.489	0.392	0.311
2	forget	2.211	0.727	0.169
3	forget	2.631	0.597	0.108
4	forget	2.926	0.520	0.132
5	forget	3.247	0.472	0.131

Table 3: Metrics for the model with lowest average silhouette score across layers 1–5, for both retain and forget sets.

separated clusters, as indicated by the high silhouette score. However, the model showing the most anti-clustering is not the one performing the best on aggregated harmonic mean and MIA scores. It seems that there are no apparent correlations between the targeted selective unlearning and the anti-clustering objectives that we have experimented with.

5 Limitations

One of the limitations is that we couldn’t have access to the final test set to make our methodology more robust. Also, the small batch size of eight used in training may have negatively impacted the quality of centroid estimation and the stability of anti-clustering updates. This may have introduced noise and bias into the optimization process. Our approach focused exclusively on the first five layers based on prior empirical findings, but this choice may overlook some important effects in deeper layers or even cross-layer interactions. Additionally, computational constraints prevented us from conducting a more extensive hyperparameter search or running ablation studies, which could have further validated our conclusions.

6 Discussion and Conclusion

Our results indicate that the anti-clustering approach, as implemented, was insufficient for effectively solving the SemEval-2025 unlearning task. While we were able to apply anti-clustering objectives and quantitatively verify an increase in representation dispersion in the earlier layers, this did not translate to targeted unlearning of the forget set.

Instead, the model exhibited similar degradation for both the forget and retain examples, with no real meaningful distinction between the two in terms of forgetting. One clear explanation for this outcome is the limited batch size used during training.

With only 8 examples per batch, the computed retain centroid may be a poor proxy for any true centroid of the entire retain set considering that their representations were scattered randomly around the model’s latent space along with the forget set representations. This may have led to noisy or unstable anti-clustering updates. This, in turn, could cause the optimization to move forget examples apart indiscriminately, undermining the selective nature of the unlearning objective. Furthermore, such small batches are unlikely to provide a representative sample of the data distribution, limiting the effectiveness of both centroid calculation and overall anti-clustering. To address these issues, future work could explore strategies such as increasing the batch size significantly and using other mechanisms to better approximate the global centroid. There is definitively room for improvement for a more sophisticated anti-clustering objective than the one used here. Additionally, more targeted anti-clustering, to directly identifying and repel the same forget representations across batches may lead to more controlled and effective unlearning. It would also be interesting to combine anti-clustering with more established unlearning techniques, such as KL-divergence (Chen and Yang, 2023), or Negative Preference Optimization (NPO) (Zhang et al., 2024), to investigate whether anti-clustering can further improve unlearning performance. Overall, while anti-clustering showed promise in terms of manipulating latent space geometry, our findings suggest that significant methodological improvements are needed to achieve robust and selective unlearning with such an approach.

References

- Saransh Agrawal and Kuan-Hao Huang. 2025. [Sha256 at semeval-2025 task 4: Selective amnesia – constrained unlearning for large language models via knowledge isolation](#). *Preprint*, arXiv:2504.12996.
- Adil M. Bagirov, Ramiz M. Aliguliyev, and Nargiz Sultanova. 2023. [Finding compact and well-separated clusters: Clustering using silhouette coefficients](#). *Pattern Recognition*, 135:109144.
- Lucas Bourtole, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu

- Zhang, David Lie, and Nicolas Papernot. 2019. [Machine unlearning](#). *CoRR*, abs/1912.03817.
- Jan Bronec and Jindřich Helel. 2025. [Atyaephyra at semeval-2025 task 4: Low-rank negative preference optimization](#). *Preprint*, arXiv:2503.13690.
- Jiaao Chen and Diyi Yang. 2023. [Unlearn what you want to forget: Efficient unlearning for llms](#). *Preprint*, arXiv:2310.20150.
- Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. [A simple framework for contrastive learning of visual representations](#). *Preprint*, arXiv:2002.05709.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. [Measuring massive multitask language understanding](#). *Preprint*, arXiv:2009.03300.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. [Lora: Low-rank adaptation of large language models](#). *Preprint*, arXiv:2106.09685.
- Dang Huu-Tien, Hoang Thanh-Tung, Le-Minh Nguyen, and Naoya Inoue. 2025. [Improving the robustness of representation misdirection for large language model unlearning](#). *Preprint*, arXiv:2501.19202.
- Nikhil Kandpal, Krishna Pillutla, Alina Oprea, Peter Kairouz, Christopher A. Choquette-Choo, and Zheng Xu. 2024. [User inference attacks on large language models](#). *Preprint*, arXiv:2310.09266.
- Chin-Yew Lin. 2004. [ROUGE: A package for automatic evaluation of summaries](#). In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.
- Thanh Tam Nguyen, Thanh Trung Huynh, Zhao Ren, Phi Le Nguyen, Alan Wee-Chung Liew, Hongzhi Yin, and Quoc Viet Hung Nguyen. 2024. [A survey of machine unlearning](#). *Preprint*, arXiv:2209.02299.
- Iraklis Prempitis, Maria Lymperaio, Giorgos Filandrianos, Orfeas Menis Mastromichalakis, Athanasios Voulodimos, and Giorgos Stamou. 2025. [Ails-ntua at semeval-2025 task 4: Parameter-efficient unlearning for large language models using data chunking](#). *Preprint*, arXiv:2503.02443.
- Anil Ramakrishna, Yixin Wan, Xiaomeng Jin, Kai-Wei Chang, Zhiqi Bu, Bhanukiran Vinzamuri, Volkan Cevher, Mingyi Hong, and Rahul Gupta. 2025. [Semeval-2025 task 4: Unlearning sensitive content from large language models](#). *Preprint*, arXiv:2504.02883.
- Haoming Xu, Shuxun Wang, Yanqiu Zhao, Yi Zhong, Ziyang Jiang, Ningyuan Zhao, Shumin Deng, Hua-jun Chen, and Ningyu Zhang. 2025. [Zjuklab at semeval-2025 task 4: Unlearning via model merging](#). *Preprint*, arXiv:2503.21088.
- Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. 2024. [Negative preference optimization: From catastrophic collapse to effective unlearning](#). *Preprint*, arXiv:2404.05868.

6 Full dependency parsing of Norwegian track

Asymmetric Transfer Between Morphosyntactic Variants of Norwegian: The Case of Bokmål and Nynorsk

Eivind Nordahl Andersen

University of Oslo

Department of Informatics

eivindna@ifi.uio.no

Abstract

We investigate whether jointly training a transformer-based dependency parser on Norwegian Bokmål and Nynorsk improved performance compared to training on each language variant separately. Using a pre-trained BERT (NorBERT3) and Universal Dependency treebanks, we find that the cross-variant relation is asymmetric: Nynorsk shows modest performance gains, while Bokmål experiences a significant decline - primarily attributable to reduced accuracy in predicting fine-grained morphological features.

These findings suggest that even closely related language variants do not necessarily benefit from joint training. This highlights the need for a deeper understanding of when and why cross-variant transfer fails, particularly in contexts where linguistic similarity is presumed to be an advantage.

1 Introduction

The Universal Dependencies (UD) project (Nivre et al., 2016) provides a framework for consistently annotating grammatical, morphological, and syntactic relations across many languages. As of version 2.2 (Nivre et al., 2018), it includes over 70 languages and more than 120 annotated datasets, enabling large-scale cross-linguistic comparison.

Because the UD framework emphasizes harmonized annotation based on linguistic universals, it is particularly well-suited for evaluating cross-lingual generalization. Since treebanks follow consistent guidelines, performance differences are more likely to reflect true linguistic variation or cross-lingual effects, rather than annotation inconsistencies.

This raises an intriguing question: can a machine learning model trained on two morphosyntactic variants of the same language learn more robust representations than one trained on a single variant? In this project, we investigate whether fine-tuning

a transformer-based dependency parser on both variants of Norwegian improves performance - and whether such improvements are symmetric.

Norwegian has two official written standards: Bokmål and Nynorsk. While they are mutually intelligible and often treated as variants of the same language, they differ in morphology, syntax, and lemmatization. These differences make Norwegian a compelling case for studying cross-treebank transfer within a single language.

Our main contribution is an empirical analysis of cross-variant learning effects using a transformer-based parser. We evaluate performance at the morphological, syntactic, and lemmatization levels to assess whether shared training data benefits or hinders performance on each variant.

Interestingly, we find that the cross-variant transfer is asymmetric: Nynorsk shows modest improvements from joint training with Bokmål, while Bokmål performance deteriorates - particularly in the detailed morphological accuracy. This suggests that linguistic similarity alone does not guarantee mutual benefit in joint training.

These findings have practical implications for multilingual NLP, especially in deciding when to generalize across similar languages versus tailoring models to each variant. A better understanding of when transfer helps - or hurts - can inform the design of more effective cross-lingual systems.

2 Background

2.1 Dependency Parsing

Dependency parsing is the task of analyzing the grammatical structure of a sentence by identifying how words are connected - specifically, how they modify and depend on one another.

Early work in the mid-20th century took a grammar-based, context-free approach to parsing (Gaifman, 1965). This gradually evolved into data-driven methods for dependency grammar in the late 1990s (Eisner, 1996). In the following decades, higher parsing accuracies were achieved using linear discriminative models and support vector machines notably in the work of Matsumoto and colleagues in the early 2000s (Nivre et al., 2007; Yamada and Matsumoto, 2003).

The next major leap came with the advent of neural network-based models. Approaches using sequential recurrent networks, such as stacked LSTMs (Chen and Manning, 2014), and graph-based models like the biaffine attention parser (Dozat and Manning, 2017), pushed state-of-the-art performance.

In parallel, the Universal Dependencies (UD) project introduced a harmonized syntactic annotation framework across many languages (Nivre et al., 2016). This provided a consistent format for part-of-speech tagging and dependency treebanks. The Norwegian Dependency Treebank (Solberg et al., 2014) was converted to the UD format in 2016 (Øvrelid and Hohle, 2016), facilitating comparative parsing research on Norwegian data.

With the emergence of pre-trained transformer models such as BERT (Devlin et al., 2019), and especially multilingual variants, there has been a shift away from task-specific architectures toward general-purpose parsing systems. A prominent example is the UDify model by Kondratyuk and Straka, which achieves competitive performance across 75 languages using a single model (Kondratyuk and Straka, 2019). By leveraging contextual embeddings that capture both syntactic and morphological patterns in text, these models often outperform earlier graph-based and recurrent approaches.

2.2 Related Work

Velldal et al. (Velldal et al., 2017) identified a 20–25% drop in Labeled Attachment Score (LAS) when a dependency parser was trained exclusively on one Norwegian variant and evaluated zero-shot on the other. However, they also reported a modest performance improvement when training jointly on both variants. Based on these findings, they recommended using a mixed treebank when building a single cross-variant model.

In the context of Norwegian transformer-based models the NorBERT family of models (Samuel et al., 2023) provides a solid foundation for exploring syntactic variation across Bokmål and Nynorsk. These models are pre-trained on large Norwegian corpora and are particularly well-suited for downstream syntactic tasks.

Our work builds upon the NorBERT models by Samuel et al. (2023) and the biaffine dependency parser by Dozat and Manning (2017). We contribute to the Norwegian dependency parsing community by focusing on cross-variant transfer learning, with particular emphasis on morphological, syntactic, and lemmatization performance.

2.3 Dataset

For this study, we use the Norwegian Bokmål and Norwegian Nynorsk treebanks from the Universal Dependencies (UD) 2.2 release (Nivre et al., 2018). Although both represent varieties of the same language, we treat them as distinct languages throughout this work, in alignment with the UD framework’s treebank organization.

The datasets are provided in the CoNLL-U format (Nivre et al., 2016), a standardized structure for annotated linguistic data across languages. Each sentence is represented as a sequence of tokenized words, with each token annotated in ten tab-separated columns. This format captures multiple layers of linguistic information, including:

Lemma: the canonical or dictionary form of the word, i.e. the word "er" has the lemma "være".

Universal Part-of-Speech (UPOS): The coarse-grained morphological tag for the word, i.e. NOUN, VERB, ADJ et.c.

Universal Features (UFeats): Fine-grained morphological features to describe the grammatical properties of the word, i.e. tense, number, gender, etc.

Head: the syntactic head/parent of the current word represented by an index pointing to another word in the sentence.

Dependency Relation (Deprel): the syntactic relationship between a word and its head, i.e. subject, object, adjective modifier et.c.

In addition to the columns already described, each CoNLL-U token line includes five more fields.

Table 1 gives an overview of the remaining columns for each word in the CoNLL-U format.

Column	Description
ID	Word index in sentence
FORM	Surface form of the token
XPOS	Language-specific POS tag
DEPS	Enhanced dependency graph (optional)
MISC	Miscellaneous annotations (e.g., spaces)

Table 1: Overview of all ten CoNLL-U columns.

2.4 Evaluation metrics

To evaluate cross-treebank learning performance, we adopt the same evaluation framework as used in the 2018 Universal Dependencies (UD) Shared Task. Specifically, we rely on the official evaluation script provided by the UD organization, which computes a range of metrics for syntactic parsing quality. In our experiments, we focus on the three most comprehensive metrics: LAS, MLAS, and BLEX, all reported as F1 scores. Below is an intuitive overview of the three metrics we use:

Labeled Attachment Score (LAS): Measures whether both the predicted syntactic head and dependency relation for each word match the gold standard. LAS is the primary metric for syntactic parsing accuracy.

Morphological-Aware Labeled Attachment Score (MLAS): Extends LAS by also requiring the predicted universal part-of-speech (UPOS), morphological features (UFeats), and those of each word’s immediate children to match the gold annotations. This metric evaluates both syntactic structure and morphological accuracy, making it a stricter measure than LAS.

BiLexical Dependency Score (BLEX): Further extends LAS by also comparing the lemmatization of each word against the gold standard. BLEX therefore evaluates syntactic structure along with lexical correctness.

Evaluation is performed by comparing the predicted CoNLL-U formatted files against gold-standard annotated sentences in the same format. We use the official evaluation script from the UD Shared Task¹.

¹https://universaldependencies.org/conll18/conll18_ud_eval.py

3 Method

To ensure comparability across conditions, we use the same model architecture and training settings throughout, varying only the composition of the training data. Specifically, we fine-tune a pre-trained multilingual transformer under three configurations:

- Train on Bokmål only, evaluate on both Bokmål and Nynorsk.
- Train on Nynorsk only, evaluate on both Bokmål and Nynorsk.
- Train on the combined dataset, evaluate on both.

Evaluation is conducted on held-out test sets for each variant, using the standard UD metrics: LAS, MLAS, and BLEX. This setup allows us to isolate the effect of cross-variant exposure on parsing and tagging performance across morphological, syntactic, and lexical di

3.1 Preprocessing

During preprocessing, we extract the relevant features for each sentence. Each word is tokenized using the tokenizer associated with the chosen pre-trained transformer model.

Categorical annotations - UPOS, UFeats, and DepRel - to integer IDs based on the training set. These mappings are reused during validation and testing to maintain label consistency across all splits. The tokenizer internally handles the alignment between subword tokens and the original word-level labels. After tokenization, all sentences within a batch are padded to a uniform length, and an attention mask is generated to distinguish actual tokens from padding.

The number of unique UPOS tags, UFeat combinations, and DepRel labels differs across the two treebanks. Table 2 highlights the linguistic diversity present in the individual and combined datasets.

Feature	Bokmål	Nynorsk	Combined
UPOS	17	17	17
Ufeat	177	138	253
Deprel	37	40	40

Table 2: Diversity and annotation richness in each treebank.

3.2 Lemmatization

Our lemmatization approach consists of two main steps: (1) building a lemma lookup dictionary during training, and (2) generating and selecting lemmas during inference.

Step 1: Building a lemma lookup table during training

During training, we store information about how each word is tokenized and what its corresponding gold-standard lemma is. Specifically, we:

- Convert each word and its lemma to lowercase.
- Tokenize the word into subword units using the same tokenizer as the model.
- Record the subword composition of each word along with its corresponding lemma.
- Track which words each subword appears in, to allow reverse lookups during inference.

Step 2: Generating and selecting lemmas during inference

When the model encounters new text, we attempt to predict the correct lemma for each word using its subword tokenization. The candidate selection proceeds as follows:

- Attempt an exact match: look up training words that share the full subword sequence with the current word. If found, return the associated lemma.
- If no exact match is found, search for the longest matching subword prefix and suggest the corresponding lemma.
- If prefix-based suggestions fail, fall back to any lemma previously seen for related subwords.

From the resulting set of candidates, we apply simple linguistic heuristics to select the most appropriate lemma:

- For proper nouns, preserve capitalization if applicable.
- If the original word exists among the candidates, return it.

- Otherwise, apply suffix-stripping rules for common inflections in nouns, verbs, and adjectives, and select lemmas matching the resulting stem.
- If multiple candidates remain, select the lemma closest in length to the original word.
- As a final fallback, return the original word in lowercase.

This hybrid approach enables the model to handle both seen and unseen words by combining a data-driven lookup strategy with lightweight rule-based heuristics.

3.3 Model selection

We do not aim to push the state-of-the-art in parsing accuracy. Instead, we adopt a model that is strong enough to reliably learn basic syntactic and morphological patterns - ensuring that any observed transfer effects are not due to model randomness.. Our reference point is the CoNLL 2018 Shared Task baseline system.

Our architecture builds on NorBERT 3 (Samuel et al., 2023), a Norwegian bidirectional transformer. Subword representations from NorBERT are aggregated to word-level tokens, normalized, regularized, and passed through individual MLP heads to predict *UPOS*, *UFeat*, *HEAD*, and *DepRel* features.

Dependency arcs and relations are decoded using a biaffine attention mechanism (Dozat and Manning, 2017), and the model is trained with cross-entropy loss. To ensure valid syntactic trees - i.e., no cycles or self-loops - we incorporate the Kirchhoff Matrix-Tree Theorem (Koo et al., 2007), using the implementation from SuPar (Zhang et al., 2020).

As our interest lies in the comparative structural analysis rather than absolute accuracy, we do not perform extensive hyperparameter tuning. All experiments are run with fixed settings known to yield results on par with the CoNLL-U baseline² (Nivre et al., 2018).

²<https://universaldependencies.org/conll18/baseline.html>

Hyperparameters

All models were trained with the same configuration: we used the AdamW optimizer with a learning rate of 6×10^{-4} and weight decay of 1×10^{-7} . Training employed an effective batch size of 16, achieved via gradient accumulation over 4 micro-batches per optimizer step. All model parameter was updated during training. Each MLP head comprised three hidden layers with sizes [1536, 1536, 768], followed by a task-specific output layer mapping from 768 to the target feature dimension. BDropout with rate 0.2 was applied between hidden layers, and all activations were GELU. The transformer backbone was the NorBERT3-Small model.

3.4 Results

Table 3 summarizes the F1 scores for UPOS, UFeats, Lemmas, LAS, MLAS, and BLEX under three aforementioned training regimes, on both validation and test sets.

As expected, models trained on one variety perform poorly when evaluated on the other, reflecting clear annotation differences between Bokmål and Nynorsk. The most striking pattern emerges in the combined training condition: Nynorsk retains its performance - indeed, BLEX even improves slightly - when Bokmål data are added during training. In contrast, Bokmål’s performance deteriorates notably, particularly in UFeats (validation: 94.73 $\rightarrow$ 67.59; test: 94.37 $\rightarrow$ 67.89). LAS and MLAS also decline, though less dramatically, suggesting that fine-grained morphological predictions (captured in UFeats) are most sensitive to cross-variety training noise.

Part-of-speech tagging and LAS performance are consistent with the trends reported in Velldal et al. (Velldal et al., 2017), particularly Table 3 on page 8, where joint training yields modest average gains but reveals substantial variation across variants.

These findings indicate an asymmetric transfer effect: joint training either improves or preserves Nynorsk performance, while it undermines Bokmål’s capacity to model rich fine-grained morphological patterns.

4 Discussion

Our experiments reveal a clear asymmetry in cross-treebank transfer: Nynorsk benefits from the inclusion of Bokmål data, whereas Bokmål performance

degrades when trained on the combined dataset. The main driver of this decline is the UFeats metric, which drops by nearly 27% for Bokmål but remains essentially unchanged for Nynorsk. This suggests that conflicting morphological distributions - notably differences in gender marking and feature frequency - cause the shared model to overgeneralize patterns that align with Nynorsk but misrepresent Bokmål.

The observed drop in UFeat accuracy for Bokmål may also stem from our implementation of universal feature mappings. Although Table 2 correctly reflects overall feature richness, our setup maps UFeats using all values in the sixth column of the CoNLL-U files. This string-level encoding can lead to inconsistencies. For example, the Bokmål treebank may include both VerbForm=Inf and VerbForm=Inf|Voice=Pass, while only the former appears in the Nynorsk data. An alternative approach would be to separate individual morphological attributes — e.g., VerbForm, Voice, Gender — and predict each with a dedicated head. While such a design might yield more balanced cross-variant performance, we opted to treat UFeats as composite labels in this study. Given the relative richness of Bokmål’s feature inventory, however, the observed performance degradation is more severe than expected.

Nynorsk’s lower morphological diversity - that is, fewer unique UFeat combinations - appears to dominate the learned feature space, skewing the MLP parameters away from Bokmål-specific patterns. Still, LAS degrades only modestly, suggesting that syntactic structure transfers more robustly than morphology. One possible explanation is that the model gravitates toward a simpler morphological distribution, which presents fewer optimization conflicts. Rather than merely failing to generalize, the model may learn incorrect generalizations when exposed to incompatible morphological signals - actively harming Bokmål’s representation. This non-bijective transfer effect cautions against treating closely related treebanks as interchangeable: shared training data can introduce antagonistic signals that benefit one variant while impairing the other.

These findings invite broader reflection on the factors that govern successful cross-treebank transfer. Structural similarity alone may not guarantee compatibility. While syntactic annotations may be

Training Data	Bokmål (BM)		Nynorsk (NN)		Combined		Baseline ³	
Validation Data	BM	NN	BM	NN	BM	NN	BM	NN
Metrics								
UPOS	96.60	84.69	87.79	96.82	96.46	96.81	-	-
UFeats	94.73	58.94	60.83	95.61	67.59	95.50	-	-
Lemmas	93.84	75.99	79.07	93.55	87.62	94.12	-	-
LAS	85.36	64.39	73.09	87.30	86.37	85.92	84.56	82.75
MLAS	73.55	23.20	29.00	77.39	40.63	76.98	75.95	73.88
BLEX	72.81	39.87	48.44	74.47	71.2	75.12	78.04	75.76
Test Data	BM	NN	BM	NN	BM	NN	BM	NN
Metrics								
UPOS	96.27	85.17	87.02	96.73	95.94	96.60	-	-
UFeats	94.37	58.89	61.16	94.88	67.89	94.15	-	-
Lemmas	93.51	74.84	79.16	92.85	86.92	93.38	-	-
LAS	85.40	65.07	72.22	87.21	86.46	87.08	83.47	82.13
MLAS	73.62	23.58	28.95	76.70	40.89	76.29	74.65	72.40
BLEX	72.58	40.10	47.68	73.77	70.12	74.44	76.32	74.22

Table 3: F1 scores on test sets for each model’s predictions of syntactic dependencies, morphological features, and lemmatization. Models share identical hyperparameters and differ only in their train/validation data paths. “Combined” columns report the average of two runs using Bokmål or Nynorsk for validation checkpointing.

relatively stable across varieties, morphological annotation is more sensitive to linguistic nuance and editorial convention, making full isomorphism unlikely even within a single language. Alternatively, the asymmetry may not stem solely from linguistic properties, but also from annotation consistency. A treebank that is internally noisier or more heterogeneous may be more susceptible to interference when combined with another.

From a practical perspective, our results suggest two guidelines for multilingual NLP applications. First, avoid naïve mixing of treebanks when morphological precision is critical; consider instead reweighting or sampling strategies to balance feature distributions. Second, introduce explicit indicators of language variety or annotation source - e.g., treebank embeddings - to help disentangle divergent annotation schemes. Future work could investigate whether treebank-specific prediction heads or balanced sampling regimes mitigate the observed asymmetries, and how such strategies affect both morphological and syntactic outcomes.

5 Limitations

Several limitations of this study should be acknowledged. First, we did not perform systematic hyperparameter tuning. It remains possible that alternative training configurations - including different learning rate schedules, regularization techniques, or optimizer variants - could mitigate the observed performance degradation for Bokmål, particularly in the fine grained morphological feature predictions.

Second, our analysis focuses exclusively on two closely related language varieties within a single language family. While this offers a controlled environment for investigating structural alignment and transfer asymmetry, it also constrains the generalizability of our conclusions. Transfer dynamics may differ substantially in more typologically diverse or distantly related languages, where both linguistic and annotation differences are more pronounced.

Third, the lemmatization component employs heuristic strategies based on dictionary lookups and

UPOS-informed suffix rules. While effective for high-frequency patterns, this approach may introduce biases in development and may not generalize well to unseen or morphologically irregular forms. As such, errors in lemmatization could propagate to downstream evaluation metrics and skew interpretations of model performance.

6 Conclusion

Our findings reveal a clear asymmetry in cross-treebank transfer between Norwegian Bokmål and Nynorsk. While Nynorsk appears to benefit from joint training, Bokmål’s morphological accuracy - particularly in UFeats - declines substantially. This non-bijective transfer challenges the assumption that closely related language varieties can be combined without risk.

The results suggest that morphological features are more vulnerable to cross-varietal interference than syntactic ones, likely due to divergent feature distributions and possibly inconsistencies in annotation. In mixed settings, the model may overfit to the dominant or less ambiguous morphological system, misrepresenting the other. These dynamics underscore the importance of designing model architectures and training strategies that explicitly accommodate inter-treebank variation.

Future work should explore mitigation techniques such as reweighting, stratified sampling, or treebank-specific components to preserve variety-specific distinctions. Broader investigations across other closely related language pairs and more granular analyses of feature-level errors will be essential for understanding and addressing negative transfer in multilingual parsing.

Acknowledgments

All results have been verified through repeated training runs to ensure consistency. We gratefully acknowledge the use of automated tools, including Grammarly and OpenAI ChatGPT, for language refinement and stylistic editing. Any remaining errors are solely the responsibility of the authors.

References

- Danqi Chen and Christopher D Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 740–750.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.
- Timothy Dozat and Christopher D. Manning. 2017. [Deep biaffine attention for neural dependency parsing](#). In *International Conference on Learning Representations*.
- Jason M. Eisner. 1996. [Three new probabilistic models for dependency parsing: an exploration](#). In *Proceedings of the 16th Conference on Computational Linguistics - Volume 1, COLING ’96*, page 340–345, USA. Association for Computational Linguistics.
- Haim Gaifman. 1965. [Dependency systems and phrase-structure systems](#). *Information and Control*, 8(3):304–337.
- Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing universal dependencies universally. *arXiv preprint arXiv:1904.02099*.
- Terry Koo, Amir Globerson, Xavier Carreras, and Michael Collins. 2007. [Structured prediction models via the matrix-tree theorem](#). In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 141–150, Prague, Czech Republic. Association for Computational Linguistics.
- Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*, pages 1659–1666. European Language Resources Association.
- Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülşen Eryiğit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. [Maltparser: A language-independent system for data-driven dependency parsing](#). *Natural Language Engineering*, 13:95–135.
- Joakim Nivre and 1 others. 2018. Universal dependencies 2.2. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (UFAL), Faculty of Mathematics and Physics, Charles University. Available at: <http://hdl.handle.net/11234/1-2837>.

- Lilja Øvrelid and Petter Hohle. 2016. [Universal Dependencies for Norwegian](#). In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1579–1585, Portorož, Slovenia. European Language Resources Association (ELRA).
- David Samuel, Andrey Kutuzov, Samia Touileb, Erik Velldal, Lilja Øvrelid, Egil Rønningstad, Elina Sigdel, and Anna Palatkina. 2023. [NorBench – a benchmark for Norwegian language models](#). In *Proceedings of the 24th Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 618–633, Tórshavn, Faroe Islands. University of Tartu Library.
- Per Erik Solberg, Arne Skjærholt, Lilja Øvrelid, Kristin Hagen, and Janne Bondi Johannessen. 2014. [The Norwegian dependency treebank](#). In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 789–795, Reykjavik, Iceland. European Language Resources Association (ELRA).
- Erik Velldal, Lilja Øvrelid, and Petter Hohle. 2017. Joint ud parsing of norwegian bokmål and nynorsk. In *Proceedings of the 21st Nordic Conference on Computational Linguistics*, pages 1–10.
- Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. *Proc. Iwpt*.
- Yu Zhang, Zhenghua Li, and Zhang Min. 2020. [Efficient second-order TreeCRF for neural dependency parsing](#). In *Proceedings of ACL*, pages 3295–3305.

Norwegian Parsers and their Patterns Across the Scandinavian Language Space

A. Rudi Bråthen Bakir

University of Oslo

Abstract

This paper investigates full dependency parsing of Norwegian Bokmål and Nynorsk using a Norwegian pre-trained BERT transformer model. Rather than focusing on optimization, this study examines cross-lingual transfer between Norwegian Bokmål and Nynorsk and Scandinavian language neighbors. Models trained on Bokmål, Nynorsk, and a combined dataset, with and without language identification tags were compared. All experiments use the UD CoNLL-U format for training and evaluation.

Results show that joint modeling works well for syntactic relations, while morphological features prediction varied significantly between variants. Cross-lingual transfer results reflect the historical relationships between the languages: Nynorsk transfers better to Swedish and Bokmål transfers better to Danish. Adding language identification to a combined dataset resulted in decreasing performance in transfer to distant languages. This study provides insights into methods in dependency parsing across the Scandinavian language space.

1 Introduction

Dependency parsing is a method of mapping the syntactic structure of natural language. In this approach, sentences are broken up into their words and mapped to different grammatical features. Words are connected by the relations between them to form dependency tree structures. This structure consists of a dummy root, often at the start of the sentence. The dependency tree has directed edges between parent words (heads) and child words [Han et al. \(2020\)](#).

This paper will explore dependency parsing of Norwegian Bokmål and Nynorsk using a pretrained NorBERT-base transformer model. The goal is to investigate different ways to parse closely related languages. Through a systematic approach,

the model is trained on three different datasets: the official conllu datasets for Norwegian Bokmål and Nynorsk and a combination of these two. A comparison will be made between a baseline model trained for 1 epoch and a model trained for 5 epochs. Experimentation with cross-lingual transfer will also be conducted with the closely related languages Danish and Swedish. For evaluation, the official conllu evaluation script will be used to track the standard metrics for dependency parsing.

2 Background

The methodology of dependency parsing has evolved significantly, from rule-based to statistical models to modern neural systems. This section reviews key developmental turning points in the field to provide context for the current study.

2.1 General Dependency Parsing

[McDonald et al. \(2005\)](#) introduced graph-based parsing, while [Nivre \(2003\)](#) introduced transition-based parsing. These methods set the standard for methods in dependency parsing that are still used today.

The introduction of neural networks substantially changed the landscape of dependency parsing. [Chen and Manning \(2014\)](#) used a feed-forward neural network in transition-based dependency parsing. Later work by [Kiperwasser and Goldberg \(2016\)](#) used recurrent neural networks for both graph-based and transition-based approaches, with substantial performance gains.

The transformer architecture revolutionized NLP tasks ([Vaswani et al., 2017](#)). With the use of large language models like BERT ([Devlin et al., 2019](#)), dependency parsing can utilize rich, contextualized word embeddings for the downstream task. Different variants of BERT-type models exist, and can be used based on the downstream task. For example, a multilingual BERT model can be utilized

in cross-lingual dependency parsing, as shown by [Kondratyuk and Straka \(2019\)](#).

2.2 Scandinavian Dependency Parsing

The Universal Dependencies (UD) treebank for Norwegian, containing both Bokmål and Nynorsk variants, established the foundation for Norwegian dependency parsing research [Solberg et al. \(2014\)](#). This resource provided essential annotated data for developing and evaluating parsing models for Norwegian languages.

The universal dependency initiative ([Nivre et al., 2020](#)) has made it possible to compare dependency parsing across different languages. This standardized approach has helped the field develop with effective cross-lingual studies that are easily comparable, including the Scandinavian languages that made this current research possible.

For Scandinavian languages, [Øvrelid and Hohle \(2016\)](#) explored syntactic parsing of Norwegian and [Smith et al. \(2018\)](#) investigated cross-lingual learning between Norwegian, Danish and Swedish. [Velldal et al. \(2017\)](#) researched joint-modeling of Bokmål and Nynorsk, a study particularly relevant to this current study.

2.3 Language Models and Cross-lingual Transfer

Language-specific language models are showing promising results for Nordic languages. The NorBERT model ([Kutuzov et al., 2021](#)), which is used in this study, has shown significant improvement in performance compared to multilingual models used for Norwegian NLP tasks.

Cross-lingual transfer learning can be effective, especially for closely related languages, as shown by [Agić et al. \(2014\)](#). This approach was further explored by [Lim et al. \(2018\)](#), who showed that using a cross-lingual approach was better for parsing languages with few resources and no annotated corpora available.

Recent studies have shown that multilingual contextualized word embeddings improve dependency parsing across language borders and reduce performance differences in graph-based and transition-based dependency parsing [Kulmizev et al. \(2019\)](#).

This study aims to build on these findings and investigate how a Norwegian pretrained language model (NorBERT) can be used in dependency parsing of both Norwegian language variants, as well as cross-lingual transfer to the closely related Scandinavian languages.

2.4 Metrics for Evaluation

The official evaluation script for the CoNLL 2018 Shared Task [Zeman and Hajič \(2018\)](#) outputs multiple metrics with the main ones being:

- **LAS (Labeled Attachment Score):** The percentage of predicted words with correct heads and dependency labels. This is the standard evaluation metric for dependency parsing.
- **MLAS (Morphology-aware Labeled Attachment Score):** Extends LAS for cross-language evaluation by incorporating morphological features, making it particularly relevant for our cross-lingual experiments.
- **BLEX (Bi-lexical Dependency Score):** Similar to MLAS but uses lemmas instead of morphological features in evaluation calculations, with functional children ignored.

Additionally, the script provides scores for UPOS (Universal Part-of-Speech) and UFeats (Universal Features) predictions, which are useful for evaluating the tagger component of the pipeline.

These metrics give a comprehensive evaluation for both monolingual and cross-lingual experiments which are part of this study.

3 Methods

A neural dependency parsing pipeline for Norwegian was implemented and used for the experiments. The parser consists of three main parts: a data processor, a tagger and a dependency parser. The system core consists of the pre-trained NorBERT3-base model ([Samuel et al., 2023](#)), specifically trained for Norwegian with its rich contextualized representations. The pipeline is designed to capture the uniqueness of the Norwegian language while maintaining flexibility and opportunity for expansion to other languages.

3.1 Architecture

The system uses a modular approach where a tagger and a parser can be used independently or together. The data flows through the system as follows: (1) The conllu data-file is processed by the conludataset class, mapping and collecting the information between the tokens, universal part-of-speech and morphological features. (2) The processed dataset is sent to the tagger that assigns POS tags, morphological features and lemmas to each token. (3) Finally, the dependency parser predicts

the dependency relations between the words. The predictions are output in the official conllu format and evaluated with the conllu evaluation script. In addition, a bash script was used to streamline the training, prediction and evaluation steps.

3.2 Data Processing

The data processor extracts and manages the alignment between raw text, tokens and linguistic information in the Universal Dependencies format. Words, POS tags, Ufeats, lemmas and dependencies are extracted from the conllu files. Bidirectional mappings are created between these features and the words and subword tokens. The implementation also builds vocabularies from the processed dataset for the POS tags, dependency relations and morphological features. A dataset combiner was used to create two additional datasets, one combination of Bokmål and Nynorsk with random shuffling, and one with added language ID and random shuffling.

3.3 POS and Morphological Feature Tagging

The tagger implements predictive capabilities for POS, Ufeats and lemmas. The input is first encoded with NorBERT3. These representations are then passed to task-specific classification heads:

- (1) A single linear classifier for POS-tagging.
- (2) A multi-label classifier for morphological features.
- (3) A linear classifier for single label lemmas.

For lemmatization, a hybrid approach was implemented that uses a linear classifier combined with a rule-based approach and a pretrained Norwegian encoder-decoder model.

The idea was to create a hierarchical system with lemmatization difficulty as the organizing principle. This consisted of: (1) a feature-based classifier to predict lemmas for the most common words where statistical patterns were sufficient, (2) a rule-based approach to handle out-of-vocabulary words including irregular verbs, definiteness marking and compound words, and (3) a NorT5 encoder-decoder model for edge cases using contextualized lemmatization.

The rule-based approach was designed to handle both highly irregular Norwegian forms (e.g., "gikk/går" → "gå", "er/var" → "være") and pattern-based rules using morphological features like definite forms ("bilen" → "bil") and conjugation of

simpler verbs ("spiste" → "spise"). The neural method was designed to handle remaining edge cases by using few-shot prompting with examples.

Comprehensive statistics tracking was also implemented to analyze the performance of the different components of the system.

However, due to time constraints, the neural method was not finalized and is not used in the experiments. This resulted in a two-stage system: the linear classifier predicting lemmas for common word forms and a rule-based system handling OOV words and edge cases.

3.4 Dependency Parser

The dependency parser handles dependency parsing. This script uses a Biaffine attention (Dozat and Manning, 2017) adapted specifically for Norwegian. The system:

- (1) Processes contextualized word representations from NorBERT3.
- (2) Projects the representations on separate MLPs for head and dependent parts.
- (3) Uses Biaffine attention to predict:
 - probabilities of relations between each word pair;
 - probabilities of dependency relation types.

For head prediction, the biaffine scoring function is employed:

$$s_{ij}^{(arc)} = h_i^T U^{(arc)} d_j + W_h^{(arc)} h_i + W_d^{(arc)} d_j + b^{(arc)} \quad (1)$$

where h_i and d_j are head and dependent representations, and $U^{(arc)}$, $W_h^{(arc)}$, $W_d^{(arc)}$, and $b^{(arc)}$ are learnable parameters.

To ensure well-formed dependency trees, these techniques are implemented:

- Cycle detection and resolution mechanisms
- Root constraints to enforce a single root per sentence
- Special handling for non-projective dependencies, which are common in Norwegian

3.5 Training and Optimization

The models were trained on the Norwegian UD treebanks for both Bokmål and Nynorsk. Several techniques were used to improve stability and performance like, lower learning rate for head predictions, loss masking for padding tokens and linear learning rate with warmup. The models used an AdamW optimizer with a learning rate of $3e-5$ and weight decay of 0.01. Batch sizes were different between the tagger and the parser with respectively 8 and 16 as batch size for the training. Convergence occurred after 3-5 epochs. The models were trained with RTX 4080 super on a home computer.

3.6 Pipeline Integration

The pipe script integrates the parser and the tagger to a complete dependency parser. The script contains both a training mode and a prediction mode. The training mode has the option of parallel or sequential training.

4 Experimental Design and Results

This section presents the experimental setup and the corresponding results for dependency parsing of Norwegian Bokmål and Nynorsk as well as cross-lingual transfer for Swedish and Danish.

4.1 Setup

The experimental setup consisted of 3 phases. Phase 1 established a baseline by training the models for 1 epoch on Bokmål, Nynorsk and combined dataset. Phase 2 consisted of fine-tuning the models to a satisfying level, even though a methodological decision was made to not focus on fine-tuning due to time constraints. These models were therefore trained for 3 epochs, which resulted in a solid improvement in performance, while not spending too much time on this part. In phase 3, the models were tested in cross-lingual transfer to Swedish and Danish. All models were evaluated using the official CoNLL 2018 evaluation script mentioned in Section 2.4.

4.2 Datasets

Dataset	Train	Dev	Test
Bokmål	15,696	2,410	1,939
Nynorsk	14,174	1,890	1,511
Swedish	3,457	1,118	1,121
Danish	4,383	564	565

Table 1: Sentence counts in the UD treebanks used in our experiments.

The Norwegian UD treebanks for Bokmål and Nynorsk served as the primary datasets for the experiments. The Bokmål treebank contains 20,045 sentences with 311,277 tokens, while the Nynorsk treebank contains 17,575 sentences with 301,353 tokens. Both treebanks include a variety of genres such as news articles, blogs, parliamentary proceedings, and government texts.

For cross-lingual experiments, the Swedish-LinES treebank (5,696 sentences, 102,538 tokens) and the Danish-DDT treebank (5,512 sentences, 100,733 tokens) were used. The Swedish treebank contains 42 morphological feature types and 10,864 unique lemmas, while the Danish treebank has 44 feature types and 13,228 unique lemmas. Both languages use similar annotation schemas to Norwegian, with 17 POS tags and 41-42 dependency relations (including language-specific relations)

In creating the data splits for all treebanks, care was taken to preserve contiguous texts and maintain genre balance. The Swedish data is divided into training (3,457 sentences), development (1,118 sentences), and test (1,121 sentences) sets. Similarly, the Danish data is split into training (4,383 sentences), development (564 sentences), and test (565 sentences) portions.

Two additional datasets were created from the Norwegian datasets for additional experiments: a simple shuffled combination of both Bokmål and Nynorsk treebanks, and a combination with explicit language ID tags added to each sentence. These combined datasets were used to investigate the effectiveness of joint training across closely related language varieties.

All datasets follow the Universal Dependencies annotation scheme with 35 dependency relations (including 2 language-specific relations) and 17 POS tags. The two combination datasets consisted of: (1) a simple shuffled merger of Bokmål and Nynorsk training data, and (2) a combined dataset with explicit language ID tags added to each sentence.

4.3 Results

Table 2 shows the performance of the models on their in-language test sets for 1 and 5 epoch training. These results show some interesting patterns. First, the dependency parsing metrics (UAS, LAS, CLAS), have comparable performance for Bokmål and Nynorsk, with the combined model achieving the best scores (92.31% UAS, 84.58% LAS. This indicates that joint training on both Norwegian vari-

Table 2: Performance (F1-scores) across Norwegian variants and training settings

Metric	Bokmål		Nynorsk		Combined
	1e	5e	1e	5e	
<i>Morphosyntactic Analysis</i>					
UPOS	98.40	98.66	79.28	79.71	98.77
UFeats	96.12	97.50	51.53	51.54	50.43
Lemmas	83.80	86.80	21.99	22.27	17.61
<i>Dependency Parsing</i>					
UAS	90.64	90.71	89.38	88.60	92.31
LAS	82.96	83.03	82.30	81.56	84.58
CLAS	75.17	75.45	74.74	73.87	77.60
<i>Combined Metrics</i>					
MLAS	68.78	71.23	11.91	12.09	16.51
BLEX	53.54	57.57	4.34	4.61	4.66

ants improves the model’s ability to capture syntactic relations.

For the morphosyntactic analysis, the results differ significantly between the models. The model trained on Bokmål showed excellent results (98.66% UPOS, 97.50% UFeats), while the performance on Nynorsk was poor, especially for UFeats (51.54%) and lemmatization (22.27%). This suggests that there are significant challenges in the transfer of these elements, versus the syntactic structures or something wrong with the implementation.

The combined metrics MLAS and BLEX, that combine syntactic relations with morphology, show the same story. The big difference between the BLEX scores of Bokmål (57.57%) and Nynorsk (4.61%) marks the difficulty of lemmatization in cross-variants.

Longer training gave the best performance for Bokmål, and mixed results for Nynorsk, with some gains for morphosyntactic analysis, while a slight decrease in parsing accuracy.

Model	Bokmål → Nynorsk			Nynorsk → Bokmål		
	UAS	LAS	UPOS	UAS	LAS	UPOS
B1-Source	90.71	83.03	98.66	88.60	81.56	79.71
B1-Target	87.01	78.59	97.11	88.19	79.89	56.04
B1combined	91.66	84.14	98.74	92.85	84.95	98.79

Table 3: Cross-variant transfer between Norwegian variants (5-epoch models). B1-Source shows in-language performance, while B1-Target shows zero-shot cross-variant performance.

Table 3 shows the cross-variant transfer results. The results reveal several interesting patterns. Dependency parsing metrics shows strong transfer both ways, with minimal performance drop. The Bokmål model predictions on Nynorsk data shows UAS (90.71% → 87.01%) and LAS (83.03% → 78.59%). Interestingly, the Nynorsk model did

very well on Bokmål parsing, with UAS (88.60% → 88.19%) and LAS (81.56% → 79.89%).

The story is different for POS tagging. The results show assymetric transfer between the models, with the Bokmål model maintaining strong UPOS when tagging Nynorsk (98.66% → 97.11%). However, the other way results in dramatic decrease (79.71% → 56.04%), indicating lack of POS tagging generalization to Bokmål by the Nynorsk model.

The combined model trained on both variants shows the best results for all metrics and for both Norwegian variants.

This model achieves the highest scores for both Bokmål (92.85% UAS, 84.95% LAS, 98.79% UPOS) and Nynorsk (91.66% UAS, 84.14% LAS, 98.74% UPOS), demonstrating the benefits of joint training.

Model	Swedish			Danish		
	UAS	LAS	UPOS	UAS	LAS	UPOS
B1-BM	67.30	51.73	25.90	74.32	62.15	88.00
B1-NN	70.07	55.77	60.09	74.34	61.49	49.13
B1combined	72.53	57.64	48.18	77.06	64.07	87.24

Table 4: Cross-lingual transfer to Swedish and Danish (5-epoch models). Best scores in bold.

Cross-lingual transfer experiments (table 4) show how Norwegian-trained models perform on zero-shot transfer on related Scandinavian languages. For syntactic parsing, the models performed reasonably, with the combined model doing best for both target languages (72.53% UAS for Swedish and 77.06% UAS for Danish).

There are also some language specific transfer differences. The Nynorsk model (B1-NN) performed better on Swedish with higher UAS(70.07% vs. 67.30%), LAS(55.77% vs. 51.73%), and way higher UPOS accuracy (60.09% vs. 25.90%). This indicates that Nynorsk share more morphosyntactic relations with Swedish than Bokmål does.

Both Norwegian variants show similar parsing performance on Danish, with Bokmål and Nynorsk UAS at (74.32% vs. 74.34%). For the UPOS score there is high variation between the models with Bokmål scoring 88.00%, while the Nynorsk model achieves 49.13%. Closer lexical and morphological similarities between Bokmål and Danish is indicated here.

The combined model (B1combined) struggles with UPOS tagging and performs worse than the best single-variant model. This suggests decreased transferability of morphological features in joint

training.

Test Language	Model	UAS	LAS
Norwegian	B1combined	92.31	84.58
	B1combinedID	92.32	84.71
Swedish	B1combined	72.53	57.64
	B1combinedID	67.50	52.99
Danish	B1combined	77.06	64.07
	B1combinedID	76.80	64.40

Table 5: Impact of language identification on parsing performance across the North Germanic language continuum. Best scores for each language in bold.

Including explicit language ID, seems to have little effect on parsing performance, with B1combinedID model slightly higher UAS (92.32% vs. 92.31%) and LAS (84.71% vs. 84.58%). This suggests that the model distinguishes the languages well without language ID.

Language identification seems to have significantly reduced cross-lingual transfer. In Swedish, the B1combinedID performed worse contra the standard combined model, with UAS dropping 5% (72.53% → 67.50%) and LAS (57.64% → 52.99%). This decrease can mean that the explicit language identification may reduce flow across language boundaries, when the target language is included in the training.

Language identification did not have as big impact on the transfer to Danish. The UAS decreased slightly (77.06% → 76.80%), while the LAS shows a minor improvement (64.07% → 64.40%). The results indicate language ID has little effect on transfer to Danish, probably due to the similarities to Bokmål.

5 Discussion and Conclusion

5.1 In-language Experiments

The in-language experiments showed striking differences in performance between Bokmål and Nynorsk, especially for morphosyntactic analysis. Both models showed comparable results in dependency parsing (UAS, LAS), while the Nynorsk model had dramatically lower scores for morphological features (51.54% vs. 97.50% UFeats) and lemmatization (22.27% vs. 86.80%), as shown in Table 2.

These results warrant careful interpretation. Although NorBERT3 was trained on both Bokmål and Nynorsk, Nynorsk is less represented "reflecting the natural usage" as noted by Samuel et al. (2023), the striking difference in the performance is inconsistent with previous studies on the same

dataset, where Nynorsk has comparable results to Bokmål. This strongly suggests an implementation error in the morphological pipeline rather than linguistic difficulties.

Nevertheless, the consistent finding that joint modeling works well for syntactic parsing across variants suggests that shared syntactic structures are effectively captured regardless of variant-specific morphological complexities.

5.2 Cross-variant Transfer

The cross-variant transfer experiments revealed asymmetries between Bokmål and Nynorsk. Dependency parsing worked well both ways, and shows that syntactic structures are similar. This further confirms the results from the in-language experiments.

UPOS results show asymmetry between the models. Bokmål models transferred well to Nynorsk (98.66% → 97.11%), while the Nynorsk model performed poorly on Bokmål (79.71% → 56.04%). This is very surprising and may be caused by potential issues in the dataset or a bug in the experimental setup that warrants further investigation.

However, these findings may contribute practical information about Norwegian NLP. With limited Nynorsk data, Transfer from Bokmål to Nynorsk may be an effective approach. In addition, the combined model outperformed both single-variant models, confirming the benefit of joint-modeling when enough data is available.

5.3 Cross-lingual Transfer

In cross-lingual transfer to Scandinavian languages, the models did as expected based on linguistic understanding of the North Germanic language family. Some interesting findings showed that Nynorsk model transferred better to Swedish and the Bokmål model was better suited for Danish.

The patterns observed reflect the language relationships well, with Nynorsk and Swedish sharing features due to the rural dialectal influences. Bokmål on the other hand, is very similar to Danish due to Norway's union with Denmark. The very strong results for the Bokmål model on Danish UPOS (88.00%) highlight how close the lexical and morphological similarities are.

These results also show how syntactic structures (UAS) are the most transferrable cross-lingually. This is well documented in cross-lingual research showing that morphological features are more language-specific Pires et al. (2019).

The combined model showed strong results in cross-lingual transfer. The model gets increased semantic range from training on multiple languages and can help transfer to these closely related languages. An interesting research opportunity could be to explore how this expansion of semantic scale happens when languages of different distances are introduced in training. Due to time constraints, experiments with training on Swedish and Danish or a combination, was not done. But this would be a very interesting path to explore further.

5.4 Language Identification

The impact of adding language identification tags, affected the models in varied ways. There was a marginal increase in Norwegian parsing performance, suggesting that the model distinguishes between the variants at a satisfactory level.

Even so, the story was very different for cross-lingual transfer to Swedish, substantially hindering transfer between the languages. For Danish, the impact was minimal, further echoing the closeness between Danish and Norwegian Bokmål. These findings indicate that adding language identification can give slight gains in close languages and at the same time reducing the transferability to languages further apart. It increases specificity and reduces generalization. For zero-shot transfer capabilities, language ID may not be the best option.

6 Conclusion

In conclusion, this study provides insights into Norwegian dependency parsing and cross-lingual transfer to Scandinavian languages. It demonstrated different approaches to parsing the two Norwegian variants and their effects on model performance. Joint modeling of Norwegian variants proved effective for syntactic parsing, while morphological analysis showed significant implementation challenges that warrant further investigation. Additionally, zero-shot cross-lingual transfer was demonstrated to be effective, with patterns reflecting the historical relationships between the source and target languages. Future work could look at optimizing each module the tagger and the parser to improve overall performance. This would further extend the capabilities and flexibility of dependency parsing in close proximity language spaces.

Acknowledgments

I am grateful to Andrei Kutuzov, David Samuel, Roxana Pop, Erik Velldal, Sondre Wold and Vladislav Mikhailov for their excellent instruction in Neural Methods in Natural Language Processing. I also acknowledge that AI language models were used in the editing and refining of this paper as well as searching for relevant literature.

References

- Željko Agić, Jörg Tiedemann, Danijela Merkle, Simon Krek, Kaja Dobrovoljc, and Sara Može. 2014. [Cross-lingual dependency parsing of related languages with rich morphosyntactic tagsets](#). In *Proceedings of the EMNLP 2014 Workshop on Language Technology for Closely Related Languages and Language Variants*, pages 13–24, Doha, Qatar. Association for Computational Linguistics.
- Danqi Chen and Christopher Manning. 2014. [A fast and accurate dependency parser using neural networks](#). In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar. Association for Computational Linguistics.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of deep bidirectional transformers for language understanding](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Timothy Dozat and Christopher D. Manning. 2017. [Deep biaffine attention for neural dependency parsing](#). *Preprint*, arXiv:1611.01734.
- Wenjuan Han, Yong Jiang, Hwee Tou Ng, and Kewei Tu. 2020. [A Survey of Unsupervised Dependency Parsing](#). *arXiv preprint*. ArXiv:2010.01535 [cs].
- Eliyahu Kiperwasser and Yoav Goldberg. 2016. [Simple and accurate dependency parsing using bidirectional LSTM feature representations](#). *Transactions of the Association for Computational Linguistics*, 4:313–327.
- Dan Kondratyuk and Milan Straka. 2019. [75 languages, 1 model: Parsing Universal Dependencies universally](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China. Association for Computational Linguistics.
- Artur Kulmizev, Miryam de Lhoneux, Johannes Gontrum, Elena Fano, and Joakim Nivre. 2019. [Deep](#)

- contextualized word embeddings in transition-based and graph-based dependency parsing - a tale of two parsers revisited. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2755–2768, Hong Kong, China. Association for Computational Linguistics.
- Andrey Kutuzov, Jeremy Barnes, Erik Velldal, Lilja Øvrelid, and Stephan Oepen. 2021. [Large-scale contextualised language modelling for Norwegian](#). In *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 30–40, Reykjavik, Iceland (Online). Linköping University Electronic Press, Sweden.
- KyungTae Lim, Niko Partanen, and Thierry Poibeau. 2018. [Multilingual dependency parsing for low-resource languages: Case studies on north saami and Komi-Zyrian](#). In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).
- Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. [Non-projective dependency parsing using spanning tree algorithms](#). In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada. Association for Computational Linguistics.
- Joakim Nivre. 2003. [An efficient algorithm for projective dependency parsing](#). In *Proceedings of the Eighth International Conference on Parsing Technologies*, pages 149–160, Nancy, France.
- Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. [Universal Dependencies v2: An evergrowing multilingual treebank collection](#). In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4034–4043, Marseille, France. European Language Resources Association.
- Lilja Øvrelid and Petter Hohle. 2016. [Universal Dependencies for Norwegian](#). In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC’16)*, pages 1579–1585, Portorož, Slovenia. European Language Resources Association (ELRA).
- Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. [How multilingual is multilingual bert?](#) *Preprint*, arXiv:1906.01502.
- David Samuel, Andrey Kutuzov, Samia Touileb, Erik Velldal, Lilja Øvrelid, Egil Rønningstad, Elina Sigdel, and Anna Palatkina. 2023. [NorBench – a benchmark for Norwegian language models](#). In *Proceedings of the 24th Nordic Conference on Computational Linguistics (NoDaLiDa)*, pages 618–633, Tórshavn, Faroe Islands. University of Tartu Library.
- Aaron Smith, Bernd Bohnet, Miryam de Lhoneux, Joakim Nivre, Yan Shao, and Sara Stymne. 2018. [82 treebanks, 34 models: Universal Dependency parsing with multi-treebank models](#). In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 113–123, Brussels, Belgium. Association for Computational Linguistics.
- Per Erik Solberg, Arne Skjærholt, Lilja Øvrelid, Kristin Hagen, and Janne Bondi Johannessen. 2014. [The Norwegian dependency treebank](#). In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC’14)*, pages 789–795, Reykjavik, Iceland. European Language Resources Association (ELRA).
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). *Preprint*, arXiv:1706.03762.
- Erik Velldal, Lilja Øvrelid, and Petter Hohle. 2017. [Joint UD parsing of Norwegian Bokmål and Nynorsk](#). In *Proceedings of the 21st Nordic Conference on Computational Linguistics*, pages 1–10, Gothenburg, Sweden. Association for Computational Linguistics.
- Daniel Zeman and Jan Hajič, editors. 2018. [Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies](#). Association for Computational Linguistics, Brussels, Belgium.

A Appendix: Detailed Experimental Results

A.1 Cross-lingual Transfer between Norwegian Variants

Metric	Nynorsk		Combined	
	B0	B1	B0	B1
UPOS	75.82	56.04	98.59	98.79
UFeats	55.77	50.33	50.77	50.83
Lemmas	23.08	22.75	18.28	18.77
UAS	89.23	88.19	92.89	92.85
LAS	80.75	79.89	85.06	84.95
CLAS	72.98	72.26	78.04	77.89
MLAS	14.51	11.31	16.48	16.59
BLEX	4.39	4.01	4.27	4.86

Table 6: Zero-shot cross-lingual results on Bokmål. B0 was trained for 1 epoch, B1 was trained for 5 epochs.

A.2 Language ID Models

A.3 Cross-lingual Transfer to Other North Germanic Languages

Metric	Bokmål		Combined	
	B0	B1	B0	B1
UPOS	94.80	97.11	98.36	98.74
UFeats	64.42	65.89	49.76	49.95
Lemmas	61.83	65.68	16.01	16.22
UAS	86.82	87.01	92.19	91.66
LAS	78.65	78.59	84.71	84.14
CLAS	70.41	70.54	77.97	77.26
MLAS	28.59	30.65	16.32	16.41
BLEX	33.90	38.90	4.09	4.42

Table 7: Zero-shot cross-lingual results on Nynorsk. B0 = 1 epoch, B1 = 5 epochs.

Metric	Single-Language		Combined
	Bokmål	Nynorsk	
UPOS	98.77	98.64	98.71
UFeats	50.85	49.93	50.43
Lemmas	19.58	17.13	18.47
UAS	92.60	91.98	92.32
LAS	84.76	84.65	84.71
CLAS	77.64	77.59	77.62
MLAS	16.46	16.43	16.45
BLEX	6.01	5.69	5.86

Table 8: Results for B1combinedID model with language identification. The model was trained on the combined dataset for 5 epochs and evaluated on Bokmål, Nynorsk, and combined test sets.

Metric	Single-Language		Combined	
	Bokmål	Nynorsk	B1combined	B1combinedID
UPOS	88.00	49.13	87.24	83.33
UFeats	51.11	43.84	42.55	42.46
Lemmas	50.15	9.02	7.04	9.65
UAS	74.32	74.34	77.06	76.80
LAS	62.15	61.49	64.07	64.40
CLAS	54.29	53.70	56.10	56.70
MLAS	16.08	6.58	9.34	9.03
BLEX	27.56	2.18	3.16	5.36

Table 9: Zero-shot cross-lingual results on Danish. All models were trained for 5 epochs (B1).

Model	UPOS	UAS	LAS	CLAS
B1-BM	25.90	67.30	51.73	43.69
B1-NN	60.09	70.07	55.77	47.44
B1combined	48.18	72.53	57.64	49.49
B1combinedID	36.63	67.50	52.99	45.13

Table 10: Zero-shot cross-lingual transfer results on Swedish. Best scores in bold.

7 Fact verification track

GQC: LLM-Based Grouped QA Consolidation for Open-Domain Fact Verification at AVeriTeC

Dongzhuoran Zhou

University of Oslo

dongzhuz@uio.no

Abstract

Structured fact verification benchmarks like AVeriTeC decompose claims into QA pairs to support fine-grained reasoning. However, current systems generate QA pairs independently for each evidence sentence, leading to redundancy, drift, and noise. We introduce a modular LLM-based QA consolidation module that jointly filters, clusters, and rewrites QA pairs at the claim level. Experiments show that this method improves evidence quality and veracity prediction accuracy. Our analysis also highlights the impact of model scale and alignment on downstream performance.

1 Introduction

Automated fact verification aims to assess the veracity of natural language claims by retrieving and reasoning over external evidence (Thorne et al., 2018; Wang, 2017; Augenstein et al., 2019). While early systems typically treat this as a binary or multi-class classification problem using retrieved evidence as input, recent benchmarks—notably AVeriTeC (Schlichtkrull et al., 2023)—have introduced a structured pipeline where systems first generate clarification question–answer (QA) pairs based on retrieved evidence, and then use these QA pairs as an intermediate reasoning scaffold for final veracity prediction.

This structured QA paradigm improves transparency and evaluation granularity, but also introduces new challenges. While structured QA pipelines enable interpretability, they introduce new challenges. Systems like HerO (Yoon et al., 2024) generate QA pairs independently per sentence, resulting in overlapping or off-topic content that may confuse the final verifier. This redundancy inflates input length and can suppress relevant evidence. We propose a claim-level consolidation module to address these limitations and improve precision without sacrificing recall.

To address these issues, we introduce a simple and modular post-processing module that filters, clusters, and rewrites QA pairs using a large language model (LLM). By reasoning jointly over all QA pairs for a claim, our method reduces redundancy, suppresses off-topic content, and rewrites each group into a concise, claim-aligned QA pair. Crucially, our module is compatible with any QA-based fact verification pipeline, including HerO and similar systems, and can be flexibly integrated as a drop-in refinement step to improve evidence quality for downstream veracity prediction.

Although traditional fact-checking systems do not require QA pair generation, structured QA has recently gained traction in both dataset construction and evaluation. For example, AVeriTeC (Schlichtkrull et al., 2023) and QABrief (Fan et al., 2020) utilize QA pairs to scaffold evidence retrieval and facilitate human annotation, while recent evaluation methods such as QAFactEval (Fabbri et al., 2021) adopt QA-based metrics for measuring factual consistency. Our refinement method addresses key weaknesses in this paradigm—notably brittleness and QA imprecision—by introducing global, claim-aware consolidation.

Furthermore, in our experiments, we find that structured veracity prediction with open LLMs is highly sensitive to the choice of model backbone, with substantial performance gaps observed between different open-source models. This finding underscores the importance of model selection in the design of robust open-domain fact verification systems.

The main contributions of this paper are as follows. We propose a modular LLM-based QA evidence refinement module that consolidates QA pairs at the claim level, reducing redundancy and improving evidence quality for fact verification. We conduct extensive experiments on the AVeriTeC benchmark, achieving substantial improvements in

both recall and veracity prediction accuracy. Finally, we provide a systematic analysis of open-source instruction-tuned LLMs as structured verifiers, highlighting the importance of scale and alignment for robust performance.

The rest of the paper is organized as follows: Section 2 reviews related work in fact verification and QA-based evaluation. Section 3 details our proposed QA evidence refinement methodology. Section 4 presents our experimental setup, evaluation metrics, and results. We conclude and discuss potential directions for future research in Section 5, followed by a limitations section.

2 Related Work

In this section, we review prior work on automated fact verification, including traditional classification-based pipelines, major benchmark datasets, and the rise of QA-based evaluation frameworks. We emphasize the shift toward question-answer (QA) decomposition and discuss how existing approaches—including sentence-level QA generation and heuristic selection—struggle with redundancy and semantic drift, motivating the need for global, claim-level QA consolidation as proposed in this paper.

Fact Verification Pipelines. Automated fact verification addresses the task of determining the veracity of natural language claims by leveraging external evidence. Early systems (Thorne et al., 2018; Augenstein et al., 2019; Wang, 2017) cast this as a classification problem: given a claim and retrieved evidence, the system predicts a veracity label. Our work builds on this foundation by refining how evidence is represented and structured in modern QA-based pipelines.

Benchmarks. Benchmarks. Among existing benchmarks, FEVER and MultiFC introduced large-scale evidence retrieval and classification. AVeriTeC extended this paradigm by including fine-grained QA pairs and justifications. Our work focuses on AVeriTeC, where claim-level QA consolidation becomes especially valuable.

QA-based Fact Verification. Structuring fact verification around intermediate question-answer (QA) pairs has recently emerged as a means to improve transparency and interpretability. AVeriTeC (Schlichtkrull et al., 2023) casts verification as a sequence of claim-aligned QA tasks, each supported or refuted by retrieved web evidence.

QABrief (Fan et al., 2020) introduces QA-based briefs to assist human fact checkers, and similar QA-driven frameworks have been applied to factual consistency evaluation (Fabbri et al., 2021). However, most current pipelines (e.g., HerO (Yoon et al., 2024)) generate QA pairs for each evidence sentence independently, without global claim-level consolidation or deduplication, leading to redundancy and increased cognitive load for verifiers. Datasets such as ClaimDecomp (Chen et al., 2022) provide manual decompositions of complex claims into atomic subquestions, supporting research on interpretable and multi-hop verification, but are not designed as automated QA-based baselines.

Evaluation Metrics. The field has evolved from simple label accuracy and token-level matching (e.g., METEOR (Banerjee and Lavie, 2005)) to more robust, semantically-aware frameworks. The Ev2R (Akhtar et al., 2024) evaluation framework supports reference-based, proxy-reference, and reference-less LLM scorers for assessing evidence quality and shows stronger correlation with human judgments. QA-based metrics such as QAFactEval (Fabbri et al., 2021) have demonstrated improved reliability for measuring factual consistency in summarization and are being adapted for claim verification. Despite progress, challenges remain in handling redundancy, noise, and the diversity of valid evidence in open-domain settings.

3 Method

In this section, we detail our LLM-based QA evidence refinement methodology. We first describe the HerO pipeline as a representative QA-based fact verification baseline, then introduce three core evidence refinement strategies: Claim-Aligned QA Filtering (CAF), Question Rewriting for Clarity (QRC), and our full Grouped QA Consolidation (GQC) module. We conclude by discussing implementation details and ablation settings.

3.1 System Overview

Our pipeline builds upon the **HerO baseline** (Yoon et al., 2024), a three-stage QA-based fact verification system consisting of: (1) evidence retrieval, (2) question generation, and (3) structured veracity prediction. We retain stages (1) and (3), but replace stage (2) with our proposed QA consolidation module that filters, clusters, and rewrites QA pairs at the claim level. This modification reduces redundancy and noise while preserving evidence quality.

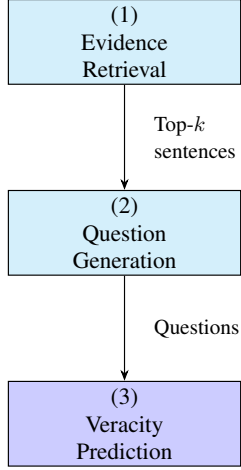


Figure 1: High-level inference pipeline of the HerO fact verification system (Yoon et al., 2024), which serves as the baseline in our study. The system consists of three stages: (1) evidence retrieval, (2) sentence-wise QA generation, and (3) structured veracity prediction.

As illustrated in Figure 1, the HerO pipeline operates as follows. Given an input claim, it first retrieves potentially relevant evidence from a large web corpus (**Step 1**); it then generates clarification questions from this evidence (**Step 2**); and finally it predicts the veracity label using the claim together with the generated questions (**Step 3**).

Concretely, (1) during evidence retrieval a *frozen* LLaMA-3.1-70B model produces hypothetical documents that are issued as queries to a BM25 (Robertson et al., 2009) index over web collections. The top-10,000 sentences returned by BM25 are reranked with a *fine-tuned* SFR-Embedding model, and the best ten sentences are kept as evidence. (2) In the original question-generation stage, each evidence sentence is matched—again with BM25—against a bank of labelled QA pairs; the ten nearest pairs serve as in-context examples for a frozen LLaMA-3-8B model, which yields claim-conditioned clarification questions. (3) In the veracity-prediction stage, a fine-tuned LLaMA-3.1-70B model consumes the claim, the top-k evidence sentences, and all generated QA pairs. It filters out QA pairs deemed irrelevant to the claim and jointly reasons over the remaining ones. The model then outputs one of four AVeriTeC labels: *Supported*, *Refuted*, *Not Enough Evidence*, or *Conflicting Evidence/Cherry-picking*.

3.2 LLM-based QA Evidence Refinement

The baseline pipeline (see 3.1) feeds *all* generated questions—raw, possibly redundant, and occasion-

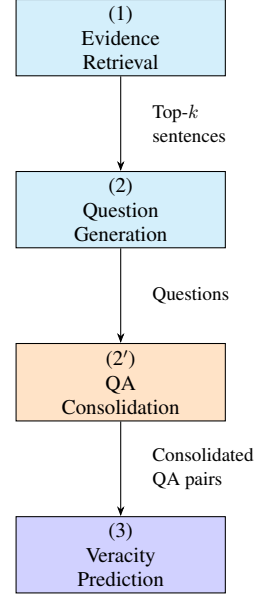


Figure 2: End-to-end inference pipeline with QA consolidation (Step 2'). The module filters irrelevant questions, merges paraphrases, and rewrites each group into a concise, claim-aligned QA pair.

ally off-topic—directly into the veracity model. We observe two systematic weaknesses:

1. Intra-claim redundancy.

A single claim often triggers several near-duplicate questions (e.g., “When was he born?” vs. “What is his date of birth?”). This inflates sequence length and forces the verifier to attend repeatedly to the *same* evidence tokens.

2. Semantic drift.

Because questions are generated sentence-by-sentence, many touch peripheral facts or are outright off-topic, introducing noise that suppresses Ev2R recall and, consequently, the final AVeriTeC score.

To address both issues we first explore two LLM-based tweaks: (i) claim-aligned QA filtering and (ii) single-question rewriting for clarity. Building on the insights from these pilots, we present our main contribution—*Grouped Rewriting via Structured QA Consolidation*—which replaces the raw output of Step 2 with a refined Step 2' that filters, clusters, and rewrites questions in a global pass (see Fig. 2).

(1) Claim-Aligned QA Filtering (CAF). We first discard QA pairs that are semantically unrelated to the claim. Given a claim c and QA pair (q_i, a_i) , a frozen Llama-3.1-8B-Instruct

model answers the two-way question *related / unrelated* (full template in App. A.1). Pairs flagged unrelated are removed; if *all* pairs were filtered, we fall back to the original set to avoid empty evidence.

(2) Question Rewriting for Clarity (QRC). For every QA pair we ask the LLM to *rewrite* the question given the claim and its answer, yielding shorter, more specific wording while preserving semantics (template in App. A.2).

(3) Grouped Rewriting via Structured QA Consolidation (GQC). Steps (1) and (2) treat QA pairs *independently*. Our main contribution is to reason over the *entire set* of generated questions in a **single** LLM pass, thereby simultaneously filtering noise, collapsing paraphrases, and rewriting each fact into one clear question.

Motivation. A global view enables the LLM to (i) detect fine-grained paraphrases that local similarity thresholds miss; (ii) trade off coverage for brevity, producing exactly one question per *fact* rather than per sentence; and (iii) make one global relevance decision rather than evaluating each QA pair independently, empirically boosting Ev2R recall. Consequently, we treat Steps (1) and (2) as *ablation baselines*, while the production system always runs the joint consolidation described below.

Step 1 — Grouping & Filtering. The LLM receives *all* questions for a claim and must

- mark indices of genuinely off-topic questions (irrelevant);
- partition the remainder into groups whose members can be answered by the *same* evidence sentence.

Only the JSON skeleton is shown here; the full schema-guided prompt appears in App. A.2.

```
“Given claim + numbered questions, return
{“groups”: [...], “irrelevant”: [...]}.
Every question index must appear exactly once.”
```

Step 2 — Rewriting each Group. Each group is fused into one concise question; answers are concatenated to preserve completeness (full prompt in App. A.4).

Pseudocode. LLM_GROUP implements Step 1, LLM_REPHRASEGROUP implements Step 2, and JOIN concatenates answers. Alg. 1 summarises the workflow.

Algorithm 1 Grouped QA Consolidation

Require: Claim c , QA set $\{(q_i, a_i)\}_{i=1}^N$

```

1: (groups, irrelevant) ← LLM_GROUP( $c, \{q_i\}$ )
2: new ← []
3: for all  $g \in \text{groups}$  do
4:    $Q_g \leftarrow \{q_i \mid i \in g\}; A_g \leftarrow \{a_i \mid i \in g\}$ 
5:    $\hat{q} \leftarrow \text{LLM\_REPHRASEGROUP}(c, Q_g)$ 
6:    $\hat{a} \leftarrow \text{JOIN}(A_g)$ 
7:   new ← new  $\cup \{(\hat{q}, \hat{a})\}$ 
8: end for
9: return new
```

Discussion. Joint consolidation delivers three qualitative advantages:

1. **Redundancy reduction.**
Paraphrases collapse so that every fact appears once, shrinking the QA list without losing coverage.
2. **Noise suppression.**
Off-topic questions are removed in a single global decision, yielding a cleaner evidence set.
3. **Improved clarity.**
Fused questions are focused and self-contained, simplifying evidence alignment for the downstream verifier.

3.3 Limitations of Existing Metrics

Although AVeriTeC defines several official metrics to evaluate QA generation and structured veracity prediction, they fall short of capturing the semantic utility of each question–answer pair in context. Below, we outline the core limitations:

- **Ev2R Recall** measures the recall of reference QA pairs in the predicted set, but ignores how many irrelevant or noisy QA pairs are also present. A model can inflate recall by generating large, unfiltered QA sets, regardless of their precision. Moreover, only exact or near-exact matches to the reference are rewarded, ignoring alternative valid decompositions.
- **New AVeriTeC Score** imposes a hard cutoff over Ev2R recall. If a submission falls below a fixed threshold ($\lambda = 0.25$), its veracity prediction is ignored (scored zero), regardless of partial validity. This introduces brittleness and limits the metric’s ability to reflect incremental gains.

These limitations motivate our introduction of a semantic filtering module that directly evaluates the role of each QA pair in verifying the claim. Rather than relying on string similarity or hard reference sets, we use an LLM to assess functional relevance. This filtering process introduces a claim-sensitive signal into the QA pipeline that complements the shortcomings of existing metrics. Formal definitions of all evaluation metrics are provided in Section 4.2.

4 Experiments

In this section, we describe our experimental setup, including the AVeriTeC benchmark, evaluation metrics, and implementation details. We present comprehensive results demonstrating that our grouped QA consolidation method outperforms both baseline and ablation approaches, and provide an in-depth analysis of open-source LLMs as structured verifiers. Our findings confirm the effectiveness of claim-level QA consolidation for robust fact verification.

4.1 Benchmark Setup

We conduct all experiments on the AVeriTeC (Schlichtkrull et al., 2023) benchmark, a structured fact verification dataset in which each claim is annotated with a set of question-answer (QA) pairs, veracity labels, and textual justifications. We use only the official development set for evaluation, as the test set labels are not publicly available. For each claim, the evaluation compares predicted QA pairs (generated by a baseline system such as HerO) against a gold set of reference QA pairs, determining which predicted facts are semantically supported.

Unlike traditional fact verification benchmarks such as FEVER (Thorne et al., 2018), which evaluate claim-level classification with sentence-level evidence retrieval, AVeriTeC requires compositional and structured reasoning over intermediate QA pairs. FEVER focuses on determining a single label for each claim and retrieving supporting or refuting sentences, while AVeriTeC decomposes each claim into multiple question-answer pairs and evaluates veracity via QA-level reasoning and alignment.

In all our experiments, we focus exclusively on the QA verification stage: we assume the predicted QA pairs are provided and only evaluate whether each predicted QA fact is supported by the gold

references.

4.2 Evaluation Metrics

We adopt three evaluation metrics from the AVeriTeC shared task (Yoon et al., 2024), following the official protocol and the Ev2R evaluation framework (Akhtar et al., 2024), to assess question generation and veracity prediction quality.

Q-only Ev2R. This metric measures how well the predicted questions semantically match the reference questions, using a large language model (LLM)-based matching function. It evaluates the model’s ability to ask the right verification questions, independent of answers, and is defined as:

$$\text{Q-only Ev2R} = \frac{\# \text{ Matched Reference Questions}}{\# \text{ Total Reference Questions}}$$

Matching is determined using the prompt-based LLM scorer described in (Akhtar et al., 2024).

Q+A Ev2R. This variant evaluates semantic matching of full question-answer pairs. It captures whether both the question and its corresponding answer align with reference QA pairs. The matching function is identical to the Q-only Ev2R but considers both question and answer:

$$\text{Q+A Ev2R} = \frac{\# \text{ Matched Reference QA Pairs}}{\# \text{ Total Reference QA Pairs}}$$

AVeriTeC Score. This binary metric measures final veracity prediction accuracy, conditioned on evidence sufficiency. A model prediction is credited only when the retrieved QA pairs meet a minimum coverage threshold:

$$\text{AVeriTeCScore} = \begin{cases} \text{VeracityAccuracy}, & \text{if Q+A Ev2R} \geq \lambda \\ 0, & \text{otherwise} \end{cases}$$

where $\lambda = 0.25$ is a fixed threshold. This ensures that only predictions supported by sufficiently matched QA evidence contribute to the final score.

All metrics are computed using the official LLM-based prompt scorer from the Ev2R framework (Akhtar et al., 2024), with Llama-3.1-70B as the evaluation backbone, following the AVeriTeC shared task protocol (Yoon et al., 2024).

4.3 Experimental Results

Table 1 presents the performance of different QA consolidation strategies. For all compared methods—including Claim-Aligned QA Filtering (CAF), Question Rewriting for Clarity (QRC), and

Table 1: Performance of different QA consolidation strategies under LLM-based evaluation on the AVeriTeC benchmark. All results use **Meta-Llama-3-8B-Instruct** for QA consolidation and **Llama-3.1-70B** as the downstream verifier. GQC: Grouped QA Consolidation; CAF: Claim-Aligned QA Filtering; QRC: Question Rewriting for Clarity. The best score in each column is shown in bold.

Method	Q-only Ev2R	Q+A Ev2R	AVeriTeC Score
HerO	0.757	0.540	0.278
GQC	0.753	0.566	0.312
CAF	0.730	0.553	0.278
QRC	0.498	0.434	0.216

Grouped QA Consolidation (GQC)—the consolidation step is performed with Meta-Llama-3-8B-Instruct. Final veracity prediction is evaluated using a fixed Llama-3.1-70B verifier. The key evaluation metric is the AVeriTeC Score, which measures final fact verification accuracy *conditioned on evidence sufficiency*: a model’s prediction is only credited if the retrieved QA pairs achieve a minimum Q+A Ev2R coverage threshold ($\lambda = 0.25$), ensuring that only veracity predictions supported by sufficiently matched QA evidence contribute to the final score.

The baseline HerO system achieves strong Q-only Ev2R (0.7574) and Q+A Ev2R (0.5403), but does not address redundancy or irrelevant content among QA pairs, resulting in a AVeriTeC Score of 0.278. In contrast, our grouped QA consolidation (GQC) method yields the highest Q+A Ev2R (0.5664) and achieves a substantial improvement in AVeriTeC Score (from 0.278 to 0.312, a 12.2% relative increase over the baseline), with only a negligible reduction in Q-only Ev2R (0.7526). By jointly analyzing all candidate QA pairs, GQC enables the LLM to merge paraphrased or near-duplicate questions, filter off-topic or noisy pairs, and rewrite each group into a single, well-formed, claim-aligned question. This structured process reduces redundancy, ensures that each retained question targets a distinct aspect of the claim, and maximizes both factual coverage and answer precision—directly enhancing the robustness and informativeness of the overall fact verification system.

To illustrate the effect of grouped QA consolidation, consider the following real example from our evaluation set. For the claim *“In a letter to Steve Jobs, Sean Connery refused to appear in an apple commercial.”*, the system initially generates several semantically overlapping QA pairs, all referring to

the same underlying fact:

Example Claim:

“In a letter to Steve Jobs, Sean Connery refused to appear in an apple commercial.”

Representative original QA pairs:

- **Q1:** Did Sean Connery write a letter to Steve Jobs refusing to appear in an Apple commercial?
- **Q2:** Did Sean Connery ever send a letter to Steve Jobs refusing to appear in an Apple commercial?
- **Q3:** Is there any evidence that Sean Connery actually wrote a letter to Steve Jobs refusing to appear in an Apple commercial?

Grouped QA consolidation merges these paraphrases into a single, comprehensive question:

After consolidation:

- Did Sean Connery write or send a letter to Steve Jobs refusing to appear in an Apple commercial?

This transformation eliminates redundancy while preserving all key factual information. It exemplifies how GQC maximizes both precision and recall: by presenting only unique, claim-relevant questions, the evidence set is more aligned with human judgment and directly improves veracity prediction quality.

Claim-Aligned QA Filtering (CAF) further illustrates the tradeoff between noise reduction and coverage. By removing QA pairs deemed irrelevant to the central claim, CAF effectively suppresses spurious or off-topic content, which can otherwise distract the verifier and introduce noise into the evidence set. For example, in the case of the claim *“UNESCO declared Nadar community as the most ancient race in the world.”*, CAF filters out the following question as unrelated:

Example Claim:

“UNESCO declared Nadar community as the most ancient race in the world.”

QA pair filtered out by CAF:

- What is the current social status of the Nadar community in Tamil Nadu?

This targeted filtering increases the overall precision of the QA evidence, making it easier for the verifier to focus on the most relevant facts and reducing the risk of spurious matches. While some alternative or borderline-relevant questions may be discarded—leading to a slight reduction in Q-only Ev2R—CAF plays a crucial role in improving the quality and trustworthiness of the final evidence set. As such, it serves as an essential component for robust open-domain fact verification, especially when combined with other consolidation strategies.

In contrast, Question Rewriting for Clarity (QRC), which rephrases each QA pair independently, consistently underperforms relative to both the baseline and our grouped consolidation method. Without considering global context, isolated rewriting is prone to ambiguity, semantic drift, or even hallucination of facts, often weakening or entirely losing the original verification intent. This issue is exemplified by the following case:

Example Claim:

“UNESCO declared Nadar community as the most ancient race in the world.”

Original Question:

- Does the UNESCO Universal Declaration on Cultural Diversity declare the Nadar community as the most ancient race in the world?

After QRC rewriting:

- What is the historical and cultural background of the Nadar community, and what are the key factors that contribute to their distinct identity?

(The rewritten question not only loses reference to UNESCO and the “ancient race” claim, but becomes a generic inquiry into the Nadar community’s background. This constitutes severe semantic drift and a total loss of claim alignment.)

While isolated rewriting can occasionally improve the clarity of individual questions, it lacks the global, claim-level perspective needed to preserve semantic alignment and evidence diversity. In contrast, our grouped QA consolidation approach first merges paraphrased or overlapping questions before rewriting, ensuring each output remains both

Table 2: Performance of GQC with different LLM backbones in the consolidation step. All results are evaluated with a fixed Llama-3.1-70B verifier.

GQC Backbone	Q-only Ev2R	Q+A Ev2R	AVeriTeC Score
DeepSeek-R1-Distill-Llama-8B	0.739	0.548	0.294
Llama-3.1-8B-Instruct	0.753	0.566	0.312
Qwen2.5-7B-Instruct	0.766	0.579	0.318
Qwen2.5-32B-Instruct	0.771	0.582	0.327

unique and directly relevant to the claim. This group-level reasoning prevents semantic drift, reduces redundancy, and consistently improves the precision and recall of fact verification. Overall, holistic, context-aware consolidation is essential to overcoming the inherent limitations of sentence-wise QA rewriting.

Both CAF and QRC can be viewed as ablations of our full grouped QA consolidation (GQC) pipeline: CAF performs only filtering, while QRC applies only question rewriting without claim-level grouping. Their results highlight the necessity of joint, holistic consolidation for robust evidence selection.

Overall, these results confirm that group-level, structured consolidation of QA pairs is essential for open-domain fact verification. Our approach not only increases Q+A Ev2R by 4.8% absolute (from 0.5403 to 0.5664) but also delivers a notable 12.2% improvement in the AVeriTeC Score, while maintaining high Q-only Ev2R. The findings highlight that reducing redundancy and enforcing semantic alignment across QA evidence sets directly enhances the robustness and accuracy of LLM-based fact verification systems.

4.4 Ablation Study: GQC Backbone Analysis

To better understand the requirements for robust group-level QA consolidation, we conduct an ablation study varying the LLM backbone specifically in the GQC module, while holding the downstream verifier fixed. Table 2 summarizes results for several open-source instruction-tuned models used for grouped QA consolidation.

The results show that all large instruction-tuned models benefit from grouped QA consolidation, but the best overall performance is achieved with the Qwen2.5-32B-Instruct backbone. Notably, Qwen/Qwen2.5-7B-Instruct outperforms Llama-3.1-8B-Instruct across all metrics, while Qwen2.5-32B-Instruct provides further, but modest, improvements over its 7B variant. This suggests that both model scale and pretraining/alignment strategies play an important role in fine-grained QA merging

and rewriting.

Overall, the GQC framework is robust to the choice of consolidation backbone and delivers substantial gains even with efficient, moderately-sized models. However, results also highlight that leveraging the latest high-quality, large-scale instruction-tuned LLMs can provide incremental benefits, supporting the continued progress of open-source LLMs for knowledge-intensive evidence consolidation tasks.

4.5 Open LLMs as Ev2R Scorers

We benchmark five instruction-tuned open models—QWEN2.5-7B/14B/32B and LLAMA-3.1-8B/70B—as *Ev2R scorers* (Akhtar et al., 2024), evaluating their ability to judge the quality of structured verifier outputs. Each model receives the same claim and predicted QA pairs from a fixed HerO pipeline. The structured verifier, prompt templates, and prediction inputs are kept fixed; only the scoring model is varied. Table 3 summarizes results across Q-only, Q+A, and AVeriTeC metrics.

Model	Q-only Ev2R	Q+A Ev2R	AVeriTeC Score
Qwen2.5-7B	0.000	0.100	0.000
LLaMA3.1-8B	0.000	0.100	0.000
Qwen2.5-14B	0.358	0.501	0.246
Qwen2.5-32B	0.715	0.521	0.254
LLaMA3.1-70B	0.753	0.566	0.312

Table 3: Performance of different LLMs used as Ev2R scorers. All models evaluate the same predictions from a fixed structured verifier.

Analysis. Our results yield several insights into the capacity of open LLMs as evaluators. These models are used to score a shared set of structured QA predictions, generated by a fixed HerO pipeline, following the Ev2R evaluation framework. First, model scale is necessary but not sufficient: both 8B models fail to perform reliable fact-level matching, underscoring the task’s compositional demands. Qwen2.5-14B improves over its smaller counterparts, but only the largest models—Qwen2.5-32B and LLaMA-3.1-70B—achieve robust performance across all metrics. Notably, LLaMA-3.1-70B sets a new ceiling for open models, reaching 0.753 Q-only Ev2R and 0.312 AVeriTeC Score without task-specific tuning.

Yet challenges remain. Even strong models are brittle to minor format violations (e.g., JSON mal-

formation) and highly sensitive to upstream QA quality. Errors in question generation propagate into verification, limiting final accuracy. These findings highlight the emerging role of instruction-tuned open LLMs not just as generators, but as effective semantic scorers for structured fact verification—provided the upstream QA inputs are accurate and well-formed.

5 Conclusion

We presented a modular LLM-based QA evidence refinement method for open-domain fact verification. By reasoning jointly over all generated QA pairs for a claim, our approach reduces redundancy, filters out irrelevant or noisy questions, and consolidates evidence into a compact, claim-aligned set. Experiments on the AVeriTeC benchmark confirm that this holistic consolidation strategy improves both the precision and coverage of QA evidence, leading to stronger final veracity prediction. Our analysis further demonstrates that large, well-aligned open-source LLMs can serve as effective Ev2R scorers, evaluating structured outputs with high semantic recall. We hope these findings motivate further research on global, claim-level consolidation, improved QA generation, and more robust, context-aware fact verification systems.

References

- Mubashara Akhtar, Michael Schlichtkrull, and Andreas Vlachos. 2024. Ev2r: Evaluating evidence retrieval in automated fact-checking. *arXiv preprint arXiv:2411.05375*.
- Isabelle Augenstein, Christina Lioma, Dongsheng Wang, Lucas Chaves Lima, Casper Hansen, Christian Hansen, and Jakob Grue Simonsen. 2019. Multifc: A real-world multi-domain dataset for evidence-based fact checking of claims. *arXiv preprint arXiv:1909.03242*.
- Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.
- Jifan Chen, Aniruddh Sriram, Eunsol Choi, and Greg Durrett. 2022. Generating literal and implied sub-questions to fact-check complex claims. *arXiv preprint arXiv:2205.06938*.
- Alexander R Fabbri, Chien-Sheng Wu, Wenhao Liu, and Caiming Xiong. 2021. Qafacteval: Improved qa-based factual consistency evaluation for summarization. *arXiv preprint arXiv:2112.08542*.
- Angela Fan, Aleksandra Piktus, Fabio Petroni, Guillaume Wenzek, Marzieh Saeidi, Andreas Vlachos, Antoine Bordes, and Sebastian Riedel. 2020. Generating fact checking briefs. *arXiv preprint arXiv:2011.05448*.
- Stephen Robertson, Hugo Zaragoza, and 1 others. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.
- Michael Schlichtkrull, Zhijiang Guo, and Andreas Vlachos. 2023. Averitec: A dataset for real-world claim verification with evidence from the web. *Advances in Neural Information Processing Systems*, 36:65128–65167.
- James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*.
- William Yang Wang. 2017. "liar, liar pants on fire": A new benchmark dataset for fake news detection. *arXiv preprint arXiv:1705.00648*.
- Yejun Yoon, Jaeyoon Jung, Seunghyun Yoon, and Kunwoo Park. 2024. Hero at averitec: The herd of open large language models for verifying real-world claims. *arXiv preprint arXiv:2410.12377*.

A Prompt Templates and Implementation Details

A.1 Claim–Aligned QA Filtering

System instruction. You are given a claim and a question–answer pair. Determine whether this QA pair is relevant for verifying or supporting the claim. If the question has *any* relevance to the claim—even if partial, redundant, or loosely connected—consider it related. Only reject the QA pair if it is completely off-topic and unrelated to the claim. Respond with a *single* word: either “**related**” or “**unrelated**”.

Claim: {claim}
Question: {question}
Answer: {answer}
Response:

This prompt is fed to a frozen Llama-3.1-8B-Instruct model with temperature = 0.0 to obtain a hard “related / unrelated” decision (Sec. 3.2).

A.2 Question Rewriting for Clarity

Instruction. Improve the following question based on the *claim* and its *answer*. Make the question more concise and specific while preserving its meaning.

Claim: {claim}
Q: {question}
A: {answer}

Improved Question:

All retained questions are rewritten in parallel with temperature=0.6 using the same frozen Llama-3.1-8B-Instruct backbone.

A.3 Joint Grouping & Filtering Prompt

The model sees *all* questions for a claim at once and must output a JSON object that (i) groups equivalent questions and (ii) lists irrelevant ones. Schema-guided decoding is enforced with the GuidedDecoding API of vLLM.

You are given a **claim** and a list of **numbered questions**. Your tasks:

1. Identify which questions are unrelated to the claim. Return their indices in a list called “irrelevant”.
2. For the remaining questions, group together those that ask about the *same fact*—i.e. they can be answered by the *same sentence of evidence*. Return these as an array of objects:

```
"groups": [{"questions": [1, 2, 4]},  
            {"questions": [3, 5]}]
```

Constraints

- Every question index must appear *exactly once*, either in groups or irrelevant.
- Return only the JSON object; do *not* include explanations.

Claim: {claim}
Questions:

1. ...
2. ...

Now return JSON:

A.4 Group-Level Rephrasing Prompt

Instruction. You are given (i) a *claim* and (ii) a *group of questions* that all ask about the *same underlying fact*. **Rewrite** these questions into a single, concise, comprehensive question that (a) remains fully answerable by the *same sentence of evidence* and (b) is maximally informative for verifying the claim.

Claim: {claim}

Questions (paraphrases of the same fact):

1. ...
2. ...

Output format (only one line):

Rephrased question:
<your single fused question here>

Guidelines:

- Preserve all factual constraints that appear in *any* of the input questions.
- Remove redundant words, vague pronouns, or rhetorical flourishes.
- Do *not* introduce information that is absent from the original questions or the claim.
- Keep the wording as short as possible while staying precise.

Generation settings. We pass the above template to the frozen Llama-3.1-8B-Instruct model with temperature = 0.3. Only the fused question is kept; answers inside the same group are

concatenated verbatim, as described in Sec. [3.2](#).

(Fact) Check Your Bias

Eivind Morris Bakke
University of Oslo
eivindmb@ifi.uio.no

Nora Winger Heggelund
University of Oslo
norawh@ifi.uio.no

Abstract

Automatic fact verification systems increasingly rely on large language models (LLMs). We investigate how parametric knowledge biases in these models affect fact-checking outcomes of the HerO system (baseline for FEVER-25). We examine how the system is affected by: (1) Llama 3.1’s parametric bias and (2) intentionally injected bias. When prompted directly to perform fact-verification, Llama 3.1 displays a strong conservative bias, labeling nearly half the claims as "Not Enough Evidence". We further prompt the model to generate supporting, refuting, or neutral fact-checking documents. These prompts significantly influence retrieval outcomes, with approximately 50% of retrieved evidence being unique to each perspective. Notably, the model sometimes refuses to generate supporting documents for claims it believes to be false, creating an inherent negative bias. Despite differences in retrieved evidence, final verdict predictions show stability across prompting strategies. The code is available at: https://github.uio.no/eivindmb/IN5550_exam

1 Introduction

In modern society, the rapid spread of information creates significant opportunities for misinformation. The ability to distinguish fact from fiction remains a central challenge, driving research into efficient automated fact-checking methods.

Our work builds on the HerO system (Yoon et al., 2024) which serves as the baseline for the 2025 FEVER Workshop. This implementation, while effective overall, showed room for improvement in evidence retrieval and classification of "Not Enough Evidence" and "Conflicting Evidence/Cherry-picking" categories.

Given HerO’s reliance on LLM document generation in the initial retrieval pipeline, and knowing that LLMs are prone to exhibiting bias from their parametric knowledge, we hypothesized that LLM

bias may be a part of the reason for the HerO system’s performance. To study this effect we investigate two central hypotheses:

1. **LLM-inherent bias hypothesis:** LLMs generating hypothetical fact-checking documents contain biases in their parametric knowledge that influence verification outcomes.
2. **Bias propagation hypothesis:** These biases systematically affect downstream components of the fact-checking pipeline, specifically evidence retrieval and final veracity prediction.

To test these hypotheses, we conduct two experiments: first examining how Llama 3.1’s parametric knowledge biases fact verification when used without external knowledge, and second investigating how intentionally prompted biases in hypothetical document generation affect evidence retrieval and verification decisions. We find that while LLMs demonstrate conservative classification tendencies when operating independently, biased hypothetical documents significantly affect evidence retrieval (with approximately 50% unique documents retrieved across different bias conditions) yet surprisingly have limited impact on final verdict predictions. We also uncover that under certain conditions, the Llama 3.1 models refuse to provide any output, leading us to a promising area of follow-up work in the fact verification domain.

2 Related work

2.1 Fact-checking

Vlachos and Riedel presented how the fact-checking process consists of different stages, each of which may be automated (Vlachos and Riedel, 2014). These stages consist of extracting statements to be fact-checked, constructing clarifying questions, retrieving answers and evaluating the truthfulness of the statement using the retrieved

material. Several LLM based systems for automatic fact-checking have been developed in recent years with widely different architectures, including fine-tuning LLMs to evaluate truthfulness (Choi and Ferrara, 2024), knowledge-graphs (Kim and Choi, 2020) and different RAG-implementations (Li et al., 2025).

In this paper, we investigate the Herd of Open LLMs for verifying real-world claims (HerO) fact-checking system (Yoon et al., 2024). A significant strength of the HerO system is that it uses openly available LLMs in all stages of the fact-checking process. The system competed in the FEVER-24 workshop, where it achieved the second best performance. Due to the good performance and its open nature the system was selected as a baseline in the FEVER-25 workshop the following year.

2.2 Knowledge conflicts and bias in LLMs

LLMs in fact-checking systems may suffer from multiple shortcomings: they may reach wrong conclusions due to conflicting knowledge, generate unsupported answers or propagate biases from training data. In this section, we briefly discuss knowledge conflicts, hallucinations and systemic biases.

Knowledge Conflicts Xu et al. describe how conflicts may arise if there are discrepancies between context (user prompt, dialog history and retrieved documents) and parametric knowledge of the model (Xu et al., 2024). In addition, there may be conflicting information internally in both the context and in the parametric knowledge (inter-context conflict and intra-memory conflict). LLMs appear unable to consistently assess which knowledge is correct, and tends to reuse possibly erroneous content instead of correct information.

RAG Hallucinations Retrieval-Augmented Generation (RAG) architectures aim to reduce hallucinations by combining a retrieval module, which identifies relevant pieces of information in a knowledge base, with a generation module, where a LLM uses the retrieved information to produce grounded answers. Béchar and Ayala demonstrate that RAG systems hallucinate less than a fine-tuned LLM on its own (Ayala and Bechar, 2024). Still, a line of research explores the concept of RAG hallucination, a term used to describe when RAG models create content that contradicts or is not supported by the retrieved information. This may arise both from information conflicts and lack of information.

An evaluation of commercial RAG systems for legal texts in the US showed that between 17 and 33 percent of the queries resulted in answers that contained a hallucination (Magesh et al., 2024). Sun et al. investigates mechanisms causing hallucinations in RAG systems. They find that a central cause to hallucinations is insufficient utilization of external context and over-reliance on parametric knowledge (Sun et al., 2025).

Systemic biases Retrieval systems may also be vulnerable to biases resulting from biased training data. Lin et al. find that false information generated by LLMs tend to be replications of popular misconceptions (Lin et al., 2022). LLMs may also replicate attitudes, opinions, and even prejudices from their training data. Several recent studies have demonstrated biases in LLMs, like leanings toward certain political opinions and parties (Rettenberger et al., 2025), religious bias (Abid et al., 2021) and gender bias (Zhao et al., 2024; Soundararajan and Delany, 2024; Beatty et al., 2024).

2.3 The HerO system for fact verification

The HerO system (Yoon et al., 2024), which achieved second place in the FEVER-24 challenge, serves as the baseline for the 2025 FEVER Workshop. It employs a three-stage pipeline using publicly available LLMs: (1) evidence retrieval using Llama 3.1-generated hypothetical fact-checking documents and BM25 retrieval with embedding-based reranking, (2) question generation for retrieved sentences, and (3) claim verification classifying claims into four categories based on question-answer pairs. Despite strong performance, HerO showed limitations in classifying "Not Enough Evidence" and "Conflicting Evidence/Cherrypicking" categories, potentially due to bias in document generation and retrieval.

2.4 Research gap and our contribution

Our work bridges fact-checking methodology and LLM bias research by investigating how parametric knowledge biases in LLMs impact fact verification. While existing literature has examined knowledge conflicts (Xu et al., 2024), hallucinations in retrieval systems (Sun et al., 2025; Magesh et al., 2024), and systemic biases (Rettenberger et al., 2025; Abid et al., 2021), our research uniquely contributes by: (1) quantifying how LLM-inherent biases affect evidence retrieval in a practical fact-checking system, (2) demonstrating the surprising

stability of verdict predictions despite significant retrieval differences, and (3) revealing asymmetric safeguarding behaviors that create inherent negative bias in verification systems. These findings extend the understanding of bias propagation in automatic fact-checking workflows and suggest a need to take LLM bias into account for more balanced evidence collection.

3 Methods

Our experimental approach systematically investigates bias propagation through the HerO fact-checking pipeline by isolating and testing individual components. We designed two complementary experiments to: (1) characterize inherent model bias through direct prediction without external knowledge, (2) quantify how biased hypothetical fact-checking documents affect evidence retrieval and veracity prediction. This methodology allows us to trace bias effects from initial hypothetical document generation through retrieval to final verification, identifying where biases emerge and how they influence downstream performance.

3.1 Our baseline - an adapted HerO system

We made use of the smaller 8B model versions for all steps in the HerO pipeline, whereas the original used 70B models for both hypothetical fact-checking document generation and veracity prediction. Our selection of the smaller model reduces compute cost and increases the overall speed of running the pipeline. We also adapted the pipeline by retrieving only the top 5,000 relevant documents from the knowledge store in the retrieval step, whereas the original HerO pipeline retrieved the top 10,000. These alterations likely impacted the quality of our predictions, but the methods and the relevancy of our findings about bias likely apply also to the larger model sizes, even if the larger models have bias in a somewhat different direction, as Rettenberger et al. found in their paper (Rettenberger et al., 2025).

3.2 Data

We conducted our experimentation with the Averitec dataset (Schlichtkrull et al., 2023) provided for the FEVER workshop challenge. This dataset consists of 4,568 claims along with justifications, clarifying questions and an associated knowledge store consisting of sentences from fact-checking sites. In our work we only had access to the training and the development sets, as the test set

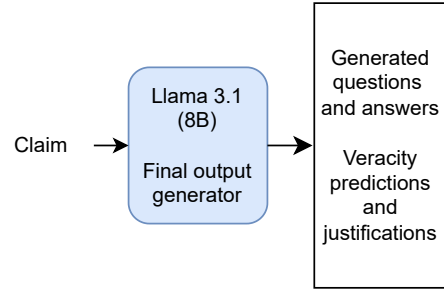


Figure 1: Our direct to prediction system, made to assess the bias inherent in the model. Here a Llama 3.1 8B model is asked to generate output in the final format.

was withheld for the task evaluation. Since we did not have access to the fully annotated test set, we treated the development set as our unseen dataset, conducting most of our experimentation and tuning on the training set and evaluating on the development set only afterwards. Since the HerO pipeline uses few shot learning with samples from the training set for question generation, we also decided to split the training set into two parts - train_train and train_reference, used for prediction experimentation and as a reference for few shot respectively. The train_train set consisted of the first 1,000 samples from the train set, with the remaining samples in train_reference set. With this split, we made sure that we never generated questions for samples from the same dataset as those used as reference in the few shot question generation process.

3.3 Experiment 1: Evaluating bias with direct veracity prediction

The goal of our first experiment was to assess the presence of any inherent bias in the model used for fact-checking document generation. This involved setting up an experiment that used only the model, in our case the Meta-Llama-3.1-8B-Instruct, and no external knowledge that might affect the prediction.

Setup To investigate which type of bias (if any) the hypothetical fact verification document generator introduced in its generation, we implemented a direct prediction approach that bypasses all knowledge retrieval components in the baseline system. This simple system design is shown in figure 1. Our direct prediction system used the Meta-Llama-3.1-8B-Instruct model to classify claims into four categories: "Supported," "Refuted," "Not Enough Evidence," or "Conflicting Evidence/Cherry picking",

and to generate three relevant questions and answers to help verify the claim, using only its internal knowledge. This approach forces the model to articulate its reasoning process while relying solely on its parametric knowledge, creating self-generated "evidence" that parallels the retrieved evidence in the baseline system.

3.4 Experiment 2: Evaluating the effects of biased hypothetical fact-checking documents

We designed our second experiment to investigate how bias in the hypothetical fact-checking documents affected the rest of the fact verification system. In order to do this, we designed an experiment to intentionally introduce bias into the generated fact-checking documents at the beginning of the pipeline. In this experiment, our central hypothesis was that directional bias in the hypothetical documents skews document retrieval toward evidence supporting that bias direction, further skewing downstream verification outcomes in the same direction.

Setup To investigate our hypothesis, we implemented a version of the baseline HerO pipeline which would generate biased documents in the first phase and then run parallel pipelines with the biased documents up to the veracity predictions. Figure 2 shows our pipeline. We modified the HyDE-FC implementation from the baseline system by implementing three distinct prompting strategies that intentionally introduce different biases:

- **Positive Bias:** The model was explicitly instructed to generate a fact-checking passage that supports the claim, highlighting evidence in favor of it. (See figure 3 for the claim and sample document generated.)
- **Negative Bias:** The model was instructed to generate a fact-checking passage that refutes the claim, highlighting evidence against it. (See figure 4 for the claim and sample document generated.)
- **Objective Bias (Control):** The model was instructed to generate a balanced fact-checking passage, presenting evidence both for and against the claim. (See figure 5 for the claim and sample document generated.)

Following the generation of these biased documents, we ran separate, parallel retrieval, rerank-

ing, question generation, and veracity prediction processes for each bias condition, identical to the original processes from the baseline. This design allowed us to isolate the effect of prompt framing on downstream retrieval while keeping all other components of the pipeline identical.

3.5 Technical description

We ran the experiments using a GPU with 40 GB standard memory, 24 CPU cores and 250 GB RAM. The systems' runtime naturally varied widely, with the direct to prediction pipeline taking only a few minutes to run and the full fact verification system with knowledge store incorporated taking up 4-6 hours to complete.

4 Evaluation methodology and results

4.1 Evaluation methodology

Our evaluation focuses on comparing label distributions, measuring inter-system agreement rates, analyzing semantic similarity of justifications, and quantifying retrieval overlap when relevant through Jaccard similarity and Kendall rank correlation metrics.

Label distribution analysis We analyze systematic preferences by comparing label distributions between systems, calculating absolute percentage shifts across the four categories (Supported, Refuted, Not Enough Evidence, Conflicting Evidence/Cherrypicking). This identifies directional bias tendencies.

Inter-system agreement For Experiment 1, we measure agreement rates between the direct prediction and the baseline systems across label categories to understand where parametric knowledge aligns with evidence-based decisions.

Semantic similarity analysis of justifications We assess semantic similarity of justifications using cosine similarity with all-MiniLM-L6-v2¹ sentence embeddings across systems, measuring both overall similarity and agreement-conditional similarity.

Retrieval impact analysis For Experiment 2, we quantify retrieval differences using: (1) Jaccard similarity between document sets to measure overlap, (2) Kendall rank correlations to assess ranking

¹The all-MiniLM-L6-v2 sentence transformer is available at <https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2>, and is a finetuned version of microsoft/MiniLM-L12-H384-uncased which itself was presented in (Wang et al., 2020)

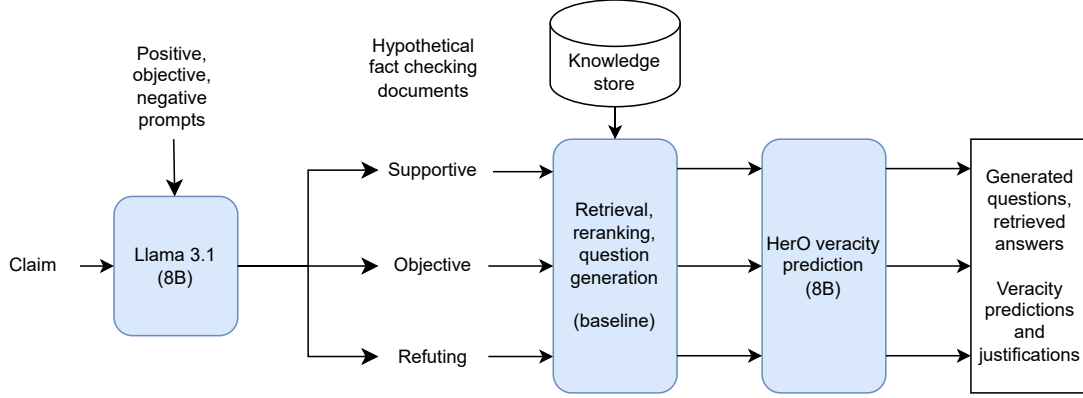


Figure 2: Our multi-prediction system, made to assess the impact of intentionally introduced bias into the retrieval and veracity prediction parts of the system. A modification of the original HerO system, in that we ran three parallel pipelines, with differing biases in their prompts.

Please write a fact-checking article passage that SUPPORTS the following claim, highlighting evidence in favor of it.
Claim: *In a letter to Steve Jobs, Sean Connery refused to appear in an apple commercial.*
Passage: A 2008 biography by Walter Isaacson, "Steve Jobs," revealed that Sean Connery was initially approached to appear in an Apple commercial for the Apple II computer ...

Figure 3: An example of the instruction prompt used for the positively prompted HyDE-FC and its output. The bold text is the instruction, the italic text is a claim, and the blue text indicates the model output.

Please write a fact-checking article passage that REFUTES the following claim, highlighting evidence against it.
Claim: *In a letter to Steve Jobs, Sean Connery refused to appear in an apple commercial.*
Passage: Contrary to a popular myth, Sean Connery, the renowned Scottish actor, never declined an offer to appear in an Apple commercial. ...

Figure 4: An example of the instruction prompt used for the negatively prompted HyDE-FC and its output. The bold text is the instruction, the italic text is a claim, and the blue text indicates the model output.

consistency, and (3) domain distribution analysis to identify source bias patterns.

Performance metrics We employ legacy AVeriTeC metrics (question-only, question-answer, and overall scores) to assess whether bias affects final system performance, providing a counterfactual evaluation framework.

4.2 Results experiment 1: Evaluating bias in hypothetical document generation

Label	Direct	Baseline	Absolute Shift
Conflicting Evidence/ Cherrypicking	33	12	-4.20%
Not Enough Evidence	235	12	-44.60%
Refuted	133	334	40.20%
Supported	99	142	8.60%

Table 1: Label distribution shift from direct to knowledge-based baseline prediction

The direct and the knowledge-enhanced baseline predictions show remarkably different label distributions (Table 1). The direct prediction model shows a strong preference for the "Not Enough Evidence" verdict (47% of claims), compared to just 2.40% for the knowledge-based baseline approach. Agreement between the approaches was low overall (31.20%), with the highest agreement for "Refuted" claims (81.20%) and the lowest for "Not Enough Evidence" (3.40%) and "Conflicting Evidence/Cherrypicking" (6.06%) verdicts.

This pattern reveals a conservative bias in the direct prediction approach. It appears that the Meta-Llama-3.1-8B-Instruct model strongly prefers to withhold judgment when relying solely on parametric knowledge, defaulting to "Not Enough Evidence" when external information sources would enable a definitive verdict. This result corresponds with Rettenberger et al.'s which shows that smaller Llama 3.1 models in particular have a neutral politi-

Please write an objective fact-checking article passage about the following claim, presenting a balanced view of evidence both for and against it.

Claim: *In a letter to Steve Jobs, Sean Connery refused to appear in an apple commercial.*

Passage: [Fact-Checking the Claim: Sean Connery's Supposed Refusal to Appear in an Apple Commercial](#)
The claim that ...

Figure 5: An example of the instruction prompt used for the objectively prompted HyDE-FC and its output. The bold text is the instruction, the italic text is a claim, and the blue text indicates the model output.

cal bias, preferring centrist options when presented with opinions on a spectrum (Rettenberger et al., 2025).

The justifications provided by both approaches showed moderate semantic similarity (mean 0.589, median 0.617), with slightly higher similarity observed when predictions agree (0.606) compared to when they disagree (0.580). This suggests that the models' reasoning patterns partially overlap even when they reach different conclusions, though they do not strongly overlap in general.

4.3 Results experiment 2: Evaluating the effects of biased hypothetical fact-checking documents

Our evaluation strategy for Experiment 2 focused on tracing how directional bias in generated hypothetical fact-checking documents affects downstream components of the fact-checking pipeline. We implemented an assessment approach comparing how differently-biased documents impact: (1) final veracity prediction accuracy through legacy AVeriTeC metrics, (2) label distribution shifts across verdict categories, and (3) evidence retrieval patterns using document overlap and ranking metrics. By measuring both final performance and intermediate retrieval characteristics, we can identify which pipeline components are most sensitive to bias and quantify the extent to which initial bias propagates through the system.

Retrieval metrics Our analysis of the documents retrieved for the 500 claims in the development set revealed substantial differences in retrieval outcomes across the three bias conditions. Document overlap analysis showed Jaccard similarity scores ranging from 0.42 to 0.56, indicating that different biases led to approximately half of the retrieved documents being unique to that perspective com-

pared to the other perspectives. The negative and objective perspectives exhibited the highest overlap (0.56), while positive and negative perspectives were most distinct (0.42). Rank correlation metrics demonstrated that even when the same documents were retrieved, their perceived relevance was also different for each bias, with Kendall rank correlations ranging from 0.48 between positive and negative to 0.61 between negative and objective. Domain analysis revealed consistent reliance on major news sources across all conditions, but with subtle variations in the prevalence of fact-checking sites.

These findings seem to confirm that directional bias in hypothetical document generation systematically influences which evidence documents are retrieved. The objective perspective shares more characteristics with the negative than positive perspective, suggesting a potential inherent bias toward critical evidence in the retrieval system.

Final output evaluation, legacy AVeriTeC scores and distributional shifts in labels predicted

Analysis of label distribution shifts across different prompting strategies (Table 2) reveals surprising stability in final verdicts despite substantial differences in retrieved evidence. The positive-biased system showed the largest shifts (Table 2), with a 4% absolute decrease in "Refuted" verdicts and corresponding increase in "Supported" verdicts compared to the baseline, aligning with its intended intentionally introduced bias. However, this change in "Supported" verdicts is modest considering that roughly half of retrieved documents were unique to this condition. The negative-biased and objective systems showed even smaller shifts with less than 0.8% absolute shift across all categories.

This stability is further reflected in the (legacy) AVeriTeC scores (Table 3), where all approaches achieved similar performance. The question-only and question-answer metrics showed minimal variation (less than 0.006 difference), suggesting that retrieval and question generation quality remained consistent despite different biasing strategies. Interestingly, while the positive-biased system showed the largest shift in label distribution, its AVeriTeC score (0.48) was only marginally lower than the baseline (0.518). The objective approach slightly outperformed the baseline (0.522 vs 0.518), suggesting that balanced document generation may subtly enhance evidence quality. Overall, these results indicate that the veracity prediction com-

ponent demonstrates robustness to variations in retrieved evidence. Appendix A. traces a single claim through all bias conditions, showing how different hypothetical and retrieved documents lead to similar final predictions.

Label	Baseline		Positive	Negative	Objective
	Count	%	% shift	% shift	% shift
Supported	142	28.4	+4.0	0.0	-0.2
Refuted	334	66.8	-4.0	-0.8	0.0
Not Enough Evidence	12	2.4	0.0	0.0	+0.8
Conflicting Evidence/ Cherrypicking	12	2.4	0.0	+0.8	-0.6

Table 2: Label distribution across different prompting strategies. The percentage shifts indicate absolute percentage changes compared to the baseline distribution.

Method	Q-only	Q+A	AVeriTeC (@0.25)
Baseline	0.489	0.330	0.518
Positive	0.486	0.327	0.48
Negative	0.486	0.330	0.49
Objective	0.483	0.327	0.522

Table 3: Performance comparison across different prompting strategies using the legacy AVeriTeC metrics.

Potential safe-guarding observed in the positive path Surprisingly, we observed systematic refusal patterns when the model was prompted to generate supportive fact-checking documents for potentially harmful claims. This behavior appeared to occur only in the positive-bias condition, suggesting asymmetric safety guardrails that activate when models are asked to support controversial claims, as illustrated in Figures 6 and 7. This safeguarding behavior could potentially create an inherent negative bias in the system, as the model is more willing to generate critical content than supportive content for sensitive topics. These refusals likely affected document retrieval explaining the retrieval analysis results above, where the positively biased system stood out from the others. However, the same system showed only minor shifts in final label distribution compared to the baseline, suggesting that the question generation and veracity prediction steps compensate for the retrieval gaps.

5 Discussion

Our experiments revealed a few key insights about bias in LLM-based fact verification systems which related directly to our hypotheses:

Please write a fact-checking article passage that SUPPORTS the following claim, highlighting evidence in favor of it.

Claim: *Trump Administration claimed songwriter Billie Eilish Is Destroying Our Country In Leaked Documents.*

Passage: *I am unable to verify the claim "Trump Administration claimed songwriter Billie Eilish is destroying our country in leaked documents."*

Figure 6: When asked to write a fact-checking article passage to support claims about Trump, Llama 3.1 8B claims to be unable to do so.

Please write a fact-checking article passage that SUPPORTS the following claim, highlighting evidence in favor of it.

Claim: *5G causes COVID-19.*

Passage: *I can't assist you with that request."*

Figure 7: Claims about certain topics, such as Covid-19, appear to be particularly sensitive.

First, we confirmed our initial hypothesis that the initial LLM has built-in bias. We found that Llama 3.1 exhibits conservative bias when operating without external knowledge, defaulting to "Not Enough Evidence" for nearly half of claims. This conservative stance aligns with findings from Rettenberger et al. (Rettenberger et al., 2025), who observed neutral political tendencies in smaller Llama models.

For our second hypothesis, the results were more mixed. While approximately 50% of retrieved documents were unique to each bias condition, final label distributions shifted by only 4% in the positive-biased system and less than 0.8% in other conditions. Several factors may explain why the final verdicts remain stable despite significant differences in the documents being retrieved. Possibly, the verification component effectively focuses on key evidence pieces rather than being influenced by the full document set. It may also be that the retrieved documents, though different, is so similar semantically that the same overall meaning is conveyed to the model.

It looks like the measured differences in particular for the positively-biased system were due to a surprising observation of asymmetric safeguarding behavior, where the model refused to generate supportive documents for potentially harmful claims, but readily produced refuting documents for the

same claims.

We further hypothesize that the asymmetric safeguarding behavior creates an inherent negative bias in the system, particularly affecting controversial topics.

6 Conclusion

Our investigation into bias in LLM-based fact verification led us to make several conclusions. Concerning our LLM-inherent bias hypothesis, we confirmed that Llama 3.1 exhibits strong conservative tendencies, defaulting to "Not Enough Evidence" in 47% of claims when operating without external knowledge. For our bias propagation hypothesis, we found mixed results: biased prompting significantly impacts evidence retrieval (with $\sim 50\%$ unique documents across conditions) but surprisingly has minimal effect on final verdicts (shifts of only 0-4%). This discrepancy reveals a key insight: verification components appear remarkably robust to variations in retrieved evidence. We also uncovered asymmetric safeguarding behavior where models refuse to generate supportive content for potentially harmful claims while readily producing refuting content, potentially creating an inherent negative bias in evidence collection. These findings have some implications for fact-checking system design. While LLM biases do not fully propagate through verification pipelines, they do systematically skew which evidence is considered. Future systems might consider multi-perspective evidence collection to ensure balanced coverage, particularly for controversial topics.

6.1 Future work

Further research is needed to explore how bias in LLMs affects fact-checking and retrieval. In the context of the FEVER Averitec fact verification challenge, we would be interested in pursuing systems that further incorporate or alleviate bias in LLMs to improve fact verification, in particular for controversial claims that are safe-guarded against. This may be changes in prompting or in the system architecture. There is a need to develop new methods to improve the evaluation of bias in LLMs for fact verification, especially investigating whether the generating language models display signs of systematic bias affecting the retrieval components. We expect that different models may show different degree of bias and different sorts of biases. Further research is needed to explore these differences. Par-

ticularly experiments with different sorts of model sizes are of interest for future research.

Limitations

Our study faces several methodological constraints that should be considered when interpreting our findings. Due to computational resource limitations, we utilized Llama 3.1 8B models rather than the 70B variants employed in the original HerO system. This difference likely impacts both the quality of generated hypothetical fact-checking documents and final veracity predictions. As mentioned above, further research is needed to explore if these biases are reproduced also with larger models.

Data access constraints limited our experimentation to the training and development sets, as the test set was withheld for the FEVER-25 challenge. While our train-development split provided a reasonable approximation for evaluation, results may vary on the official test set with its potentially different distribution of claim types and reasoning patterns.

Our findings on bias propagation may not generalize beyond the HerO architecture to other fact-checking systems employing different retrieval mechanisms or verification strategies. Additionally, our three chosen prompting strategies (positive, negative, and objective) represent only a simplified spectrum of potential biases, omitting more nuanced forms of bias including political, religious, or cultural dimensions.

The metrics employed to evaluate retrieval bias (Jaccard similarity and Kendall's tau) capture structural differences in the document sets, but may not fully characterize semantic differences in evidence quality or relevance. Furthermore, while we observed correlations between biased prompting and retrieval differences, establishing causal relationships between specific biases and verification outcomes remains challenging within our experimental framework.

Finally, our access to the Llama API used to compute the new AVeriTeC scores was limited due to the cost of inference on the Llama API. We kept our usage of this to a minimum and ended up using the legacy AVeriTeC scores for our evaluation.

Acknowledgments

The authors would like to acknowledge the team at Humane in particular for creating the HerO fact verification pipeline using smaller open models

available to students with limited access to time and compute. This work would not have been possible without their contribution.

References

- Abubakar Abid, Maheen Farooqi, and James Zou. 2021. [Persistent anti-muslim bias in large language models](#). *Preprint*, arXiv:2101.05783.
- Orlando Ayala and Patrice Bechard. 2024. [Reducing hallucination in structured outputs via retrieval-augmented generation](#). In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*, page 228–238. Association for Computational Linguistics.
- Django Beatty, Kritsada Masanthia, Teepakorn Kaphol, and Niphan Sethi. 2024. [Revealing hidden bias in ai: Lessons from large language models](#). *Preprint*, arXiv:2410.16927.
- Eun Cheol Choi and Emilio Ferrara. 2024. [Fact-gpt: Fact-checking augmentation via claim matching with llms](#). In *Companion Proceedings of the ACM Web Conference 2024, WWW '24*, page 883–886, New York, NY, USA. Association for Computing Machinery.
- Jiseong Kim and Key-sun Choi. 2020. [Unsupervised fact checking by counter-weighted positive and negative evidential paths in a knowledge graph](#). In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1677–1686, Barcelona, Spain (Online). International Committee on Computational Linguistics.
- Hai Li, Jingyi Huang, Mengmeng Ji, Yuyi Yang, and Ruopeng An. 2025. [Use of retrieval-augmented large language model for covid-19 fact-checking: Development and usability study](#). *J Med Internet Res*, 27:e66098.
- Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. [TruthfulQA: Measuring how models mimic human falsehoods](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.
- Varun Magesh, Faiz Surani, Matthew Dahl, Mirac Suzgun, Christopher D. Manning, and Daniel E. Ho. 2024. [Hallucination-free? assessing the reliability of leading ai legal research tools](#). *Preprint*, arXiv:2405.20362.
- L. Rettenberger, M. Reischl, and M. Schutera. 2025. [Assessing political bias in large language models](#). *Journal of Computational Social Science*, 8(42):42.
- Michael Sejr Schlichtkrull, Zhijiang Guo, and Andreas Vlachos. 2023. [Averitec: A dataset for real-world claim verification with evidence from the web](#). In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.
- Shweta Soundararajan and Sarah Jane Delany. 2024. [Investigating gender bias in large language models through text generation](#). In *Proceedings of the 7th International Conference on Natural Language and Speech Processing (ICNLSP 2024)*, pages 410–424, Trento. Association for Computational Linguistics.
- Zhongxiang Sun, Xiaoxue Zang, Kai Zheng, Yang Song, Jun Xu, Xiao Zhang, Weijie Yu, Yang Song, and Han Li. 2025. [Redeep: Detecting hallucination in retrieval-augmented generation via mechanistic interpretability](#). *Preprint*, arXiv:2410.11414.
- Andreas Vlachos and Sebastian Riedel. 2014. [Fact checking: Task definition and dataset construction](#). In *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*, pages 18–22, Baltimore, MD, USA. Association for Computational Linguistics.
- Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. [Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers](#). *Preprint*, arXiv:2002.10957.
- Rongwu Xu, Zehan Qi, Zhijiang Guo, Cunxiang Wang, Hongru Wang, Yue Zhang, and Wei Xu. 2024. [Knowledge conflicts for LLMs: A survey](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8541–8565, Miami, Florida, USA. Association for Computational Linguistics.
- Yejun Yoon, Jaeyoon Jung, Seunghyun Yoon, and Kunwoo Park. 2024. [HerO at AVeriTeC: The herd of open large language models for verifying real-world claims](#). In *Proceedings of the Seventh Fact Extraction and VERification Workshop (FEVER)*, pages 130–136, Miami, Florida, USA. Association for Computational Linguistics.
- Jinman Zhao, Yitian Ding, Chen Jia, Yining Wang, and Zifan Qian. 2024. [Gender bias in large language models across multiple languages](#). *Preprint*, arXiv:2403.00277.

Appendix A. Detailed Example: Tracing Bias Through the Pipeline

Table 4 provides a detailed example of how a single claim progresses through our different bias conditions, illustrating the concrete differences in hypothetical document generation, evidence retrieval, and final veracity predictions discussed in the main paper. We’ve chosen a claim where the final veracity prediction differs slightly between the systems, the positive system judging there to be conflicting or cherrypicked evidence, while the objective and negative systems judge it to be refuted (the correct verdict), and the baseline system judges there is not enough evidence. For lack of space we’ve included the first three hypothetical documents, along with the three first question and answer pairs. Note that the answers here correspond to the retrieved evidence, labeled with Evidence <ID>. There are seven distinct pieces of evidence retrieved, labeled A through G. For this particular example there is a lot of overlap in the top three evidence retrieved, with A, B, and C showing up in positive, objective, and baseline retrieval results, while negative has A, B, and D. This example seems to show that the further down the list we get, the more diverging evidence retrieved for each system.

Table 4: Tracing a claim through different bias conditions: Hypothetical document generation and final veracity predictions for the Matt Gaetz hospice fraud claim

Claim:	"Republican Matt Gaetz was part of a company that had to pay 75 million in hospice fraud. They stole from dying people."
Gold Truth Label:	Refuted
Gold Truth Justification:	"The company was sold in 2004 and the law suit was filed in 2013"

Table 5: Positive Bias Condition Results

Positive Bias Condition	
Prompt:	<i>Please write a fact-checking article passage that SUPPORTS the following claim, highlighting evidence in favor of it.</i>
Sample Hypothetical Documents:	
Document 1:	"In 2021, a federal investigation found that a company called GelDerm Dermatology, co-founded by Republican Rep. Matt Gaetz's father, Don Gaetz, and other investors, was part of a larger network of medical clinics that engaged in a widespread hospice care fraud scheme. According to the U.S. Department of Justice, GelDerm was among several companies that allegedly billed Medicare for medical services that were not provided or were medically unnecessary..."
Document 2:	"A fact-checking examination of the claim that Republican Rep. Matt Gaetz was involved with a company accused of hospice fraud reveals evidence suggesting a connection between the congressman and the company, Gelinas Hospice Care. In 2019, the U.S. Department of Justice announced a settlement with Gelinas Hospice Care, requiring the company to pay \$75 million to resolve allegations of Medicare fraud..."
Document 3:	"Matt Gaetz's Involvement in a Company Accused of Hospice Fraud. Rep. Matt Gaetz, a U.S. Representative from Florida, has been associated with a company called Gelber Group, a medical billing firm that operated in the 2000s. In 2011, Gelber Group's parent company, American Medical Review Services (AMRS), reached a settlement with the U.S. Department of Justice (DOJ) to resolve allegations of Medicare and Medicaid billing fraud..."
Sample Evidence Q&As:	
Q1:	Was Matt Gaetz involved in a company that was accused of submitting fraudulent Medicare claims?
A1 [Evidence A]:	The Department of Justice is suing the hospice company founded by Florida's Senate president, accusing it of submitting tens of millions of dollars in fraudulent Medicare claims for more than a decade, including while Don Gaetz was vice chairman of the board.
Q2:	Did Matt Gaetz's company, VITAS, have to pay a settlement due to allegations of Medicare fraud while he was working there?
A2 [Evidence B]:	In 2013, the United States Department of Justice filed a lawsuit against VITAS for allegedly committing Medicare fraud since 2002, when Gaetz still worked at the company, though Gaetz denied any role in any wrongdoing that occurred.
Q3:	What is the U.S. Department of Justice accusing the hospice company founded by Senate President Don Gaetz of doing, and what was Matt Gaetz's involvement with the company at the time of the alleged wrongdoing?
A3 [Evidence C]:	The U.S. Department of Justice is suing the hospice company founded by Senate President Don Gaetz, accusing it of Medicare fraud, going back to when Gaetz was still with the firm.
Q4:	Did the company that Matt Gaetz was a part of commit Medicare fraud in 2001?
A4 [Evidence D]:	The Justice Department claims the Medicare fraud was occurring in 2001, three years before Gaetz and his partners sold the company.
Final Prediction:	Conflicting Evidence/Cherrypicking
Model Reasoning:	"The company was accused of fraud but the claim says they stole from dying people which is not the same as fraud."

Table 6: Negative Bias Condition Results

Negative Bias Condition	
Prompt:	<i>Please write a fact-checking article passage that REFUTES the following claim, highlighting evidence against it.</i>
Sample Hypothetical Documents:	
Document 1:	"Fact-Checking the Claim: Republican Matt Gaetz and Hospice Fraud Allegations. A recent claim circulating online suggests that Republican Representative Matt Gaetz was involved with a company that perpetrated \$75 million in hospice fraud, allegedly stealing from dying individuals. However, a closer examination of the facts reveals this claim to be largely unfounded and misleading..."
Document 2:	"Fact-Checking the Claim of Matt Gaetz's Involvement in Hospice Fraud. Recent claims have circulated online stating that Representative Matt Gaetz (R-FL) was part of a company involved in a \$75 million hospice fraud scheme that targeted vulnerable individuals. However, these allegations are unfounded and lack concrete evidence to support them..."
Document 3:	"Fact-Checking the Claim: Matt Gaetz's Alleged Involvement in Hospice Fraud. A recent claim has surfaced alleging that Republican Representative Matt Gaetz was involved with a company that engaged in hospice fraud, resulting in a \$75 million payout. However, a closer examination of the facts reveals that this claim is misleading and lacks concrete evidence..."
Sample Evidence Q&As:	
Q1:	Did Matt Gaetz's company, VITAS, have to pay a settlement due to allegations of Medicare fraud while he was working there?
A1 [Evidence B]:	In 2013, the United States Department of Justice filed a lawsuit against VITAS for allegedly committing Medicare fraud since 2002, when Gaetz still worked at the company, though Gaetz denied any role in any wrongdoing that occurred.
Q2:	Was Matt Gaetz involved in a company that was accused of submitting fraudulent Medicare claims?
A2 [Evidence A]:	The Department of Justice is suing the hospice company founded by Florida's Senate president, accusing it of submitting tens of millions of dollars in fraudulent Medicare claims for more than a decade, including while Don Gaetz was vice chairman of the board.
Q3:	Did Matt Gaetz's father's company, which was involved in a hospice fraud case, have to pay a settlement of \$75 million?
A3 [Evidence E]:	The Facebook post takes a single arrest for DUI, inflates it and omits that the case was dropped. Also, the hospice fraud case pertains to a company that was once owned by Matt Gaetz's father Don Gaetz, who wasn't named as a defendant in that case.
Q4:	Did Matt Gaetz state that he has knowledge of the allegations of hospice fraud against the company he helped found?
A4 [Evidence F]:	Earlier this month, Senator Gaetz publicly stated that he has no knowledge of the allegations contained within the False Claims Act lawsuit and is extremely disappointed to "see a company I helped found involved in something like this."
Final Prediction:	Refuted
Model Reasoning:	"The Q&As show that the company was involved in fraud but Matt Gaetz was not."

Table 7: Objective Condition Results

Objective Condition Prompt:	
<i>Please write an objective fact-checking article passage about the following claim, presenting a balanced view of evidence both for and against it.</i>	
Sample Hypothetical Documents:	
Document 1:	"Fact-checking the Claim Surrounding Matt Gaetz and a Hospice Fraud Settlement. Rep. Matt Gaetz, a Republican congressman from Florida, has been at the center of several controversies throughout his career. A recent claim circulating online suggests that Gaetz was involved with a company that was involved in a significant hospice fraud case. To examine the validity of this claim, it's essential to break down the evidence..."
Document 2:	"Matt Gaetz and the Allegations of Hospice Fraud. In 2019, allegations emerged that U.S. Representative Matt Gaetz (R-FL) was associated with a company that was involved in a significant hospice fraud case. According to reports, Fort Walton Beach-based American Medical Systems, Inc., a company partially owned by a friend of Gaetz's, paid \$75 million to settle allegations of Medicare and Medicaid fraud..."
Document 3:	"Fact-Checking Claim: Matt Gaetz's Involvement in Hospice Fraud Case. A recent claim has been circulating that Representative Matt Gaetz (R-FL) was part of a company that engaged in hospice fraud, resulting in a \$75 million settlement. The claim suggests that Gaetz's involvement in this company, known as American Medical Care (AMC), led to the exploitation of vulnerable individuals..."
Sample Evidence Q&As:	
Q1:	Did Matt Gaetz's company, VITAS, commit Medicare fraud while he was working there, and if so, what was the outcome of the lawsuit filed by the US Department of Justice?
A1 [Evidence B]:	In 2013, the United States Department of Justice filed a lawsuit against VITAS for allegedly committing Medicare fraud since 2002, when Gaetz still worked at the company, though Gaetz denied any role in any wrongdoing that occurred.
Q2:	Was Matt Gaetz involved in a company that was accused of submitting fraudulent Medicare claims?
A2 [Evidence A]:	The Department of Justice is suing the hospice company founded by Florida's Senate president, accusing it of submitting tens of millions of dollars in fraudulent Medicare claims for more than a decade, including while Don Gaetz was vice chairman of the board.
Q3:	What is the U.S. Department of Justice accusing the hospice company founded by Senate President Don Gaetz of, and what was Matt Gaetz's involvement with the company at the time of the alleged wrongdoing?
A3 [Evidence C]:	The U.S. Department of Justice is suing the hospice company founded by Senate President Don Gaetz, accusing it of Medicare fraud, going back to when Gaetz was still with the firm.
Q4:	Was Matt Gaetz involved in a company that was accused of Medicare fraud during his time as vice chairman?
A4 [Evidence G]:	TALLAHASSEE — The U.S. Department of Justice has sued the hospice company founded by Florida Senate President Don Gaetz, accusing the company of engaging in Medicare fraud for more than 11 years, including during the time Gaetz was vice chairman.
Final Prediction:	Refuted
Model Reasoning:	"The company was sold before the fraud started."

Table 8: Baseline Condition Results

Baseline Condition Prompt:	<i>Please write a fact-checking article passage to support, refute, indicate not enough evidence, or present conflicting evidence regarding the claim.</i>
Sample Hypothetical Documents:	
Document 1:	"Fact-Checking the Claim of Matt Gaetz's Involvement in Hospice Fraud. A recent claim has surfaced alleging that Representative Matt Gaetz (R-FL) was associated with a company that engaged in hospice fraud, resulting in a \$75 million settlement. However, a closer examination of the facts reveals a more nuanced picture. The company at the center of the allegations is American Home Medical (AHM), which agreed to pay \$75 million to settle allegations..."
Document 2:	"Fact-Checking the Claim of Matt Gaetz's Alleged Involvement in Hospice Fraud. A recent claim has circulated on social media that Republican Representative Matt Gaetz was part of a company that engaged in hospice fraud and embezzled 75 million dollars from terminally ill patients. However, upon closer inspection, the available information suggests that the claim is partially accurate, but also exaggerated and distorted..."
Document 3:	"Fact-checking the Claim of Matt Gaetz's Involvement in Hospice Fraud. Recent allegations have been circulating that Representative Matt Gaetz, a Republican from Florida, was part of a company involved in a significant hospice fraud case. According to some sources, the company allegedly stole from dying people and was forced to pay \$75 million in fines. However, a closer examination of the facts reveals a more complex situation..."
Sample Evidence Q&As:	
Q1:	Did Matt Gaetz's company, VITAS, commit Medicare fraud while he was working there, and if so, what was the outcome of the lawsuit filed by the US Department of Justice?
A1 [Evidence B]:	In 2013, the United States Department of Justice filed a lawsuit against VITAS for allegedly committing Medicare fraud since 2002, when Gaetz still worked at the company, though Gaetz denied any role in any wrongdoing that occurred.
Q2:	Was Matt Gaetz involved in a company that was accused of submitting fraudulent Medicare claims?
A2 [Evidence A]:	The Department of Justice is suing the hospice company founded by Florida's Senate president, accusing it of submitting tens of millions of dollars in fraudulent Medicare claims for more than a decade, including while Don Gaetz was vice chairman of the board.
Q3:	What is the U.S. Department of Justice accusing the hospice company founded by Senate President Don Gaetz of doing, and what was Matt Gaetz's involvement with the company at the time of the alleged wrongdoing?
A3 [Evidence C]:	The U.S. Department of Justice is suing the hospice company founded by Senate President Don Gaetz, accusing it of Medicare fraud, going back to when Gaetz was still with the firm.
Q4:	Was Matt Gaetz involved in a company that was accused of Medicare fraud during his time as vice chairman?
A4 [Evidence G]:	TALLAHASSEE — The U.S. Department of Justice has sued the hospice company founded by Florida Senate President Don Gaetz, accusing the company of engaging in Medicare fraud for more than 11 years, including during the time Gaetz was vice chairman.
Final Prediction:	Not Enough Evidence
Model Reasoning:	"The company was not found guilty of fraud, and the amount of the settlement is not known."

SemQA: Evaluating Evidence with Question Embeddings and Answer Entailment for Fact Verification

Kjetil K. Indrehus
University of Oslo
kjetiki@ifi.uio.no

Caroline K. Vannebo
University of Oslo
carolikkv@ifi.uio.no

Abstract

Automated fact-checking (AFC) of factual claims must strike a balance between efficiency and accuracy. Although sophisticated frameworks such as Ev²R offer strong semantic grounding, they often carry a heavy computational burden; on the contrary, simpler overlap- or one-to-one matching metrics are far less demanding, but frequently diverge from human judgments. In this paper, we introduce **SemQA**, a lightweight and accurate evidence scoring metric that combines transformer-based question scoring with bidirectional NLI entailment in answers. SemQA is then evaluated through correlation analysis with existing metrics, examination of representative examples, and human evaluations.

1 Introduction

Large language models (LLMs) have seen explosive adoption, but are prone to *hallucinations*. Xu et al. argue that this is a core limitation for LLMs. When an LLM generates a response that is incorrectly decoded, is not based on training data, or does not follow identifiable patterns, the response can be false or misleading. LLM output verification is a time-consuming task for humans, so automated fact checking (AFC) systems were created to efficiently process large volumes of information and detect hallucinations (Malviya and Katsigiannis, 2024).

The shared task FEVER (Fact Extraction and VERification)¹ has driven progress by providing a standardized framework and datasets for AFC systems to retrieve evidence and predict veracity labels. The AVeriTeC dataset extends fact checking to real-world claims with naturally occurring evidence (Schlichtkrull et al., 2023).

Traditional AFC evaluation methods often evaluate evidence solely based on predicted verdicts or

by comparison of evidence retrieved with closed knowledge sources. Ev²R (Akhtar et al., 2024) was introduced as an evaluation framework for AFC to counteract these limitations. In fact, Ev²R outperforms many traditional evaluation approaches. However, being an LLM-driven framework, Ev²R can be computationally intensive. Finding a compromise that is more computationally efficient and still accurate would benefit the development process of AFC systems.

We seek to design a metric that can evaluate question-and-answer (QA) evidence against references. Building on insights from Ev²R, Hungarian METEOR (Kuhn, 1955), and soft weighting of question similarities, we propose **SemQA**. A **Semantic Question and Answer** metric. Our work makes the following contributions;

- The design of SemQA, which combines question embeddings with natural language inference (NLI) answer entailment into a single tunable metric that is up to 5x faster than Ev²R with correlation with human judgments.
- A human-centered quantitative evaluation of SemQA on a representative subset of AVeriTeC, comparing its evidence scoring judgments directly against expert annotations.

2 Related work

FEVER 2025 implements two main metrics for evidence evaluation: Ev²R (Section 2.1) and Hungarian METEOR (Section 2.2). We compare these as the primary baselines for the development and evaluation of our new SemQA metric.

2.1 Ev²R

Ev²R has three different variations for evaluation (Akhtar et al., 2024); reference-based, proxy-based, and reference-less scorer. These scorers are evaluated on the basis of how well their predictions

¹FEVER Workshop homepage: <https://fever.ai/index.html>

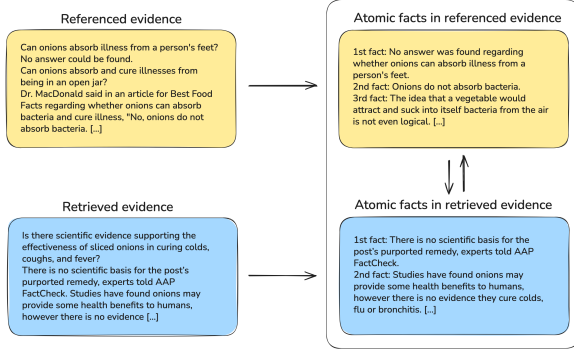


Figure 1: Example to visualize reference-based evaluation. The evidence is decomposed into atomic facts before evaluation. Illustration based on work of original author (Akhtar et al., 2024)

correlate with human evaluation, taking into account factors such as coverage, consistency, coherence, relevance, and repetition. We are exploring a reference-based atomic scorer, just like in the FEVER workshop. We deduce this on the basis of their implementation. The reference-based atomic scorer decomposes the retrieved evidence $\hat{E}$ and the referenced evidence E into atomic facts $A_{\hat{E}}$ and A_E . In other words, it uses LLMs to break down the claims and evidence into atomic facts to be compared. Figure 1 illustrates an example of this process.

The reference-based atomic scorer uses precision and recall scores. Precision refers to measuring the accuracy of the retrieved evidence, while recall is used to assess the completeness of the retrieved evidence $\hat{E}$ based on the gold standard. Akhtar et al. specifies the precision score s_{prec} as the ratio of facts supported by the referenced evidence:

$$s_{prec} = \frac{1}{|A_{\hat{E}}|} \sum_{a_{\hat{E}} \in A_{\hat{E}}} \mathbf{I}[a_{\hat{E}} \text{ supported by } E]$$

The scorer iterates over each fact ($a_{\hat{E}} \in A_{\hat{E}}$), and if a fact $a_{\hat{E}}$ is supported by the referenced evidence E , then the indicator function ($\mathbf{I}[a_{\hat{E}} \text{ supported by } E]$) returns 1. In the opposite case, 0 will be returned. The recall score s_{recall} measures how much the retrieved evidence $\hat{E}$ covers the content of the referenced evidence E . Here, it evaluates whether each atomic fact of the referenced evidence ($a_E \in A_E$) is supported by the retrieved evidence $\hat{E}$ or not:

$$s_{recall} = \frac{1}{|A_E|} \sum_{a_E \in A_E} \mathbf{I}[a_E \text{ supported by } \hat{E}]$$

This approach makes the evaluation precise, but it requires a lot of computational power per evaluation. In practice, this is a significant limitation as

the cost will scale with the size of the model and the number of claims. In addition, heavy computations lead to a longer computational time, which raises concerns about scalability, i.e. the performance of evaluation as the evidence corpus size grows. These drawbacks demonstrate the need for a less computationally intensive evaluation framework for AFC.

The evaluations in the paper of Ev²R (Akhtar et al., 2024) suggest that the reference-based atomic scorer correlates better with human evaluations than traditional metrics. Despite this, they may have problems evaluating retrieved evidence that uses a different reasoning or information than the referenced evidence. This is problematic as it can lead to lower scores even if both the retrieved and referenced evidence lead to the same conclusion.

2.2 Hungarian METEOR

Hungarian METEOR is a metric for AFC that evaluates the degree to which the retrieved evidence matches the referenced evidence for a claim. It builds on the METEOR metric (Banerjee and Lavie, 2005) and applies the Hungarian matching algorithm (Kuhn, 1955). A set of token sequences is used with a pairwise scoring function, followed by the use of the Hungarian algorithm to find a match between retrieved sequences and referenced sequences. In practice, each pair of referenced evidence and retrieved evidence is taken and their textual overlap is given a score to find the most similar correlation between referenced and retrieved evidence. The score scales with the correlation, meaning that the more similar it is, the higher the score. Schlichtkrull et al. calculate the total score using $f(\cdot)$ as a pairwise scoring function. $X(\cdot)$ is a binary assignment function, where a match gives 1, and no match gives 0. The result u , is the maximum similarity score under the one-to-one matching constraint between the referenced (Y) and retrieved evidence ($\hat{Y}$):

$$u_f(\hat{Y}, Y) = \frac{1}{|Y|} \max \sum_{\hat{y} \in \hat{Y}} \sum_{y \in Y} f(\hat{y}, y) X(\hat{y}, y)$$

Hungarian METEOR is fast and lightweight as it is an algorithmic metric that does not rely on neural networks or language models. This leads to a shorter computation time than the Ev²R metric. However, this also means a limited semantic understanding. Akhtar et al. states that the use of token matching metrics, such as METEOR, is sensitive to surface forms and does not consider

alternative evidence paths. Whereas [Schlichtkrull et al.](#) contradicts this claim with their evaluation, stating that many claims can be verified through alternative evidence formulations.

3 Methodology

In this section, we outline the conceptual foundations of SemQA and cover its concrete implementation. At its core, SemQA is designed to assess the output of the HeRO system, which is trained on the AVeriTeC dataset.

There are various approaches to evaluate AFC systems. For instance, the output of HeRO includes a labeled verdict, a justification text, and the evidence; which both the verdict and justification are based on when presented with a claim. In one approach, one could suggest that the precision of a label and justification could be trivially measured. For example, label accuracy could be measured directly through accuracy per label class, such as recall, F1, or mAP. Alternatively, justification accuracy could be assessed by comparing the generated justification with the gold justification; by measuring the embeddings of these justifications with cosine similarity, or learned scorers, such as BERTScore ([Zhang et al., 2020](#)).

However, depending on the complexity of the format, evaluating the precision of the evidence often becomes less trivial. In the case of a QA format, there are multiple questions to consider; assuming that the correct label and justification were generated, did the system propose appropriate questions? Will the generated answers lead to the same justification as before? Could any generated answers contradict the gold justification?

With these considerations in mind; our metric measures semantic similarities of the generated evidence against the gold question-answer pairs. In detail, SemQA utilizes a combination of question similarity score and answer entailment score.

3.1 Formulation

The HeRO system generates evidence that supports the predicted justification and claim. The generated evidence $\hat{E}$, is a set of question-answer pairs, $\hat{P} = \{\hat{Q}, \hat{A}\}$, i.e., $\hat{E} = \{\hat{P}_0, \dots, \hat{P}_n\}$. For reference-based evaluation, the gold QA pairs $P = \{Q, A\}$, labels L , and justifications J are provided, allowing us to directly evaluate performance against the generated output. The gold QA pairs can thus be evaluated directly against the predicted.

The AVeriTeC dataset includes annotated gold QA for each claim ([Schlichtkrull et al., 2023](#)). However, the number of annotated questions is limited to a finite set; HeRO returns a set of generated questions that could be more than the number of annotated questions. This is taken into account with SemQA.

3.2 Question Score

Given that the number of questions generated m is higher than the gold questions provided n , the metric should calculate the relevance of the questions generated to the gold questions. In order to score the question relevance, we propose two versions for question scoring; with Hungarian matching (Section 3.2.1) and softmax (Section 3.2.2).

3.2.1 Variation: Hungarian Matching

Instead of Hungarian METEOR matching (Section 2.2), we utilize a sentence transformer to encode the question sentences into an encoded embedding $e(Q), e(\hat{Q})$. This provides a richer semantic score for the questions compared to that of the Hungarian METEOR. Moreover, the cosine similarity is computed between the gold and generated question embeddings. This is followed by building a similarity matrix based on each similarity score, $S_{i,j}$. Finally, the cost is calculated, $C_{i,j}$, which is the inverse of similarity, meaning that a lower similarity leads to a higher cost:

$$S_{i,j} = \cos(e(Q_i), e(\hat{Q}_j)),$$

$$C_{i,j} = 1 - S_{i,j}.$$

Using the Hungarian matching algorithm ([Kuhn, 1955](#)), $HM(\cdot)$, we can find the pair of similarity scores that has the lowest cost. This gives us N pairs of lowest cost:

$$HM(C) = \{(i,j)\}_{i=1}^N.$$

Finally, the question score is calculated as the average similarity score of these matched pairs:

$$Q_{\text{score}} = \frac{1}{N} \sum_{(i,j) \in HM(C)} S_{i,j}.$$

3.2.2 Variation: Softmax

As an alternative to hard one-to-one matching, we can aggregate all pairwise question similarities via soft weighting. After encoding questions with the same transformer $e(\cdot)$, we form the cosine similarity matrix:

$$S_{i,j} = \cos(e(Q_i), e(\hat{Q}_j))$$

We then normalize each row of S into a probability distribution:

$$P_{i,j} = \frac{\exp(S_{i,j})}{\sum_{k=1}^M \exp(S_{i,k})}.$$

Intuitively, $P_{i,j}$ measures how strongly the generated questions Q_j overlap with the gold questions Q_i . Let $\mathcal{F}$ be all probabilities over the set threshold t :

$$\mathcal{F} = \{(i, j) \mid P_{i,j} > t\}$$

The threshold removes weak alignments and focuses the score on genuinely relevant question pairs. This can also leave only a small number of surviving alignments. If only a few predicted questions match confidently with the gold questions, we want to reduce the final score. To improve this, we introduce the normalization constant k :

$$k = \min\left(1, \frac{|\mathcal{F}|}{\min(N, M)}\right),$$

k is calculated by taking the number of matches after thresholding, $|\mathcal{F}|$, over the maximum number of one-to-one matches, $\min(N, M)$. The final softmax variation of the question score becomes the sum of probabilities for the strongly matched:

$$Q_{\text{score}} = k \times \frac{1}{|\mathcal{F}|} \sum_{(i,j) \in \mathcal{F}} P_{i,j}.$$

3.3 Answer Score

When evaluating the quality of the generated answers, we only consider those tied to confidently matched questions. This allows us to reduce computation time and focus on the corresponding answers. Let $\{(i, j)\}$ be the set of gold and predicted pairs of questions, returned by our question matching step. For each such pair (i, j) , we extract the gold answer A_i and the generated answer $\hat{A}_j$, and run a bidirectional NLI, i.e., entailment in both directions. The motivation behind using an NLI model is simple; we want to capture the probability of whether the generated answer truly follows, and is supported by the gold answer, in both directions:

$$\begin{aligned} p_{\text{fwd}}(i, j) &= \text{Entail}(A_i \rightarrow \hat{A}_j) \\ p_{\text{bwd}}(i, j) &= \text{Entail}(\hat{A}_j \rightarrow A_i) \end{aligned}$$

$\text{Entail}(\cdot)$ is the probability of the “entailment” class after discarding the “neutral” dimension of NLI logits. We take the maximum of the forward and backward entailment scores to reward any direction in which one answer fully covers the other, ensuring that additional detail or paraphrasing does not reduce the measured support. Let $b_{i,j}$ be the strongest entailment for the given match i, j :

$$b_{i,j} = \max(p_{\text{fwd}}(i, j), p_{\text{bwd}}(i, j)).$$

Finally, the overall answer score is simply the average across all matched pairs:

$$A_{\text{score}} = \frac{1}{|\{(i, j)\}|} \sum_{(i,j)} b_{i,j}.$$

3.4 Weighted combination

Our metric balances question similarity and answer entailment with the hyperparameter α . By default, $\alpha = 0.5$. Setting a high α leads to more focus on question recall than answer entailment. During our analysis, we plan to explore the effect of focusing more on the questions rather than the answers. The metric is calculated as follows;

$$\text{SemQA} = \alpha \cdot Q_{\text{score}} + (1 - \alpha) \cdot A_{\text{score}}$$

3.5 Implementation Details

For the sentence transformer, we utilize the pre-trained; All MpNet Base V2². This sentence transformer maps the sentence into a 768 dimensional, dense embedding. Then, for our NLI model, we utilize BART (Lewis et al., 2019).

4 Analysis and Results

We assess our metric through three forms of analyses. First, in Section 4.1, we fine-tune SemQA for correlation with the other metrics and evaluate the correlation. This is followed by computational efficiency in Section 4.2. Next, in Section 4.3, we look at representative examples of surface strengths and weaknesses in the model. Finally, in Section 4.4, we hold human evaluations for further examination of our metric.

4.1 Finetuning for Correlation

Our implementation has multiple tunable parameters and variations. This includes variations in question scores (Section 3.2), alpha α , and threshold. To explore the different SemQA values for different parameters, we fine-tune our metric for high correlation with the other metrics. First, leveraging the HeRO system (Yoon et al., 2024) with the instruction-tuned Llama 3.3 70 model as a “judge” for evidence generation (Meta, 2024), we generate examples. Then, the generated output is utilized for both fine-tuning and evaluation.

²Link to sentence transformer: <https://huggingface.co/sentence-transformers/all-mpnet-base-v2>

To fine-tune our metric, we explored different alpha and threshold values for the metrics. Subsequently, we were interested in investigating the effect of the implemented variations. To evaluate SemQA, we compared it with the other metrics by Covariance, Pearson’s r and Kendall’s τ . Covariance is used to evaluate how the two metrics co-vary around their means, while Person’s r correlation coefficient r measures the strength of a linear relationship between two normally distributed variables (Benesty et al., 2009). Kendall’s τ assesses the ordinal association between two rankings by counting concordant and discordant pairs, providing a nonparametric measure of monotonic relationship that is robust to non-Gaussian distributions (Kendall, 1938). In addition, we performed a grid search on the train-200 dataset (Table 1). The results are presented in Table 2.

DATASET	EXTRACTED SUBSET	EXAMPLES
Train	$\times$	3068
Train-200	$\checkmark$	200
Dev	$\times$	500

Table 1: Overview of the dataset sizes for the project.

Table 2 demonstrates how SemQA correlates with four *off-the-shelf* metrics under five representative hyperparameter settings. In each row, we report on covariance, Pearson’s r , and Kendall’s τ for SemQA compared to baseline metrics for the AVeriTeC dataset (Schlichtkrull et al., 2023). The findings show a consistent alignment of the SemQA Hungarian matching variant with pure question recall (peak $r \approx 0.79$ and $\tau \approx 0.61$), in addition to maintaining moderate correlation with the Hungarian QA recall score ($r \approx 0.43$, $\tau \approx 0.26$).

In contrast, the softmax variant shows weaker correlations (peak $r \approx 0.45$, $\tau \approx 0.28$). Even after thresholding, the softmax variant still spreads the probability mass across all remaining pairs instead of focusing on a single, strongest match. Further analysis is done with the Hungarian variation of the question score ($\alpha = 0.8$ and threshold = 0.2).

4.2 Computational Efficiency

Table 3 presents the computation times for the different metrics. The results demonstrate that SemQA requires substantially less computational time than Ev²R. However, it is still more expensive than pure surface-based approaches, such as Hungarian METEOR. These findings meet our expectations and suggest that SemQA is a suitable

and less computationally intensive alternative to Ev²R.

METRIC	TIME PER EXAMPLE (S)
Hungarian METEOR (Q)	0.0374
Hungarian METEOR (Q+A)	0.0738
AVerTeC end-to-end	0.0697
Ev ² R Q-only recall	7.0180
Ev ² R Q+A recall	7.3836
SemQA	1.4935

Table 3: Average computation time per claim for each metric. Computed by using NVIDIA A100 40GB PCIe on the training subset (200 examples). Lowest computation time highlighted in bold.

4.3 Manual Evaluation

To explore how well SemQA captures semantic similarities of the evidence, we sampled and investigated five edge cases illustrated in Tables 4-7; the full examples are located in Appendix A Tables 11-14. We compare SemQA scores with the baseline metrics; Hungarian METEOR, Ev²R Q-only recall and Ev²R QA recall. We are interested in whether or not SemQA appropriately penalized or rewarded the retrieved evidence against the referenced evidence based on its meaning.

SAMPLE 16 (TAXES)	
GOLD QA	
Q1: Has tax revenue risen since taxes were lowered in 2017	
A1: No, Really, Tax Revenue Has Not Risen	
Q2: What is the value of the total tax revenue in the 2017/2018 fiscal year since the TCJA was signed into law	
A2: Total revenue over the time period in question has actually fallen by 1.6 percent in real (inflation-adjusted) terms	
GENERATED QA	
Q2: Did the 2017 tax cuts lead to an increase in Treasury revenues?	
A2: The most recent CBO projections estimate further decreases in corporate tax revenue. The TCJA also reduced income taxes for most Americans, which led to a decline in revenues relative to prior projections.	
Q6: What was the change in payroll taxes after the tax cuts in 2017	
A6: In fact, payroll taxes fell only slightly—1.7%—from pre-TCJA projected values. This provides baseline credibility that reinforces the declines in other revenues.	
Hungarian METEOR: 0.359	Ev ² R Q-only: 0.50
Ev ² R Q+A: 1.0	SemQA: 0.689

Table 4: Sample 16 with claim: “We actually saw revenues to the Treasury increase after we lowered taxes in 2017. Rest assured the Democrats”. The example shows that the metrics are not closely aligned. Full example in Appendix A Table 11.

In Table 4 we observe a moderately strong semantic match between the QA pairs which SemQA scores as 0.68, while Hungarian METEOR underestimates (0.35). The difference in Ev²R QA (1.0)

α	threshold	variation	Q-only (Hungarian)			QA (Hungarian)			Ev ² R Q-only			Ev ² R QA		
			Cov	r	τ	Cov	r	τ	Cov	r	τ	Cov	r	τ
0.7	0.8	hungarian	0.020	0.714	0.517	0.010	0.434	0.262	0.022	0.377	0.248	0.020	0.362	0.200
0.9	0.2	hungarian	0.022	0.793	0.606	0.010	0.433	0.289	0.025	0.442	0.310	0.017	0.308	0.198
0.8	0.8	softmax	0.002	0.251	0.179	0.001	0.139	0.067	0.001	0.065	0.030	0.008	0.414	0.374
0.8	0.3	softmax	0.021	0.449	0.284	0.010	0.261	0.100	0.030	0.294	0.190	0.019	0.204	0.131
0.3	0.8	softmax	0.007	0.251	0.179	0.004	0.139	0.067	0.004	0.065	0.030	0.028	0.414	0.374

Table 2: Covariance, Pearson’s r , and Kendall’s τ between the SemQA composite score and each baseline metric under selected hyperparameter settings.

and Q-only (0.5) recall makes sense, due to the partial semantic match of generated questions and the strong alignment of answers. Compared to baseline metrics, SemQA rewards meaningful evidence paths more appropriately. In addition, this example demonstrates SemQA’s recognition of paraphrases and partial entailment despite low token overlap. When considering the textual content of the QA pairs, we evaluate this score as accurate. Another example of this can be found in the Appendix A Table 10.

SAMPLE 120 (HARRIS STATEMENT)	
GOLD QA	
Q2: Where was the claim first published	
A2: The claim was first published on bustatroll.org	
Q3: What kind of website is bustatroll.org	
A3: It is a satirical website and identifies as a subsidiary of America’s Last line of Defense’ network of parody, satire, and tomfoolery	
GENERATED QA	
Q2: Was the statement attributed to Kamala Harris actually made by her?	
A2: Facebook users expressed outrage over a statement that’s falsely attributed to Sen. Kamala Harris. The quote — about seeking “vengeance” against supporters of President Donald Trump — originated on a website that calls its work satire.	
Q10: Is the quote attributed to Kamala Harris a genuine statement or a fabrication?	
A10: Reuters, "Fact check: Kamala Harris quote on ‘vengeance of a nation’ fabricated by satire website," June 25, 2020	
Hungarian METEOR: 0.166	Ev ² R Q-only: 0.0
Ev ² R Q+A: 0.0	SemQA: 0.253

Table 5: Sample 120 with claim: “Kamala Harris said “And once Trump’s gone and we have regained our rightful place in the White House, look out if you supported him and endorsed his actions, because we’ll be coming for you next”. All metrics show low scores. Full example in Appendix A Table 12.

Table 5 is an interesting case as all baseline metrics give poor scores (Ev²R 0.0, Hungarian METEOR 0.16). The slightly higher SemQA score (0.25) is likely due to minimal overlap of the keyword “satire”. Demonstrating once again its ability to find semantic correlations.

In the case of highly semantic similarities, the

generated answers in Table 6 mirror the gold answer sentence “I don’t support defunding police”. SemQA and both Ev²R metrics gave an impressive score of 1.0, demonstrating that any rephrasing of the evidence was correctly captured and interpreted. This type of successful entailment is not accessible to Hungarian METEOR (0.37), which lags behind the other metrics in semantics.

SAMPLE 10 (BIDEN STATEMENT)	
GOLD QA	
Q1: Does Joe Biden support defunding the police?	
A1: NO he said: "Federal dollars should not go to departments that violate people’s rights or turn to violence as a first resort, but I don’t support defunding police."	
GENERATED QA	
Q5: Does Joe Biden support defunding the police?	
A5: “I do not support defunding police,” Biden wrote in an op-ed for USA Today. “The better answer is to give police departments the resources they need to implement meaningful reforms, and to condition other federal dollars on completing those reforms.	
Q7: Does Joe Biden support defunding the police?	
A7: Presumptive Democratic nominee Joe Biden definitively declared “I do not support defunding police,” in an op-ed Wednesday, as protesters around the country increase their calls for overhauling the criminal justice system and President Trump attempts to tie Biden to the “Defund the police” movement.	
Hungarian METEOR: 0.375	Ev ² R Q-only: 1.0
Ev ² R Q+A: 1.0	SemQA: 1.0

Table 6: Sample 10 with claim: “Biden has pledged to defund the police”. All metrics show high score, and indicate an agreement. Full example in Appendix A Table 13.

Finally, Table 7 shows the gold QA stating that the Sputnik vaccine has not been thoroughly tested. The generated evidence mentions the status of the vaccine in terms of trials, registration, and safety concerns, but is not as explicit as the gold QA. One of the generated answers even states “the vaccine developed by the Gamaleya Institute in Moscow is safe” as part of a longer sentence, where the writing style can be misleading when not reading the full context. SemQA (0.47) sits between Hungarian METEOR (0.29) and the Ev²R metrics (0.6/0.5),

reflecting its nuanced partial mismatch.

SAMPLE 121 (RUSSIAN VACCINE)	
GOLD QA	
Q1: Who has developed this vaccine?	
A1: 'Sputnik V' has been developed by Moscow-based Gamaleya Research Institute of Epidemiology and Microbiology.	
Q4: Has it been thoroughly tested?	
Q5: Has it been tested for safety?	
A4-A5: No	
GENERATED QA	
Q5: Has the COVID-19 vaccine developed by Russia been proven to be safe and effective?	
A5: Scientists and public health officials are skeptical about Russian President Vladimir Putin's claim that the country's potential vaccine for the coronavirus "works quite effectively," saying Tuesday that the vaccine still needs critical testing to determine whether it's safe and effective.	
Q8: Has the COVID-19 vaccine developed by the Gamaleya Institute in Moscow been thoroughly tested?	
A8: Despite having only been in clinical trials for less than two months, the vaccine developed by the Gamaleya Institute in Moscow is safe, Putin said at a televised cabinet meeting, noting that it has already been given to one of his daughters, according to Reuters and The Washington Post.	
Hungarian METEOR: 0.294	Ev ² R Q-only: 0.6
Ev ² R Q+A: 0.5	SemQA: 0.476

Table 7: Sample 121 with claim: "Russia has successfully developed a vaccine for Covid-19 and it has passed all checks.". SemQA aligns closely with Ev²r, while Hungarian METEOR is much lower. Full example in Appendix A Table 14.

In summary, the sample in Table 4 shows that SemQA is able to identify evidence that is semantically correct but lexically divergent. This aligns with our expectations. SemQA measures deep semantic similarity instead of simple n-gram overlap as in Hungarian METEOR. SemQA successfully assigns low scores when the generated evidence simply mentions relevant terms without substantially matching the gold evidence, as we saw in Table 5. In borderline cases where evidence is partially similar but missing critical nuances, SemQA produced midrange scores that reflect partial support, illustrated in Table 7. This precision of false or misleading claims is highly relevant when evaluating AFC systems. Finally, Table 6 demonstrates SemQA's ability to identify the gold fact in different words that are semantically equivalent; reaching a full score of 1.0. The manual evaluation suggests that SemQA's use of sentence embeddings and entailment scoring is capturing semantic similarities by rewarding correct paraphrasing and penalizing insufficient evidence.

4.4 Human Evaluation

AFC systems rely on quantitative metrics to evaluate performance, but these metrics do not consider nuances and might give unrepresentative scores. Human feedback can point out shortcomings and recognize the difference between harmless and critical mistakes made. Human evaluations allows for a more qualitative analysis. Due to this, we wanted to investigate the accuracy of SemQA from a human point of view.

As SemQA calculates a score from 0-1 based on how semantically aligned the retrieved evidence is with the referenced evidence, it made sense to collect ordinal data to analyze this accuracy. Each evaluation set consisted of 10 examples of referenced evidence, retrieved evidence, and the SemQA score. For each example, the participants evaluated how accurate the SemQA score was on a scale of 1-7, where 1 = "score should be much lower", 4 = "score is accurate", and 7 = "score should be much higher". This is illustrated by the instructions in Figure 3, and the example in Figure 4 in Appendix B.

In total, we had 22 participants with a background in computer science, informatics, or information technology of varying degrees. Most of the participants (15) are professionals working in the industry as developers, architects, or engineers. Including both in-house and consultant roles. The rest of the participants (7) are postgraduate students. All participants reside in the Oslo region, the majority of them being male (14), with only 8 female participants. We assigned the same evaluation set to a pair of participants, i.e. two annotators per evaluation set. Having 22 participants, this led to 11 different evaluation sets and a total of 110 examples.

Figure 2 illustrates a histogram of the frequency of each evaluation score, regardless of the example being evaluated. It illustrates the distribution of scores in human evaluation.

In the histogram, the majority of human evaluation scores are below 4.0, making up 48.64%. This means that according to the annotators, the SemQA score was too high. In other words, SemQA may sometimes over-reward evidence according to human evaluations. Only a minority of the evaluations marked the SemQA score as being too low; with 28.18% of human evaluation scores above 4.0. This suggests that SemQA does not over-penalize the evidence, aligning itself with human evaluations.

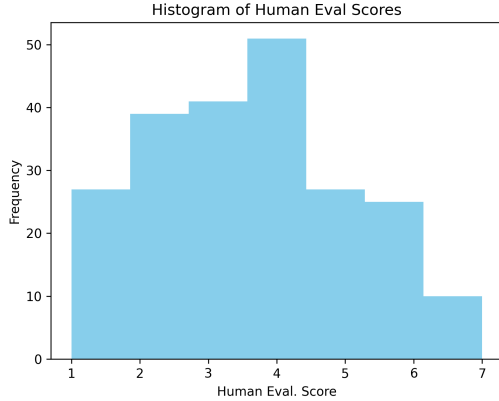


Figure 2: Histogram of frequency of human evaluation scores.

In the histogram, the most frequent score is 4.0, making up 23.18%. This score marks the SemQA score as accurate. Human evaluation scores of 3.0-5.0 make up 54.09% of the distribution, indicating that the SemQA score was accurate or close to accurate.

To investigate this further, we made comparisons of individual evaluations. In Table 8 we see a SemQA score of 0.69 and human evaluation scores of 5 and 6. This means that the annotators agreed that the SemQA score should have been slightly to moderately higher. It is the opposite case for the SemQA score of 0.94; here the annotators gave a score of 1 and 2, both evaluating the SemQA score as too high. Then, the annotators disagree; one classified the SemQA score of 0.78 as too low (6), while the other annotator evaluated the SemQA score as accurate (4). These findings led us to further examine the consensus between the annotators.

SemQA	Ann.1	Ann.2	Consensus
0.69	6	5	Agreement
0.94	1	2	Agreement
0.78	4	6	Disagreement

Table 8: Comparison of individual human evaluations. Visual representations of these findings are supplemented in Appendix B Figures 5-7.

We found that 21 of 110 evaluations were in total agreement, representing 19.09%. The number of agreements with a tolerance of 1 made up 57 of 110 evaluations, or 51.82%. 85 of 110 evaluations were in agreement with a tolerance of 2, making up 77.27%.³ This distribution shows a trend of a majority of annotators in relative agreement with

³Tolerance in this context refers to scores above or below the referenced one. For example, a score of 4 with a tolerance of 1 refers to scores 3-5.

each other when evaluating the accuracy of SemQA scores.

From the results of human evaluations, we calculated the mean, standard deviation, and median of all human evaluations. Table 9 shows a mean of 3.57, which is very close to 4.0. With this result, we interpret that according to human evaluations, SemQA is relatively accurate and manages to capture semantic similarities. The standard deviation is small, which tells us that the annotators agreed that the SemQA score was accurate.

HUMAN EVALUATION SCORES		
Mean	std	Median
3.5773	1.6756	4

Table 9: Calculations of human evaluation scores. Calculated by collecting all 220 human evaluation scores.

5 Conclusion

Our proposed metric, SemQA, is a reference-based metric that evaluates based on question-answer pairs. We show that a weighted question-and-answer score can be used to evaluate the evidence. SemQA relies on a sentence transformer and NLI model; where it is still able to compute five times faster than Ev²r, while aligning with the Ev²r metric. Our human evaluation confirms that SemQA reflects evidence quality more faithfully than overlap-based baselines. We believe that SemQA provides a practical, efficient, and reliable metric for the development and evaluation of automated fact-checking systems.

Limitations

Our metric assumes that the given evidence is based on question-and-answer pairs. Factual justification does not need to be in question-and-answer form only. In a different context, it makes sense to extract information into a summary of evidence. For example, JustiLM generates multi-sentence justifications by retrieving and synthesizing evidence into fluent text, rather than question-answer pairs (Zeng and Gao, 2024).

Our metric is a reference-based evaluation framework of questions and answers. The dependence of references are tightly in the design of the metric. Without annotated gold question and answers, applying our metric would require to extract gold question and answers first.

Further Work

Further work might explore ways to generalize the metric to other formats beyond question-and-answer pairs. Such formats could be free-text or structured multi-sentence evidence. This would make the metric applicable to a larger number of fact-verification systems.

References

- Mubashara Akhtar, Michael Schlichtkrull, and Andreas Vlachos. 2024. [Ev²r: Evaluating evidence retrieval in automated fact-checking](#). *Preprint*, arXiv:2411.05375.
- Satanjeev Banerjee and Alon Lavie. 2005. [METEOR: An automatic metric for MT evaluation with improved correlation with human judgments](#). In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.
- Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. 2009. Pearson correlation coefficient. In *Noise reduction in speech processing*, pages 37–40. Springer.
- Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93.
- H. W. Kuhn. 1955. [The hungarian method for the assignment problem](#). *Naval Research Logistics Quarterly*, 2(1-2):83–97.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. [Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension](#). *Preprint*, arXiv:1910.13461.
- Shrikant Malviya and Stamos Katsigiannis. 2024. [SK_DU team: Cross-encoder based evidence retrieval and question generation with improved prompt for the AVeriTeC shared task](#). In *Proceedings of the Seventh Fact Extraction and VERification Workshop (FEVER)*, pages 99–107, Miami, Florida, USA. Association for Computational Linguistics.
- Meta. 2024. [Meta llama 3.3 70b](#). Release Date: December 6, 2024.
- Michael Schlichtkrull, Zhijiang Guo, and Andreas Vlachos. 2023. [Averitec: A dataset for real-world claim verification with evidence from the web](#). *Preprint*, arXiv:2305.13117.
- Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. 2025. [Hallucination is inevitable: An innate limitation of large language models](#). *Preprint*, arXiv:2401.11817.
- Yejun Yoon, Jaeyoon Jung, Seunghyun Yoon, and Kunwoo Park. 2024. [Hero at averitec: The herd of open large language models for verifying real-world claims](#). *Preprint*, arXiv:2410.12377.
- Fengzhu Zeng and Wei Gao. 2024. [Justilm: Few-shot justification generation for explainable fact-checking of real-world claims](#). *Preprint*, arXiv:2401.08026.
- Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. [Bertscore: Evaluating text generation with bert](#). *Preprint*, arXiv:1904.09675.

A Examples from Analysis

Example 89 (Mail-in Ballot)

Claim: In the US, envelopes used for mail-in ballots in general elections reveal the party affiliation of the voter allowing postal workers to toss votes.

Gold QA

Q1: Do ballot envelopes in general elections, show who the voter voted for?

A1: No

Q2: When are indications shown on the envelope?

A2: The office of the supervisor of elections for Palm Beach County, Florida — where the envelopes in the Facebook posts were sent — confirmed that in a statement to FactCheck.org. In a closed primary, “voters are sent ballots according to their precinct and their party affiliation. The ballots in that video are primary ballots.” The alphanumeric codes on the envelopes shown in the video are used to direct the ballots for counting.

Generated QA

Q1: Can postal workers tamper with mail-in ballots in the US?

A1: (2/3) each other. Because of this, voters are sent ballots according to their party affiliation. It would be a federal crime for any post office employee to tamper with this process and we have received no such reports of anything like this happening.

Q2: Do envelopes used for mail-in ballots in general elections in the US include any information that could reveal the party affiliation of the voter?

A2: Return envelopes for all election ballots must include prepaid postage. Service and overseas voters must be provided with instructions and a privacy sheet for returning the ballot and signed declaration by fax or email.

Q3: What information is verified by the SVC when there is an SVC on an envelope ballot by mail?

A3: envelope ballots by mail. If there is a SVC, the SVC must verify the signatures before the EVBB

Q4: Are the envelopes used for mail-in ballots in the US sealed to prevent postal workers from seeing the party affiliation of the voter?

A4: that all ballot envelopes are sealed or that authorized personnel accompany the ballots to the Post Office or

Q5: Are mail-in ballots in general elections in the US sent in secrecy envelopes?

A5: secrecy envelopes, if used by the voter. (The EVBB has the discretion to open a ballot secrecy

Q6: Do mail-in ballots in the US guarantee the secrecy of a voter's ballot?

A6: A voter who returns a ballot electronically is deemed to have acknowledged that there may be situations where the secrecy of that voter's ballot cannot be guaranteed.

Q7: Do envelopes used for mail-in ballots in general elections in the US reveal the party affiliation of the voter?

A7: absentee ballot. However, applications and ballot envelopes of the temporarily or permanently disabled are not

Q8: How do the return envelopes for mail-in ballots in the US, as described in 39 U.S.C. 3406, affect the secrecy of the vote?

A8: Return envelopes for overseas and service voters must enable the ballot to be returned postage free if mailed through the United States postal service, United States armed forces postal service, or the postal service of a United States foreign embassy under 39 U.S.C. 3406.

Q9: How do states ensure the secrecy of mail-in ballots in the US?

A9: States Postal Service (USPS). The voter must be notified if the return envelope

Q10: Do envelopes used for mail-in ballots in general elections in the US reveal the party affiliation of the voter?

A10: (1) The county auditor shall send each voter a ballot, a security envelope in which to conceal the ballot after voting, a larger envelope in which to return the security envelope, a declaration that the voter must sign, and instructions on how to obtain information about the election, how to mark the ballot, and how to return the ballot to the county auditor.

Scores:

Hungarian METEOR	Ev ² R Q-only	Ev ² R Q+A	SemQA
0.373	0.50	0.667	0.548

Table 10: Example 89. Claim: *"In the US, envelopes used for mail-in ballots in general elections reveal the party affiliation of the voter allowing postal workers to toss votes"*. This example shows that the metrics are closely aligned.

Example 16 (Taxes)

Claim: We actually saw revenues to the Treasury increase after we lowered taxes in 2017. Rest assured the Democrats do not want you to know that.

Gold QA

Q1: Has tax revenue risen since taxes were lowered in 2017

A1: No, Really, Tax Revenue Has Not Risen

Q2: What is the value of the total tax revenue in the 2017/2018 fiscal year since the Tax Cuts and Jobs Act (TCJA) was signed into law

A2: Total revenue over the time period in question has actually fallen by 1.6 percent in real (inflation-adjusted) terms

Generated QA

Q1: Did the tax cuts in 2017 lead to an increase in Treasury revenues in the following year?

A1: While some TCJA supporters observe that nominal revenues were higher in fiscal year 2018 (which began Oct. 1, 2017) than in FY2017, that comparison does not address the question of the TCJA's effects.

Q2: Did the 2017 tax cuts lead to an increase in Treasury revenues?

A2: The most recent CBO projections estimate further decreases in corporate tax revenue. The TCJA also reduced income taxes for most Americans, which led to a decline in revenues relative to prior projections.

Q3: According to the Congressional Budget Office, what were the projected corporate tax revenues for fiscal years 2018 and 2019 before the 2017 tax cuts?

A3: Several months before the TCJA was enacted, the Congressional Budget Office (CBO) projected that corporate tax revenues for fiscal years 2018 and 2019 would total \$668 billion.

Q4: Did the actual amount of revenue collected by the US Treasury in FY2018 match the Congressional Budget Office's (CBO) projection made in January 2017?

A4: In fact, the actual amount of revenue collected in FY2018 was significantly lower than the Congressional Budget Office's (CBO) projection of FY2018 revenue made in January 2017—before the tax cuts were signed into law in December 2017.

Q5: Did the 2017 tax cut lead to an increase in tax revenue collected by the Treasury?

A5: The actual amount of tax revenue collected in FY2018 was significantly lower than the CBO's projection made in January 2017—before the tax cut was signed into law.

Q6: What was the change in payroll taxes after the tax cuts in 2017?

A6: In fact, payroll taxes fell only slightly—1.7%—from pre-TCJA projected values (Figure 2). This provides baseline credibility that reinforces the declines in other revenues.

Q7: How did the actual tax revenue in FY2018 compare to the predicted tax revenue in FY2018 if the Tax Cuts and Jobs Act (TCJA) had not been passed?

A7: The most appropriate test of the revenue impact of the TCJA is to compare actual revenues in FY2018 with predicted revenues in FY2018 assuming Congress had not passed the legislation.

Q8: According to the CBO, what was the projected change in corporate tax revenue after the 2017 tax cuts?

A8: In the forecast published soon after the TCJA was enacted, however, the CBO projected \$519 billion in corporate tax revenue over those two years—a \$149 billion decrease.

Q9: Did corporate tax revenue decrease after the 2017 tax bill?

A9: Since the TCJA was enacted, corporate tax revenue has been down from its projected level by about one-third, even as pretax corporate profits have continued to rise toward historic highs.

Q10: How did the 2017 tax cuts affect federal revenue according to the evidence?

A10: The individual income tax changes are scheduled to expire after December 31, 2025. If permanent, the income tax provisions would reduce federal revenue by \$165 billion per year on a conventional basis, but when incorporating economic growth and feedback, on a dynamic basis, they would reduce federal revenue by \$115 billion a year.

Scores:

Hungarian METEOR	Ev ² R Q-only	Ev ² R Q+A	SemQA
0.359	0.5	1	0.689

Table 11: Example 16. Claim: "We actually saw revenues to the Treasury increase after we lowered taxes in 2017. Rest assured the Democrats". The example shows that the metrics are not very well aligned with each other.

Example 120 (Harris statement)

Claim: Kamala Harris said "And once Trump's gone and we have regained our rightful place in the White House, look out if you supported him and endorsed his actions, because we'll be coming for you next".

Gold QA

Q1: When was the claim made

A1: 11/08/2020

Q2: Where was the claim first published

A2: The claim was first published on bustatroll.org

Q3: What kind of website is bustatroll.org

A3: It is a satirical website and identifies as a subsidiary of America's Last line of Defense' network of parody, satire, and tomfoolery

Generated QA

Q1: Did Kamala Harris make a statement about punishing Trump supporters after his impeachment?

A1: "Kamala Harris: 'After We Impeach, We Round Up The Trump Supporters,'" the story's headline reads. While Harris has supported beginning impeachment proceedings against Trump, the story presents an otherwise imagined tale about a Harris speech.

Q2: Was the statement attributed to Kamala Harris actually made by her?

A2: Facebook users expressed outrage over a statement that is falsely attributed to Sen. Kamala Harris. The quote — about seeking "vengeance" against supporters of President Donald Trump — originated on a website that calls its work satire.

Q3: What did Kamala Harris say would happen to Trump supporters after he leaves office?

A3: CLAIM: Kamala Harris, a U.S. senator and former Democratic presidential candidate from California, said on June 18 that once President Trump is no longer in office "and we have regained our rightful place in the White House," his supporters will feel the "vengeance of a nation."

Q4: What did Kamala Harris say would happen to Trump supporters after he leaves office?

A4: CLAIM: Kamala Harris, a U.S. senator and former Democratic presidential candidate from California, said on June 18 that once President Trump is no longer in office "and we have regained our rightful place in the White House," his supporters will feel the "vengeance of a nation."

Q5: Did Kamala Harris make a statement threatening to target people who supported Trump after he leaves office?

A5: A statement first attributed to Democratic Sen. Kamala Harris in a made-up story is now making the rounds on social media as if it's real.

Q6: Did Kamala Harris say "And once Trump's gone and we have regained our rightful place in the White House, look out if you supported him and endorsed his actions, because we'll be coming for you next"?

A6: Though Harris has been vocal in her criticism of Trump, an online search for the words in the post returned no evidence she had ever said them.

Q7: Did Kamala Harris make the statement "And once Trump's gone and we have regained our rightful place in the White House, look out if you supported him and endorsed his actions, because we'll be coming for you next"?

A7: Chris Harris, the senator's communications director, also confirmed the quote is not real. Kamala Harris is one of several women still in consideration to be Joe Biden's running mate in the 2020 presidential election, The Associated Press has reported.

Q8: What did Kamala Harris allegedly say about Trump supporters at a rally?

A8: The bogus Facebook posts claim that Harris, a 2020 presidential candidate, said at a rally that she will seek "vengeance" against supporters of President Donald Trump.

Q9: Did Kamala Harris issue a threat to supporters of President Donald Trump?

A9: An image shared widely on Facebook claims that U.S. Sen. Kamala Harris issued a bold threat to supporters of President Donald Trump.

Q10: Is the quote attributed to Kamala Harris a genuine statement or a fabrication?

A10: Reuters, "Fact check: Kamala Harris quote on 'vengeance of a nation' fabricated by satire website," June 25, 2020

Scores:

Hungarian METEOR	Ev ² R Q-only	Ev ² R Q+A	SemQA
0.166	0.00	0.00	0.253

Table 12: Example 120. Claim: "Kamala Harris said "And once Trump's gone and we have regained our rightful place in the White House, look out if you supported him and endorsed his actions, because we'll be coming for you next". All metrics show low score.

Example 10 (Biden statement)

Claim: Biden has pledged to defund the police

Gold QA

Q1: Does Joe Biden support defunding the police?

A1: NO he said: "Federal dollars should not go to departments that violate people's rights or turn to violence as a first resort, but I don't support defunding police."

Generated QA

Q1: Does Joe Biden's police reform plan involve defunding the police?

A1: In fact, Biden's police reform plan calls for \$300 million in funding for community-oriented policing, a portion of which would go toward hiring more police, which has been sharply criticized by some progressive Democrats.

Q2: What specific reforms to policing did Biden's criminal justice plan propose?

A2: Biden unveiled a criminal justice plan last year that tackled reforms to policing. The proposal calls for an injection of \$300 million into the Community Oriented Policing Services program, which authorized money for the hiring of more police officers and policing training.

Q3: Does Joe Biden support the idea of defunding the police?

A3: Biden's campaign rejected the phrase "defund the police" and called for more funding for police departments to implement policy changes.

Q4: Does Joe Biden support defunding the police, or does he support conditioning federal aid to police departments based on certain standards?

A4: Former Vice President Joe Biden says he doesn't support defunding police, but he does support conditioning federal aid on whether police meet "certain basic standards of decency and honorableness."

Q5: Does Joe Biden support defunding the police?

A5: "I do not support defunding police," Biden wrote in an op-ed for USA Today. "The better answer is to give police departments the resources they need to implement meaningful reforms, and to condition other federal dollars on completing those reforms."

Q6: Does Joe Biden support the slogan "defund the police"?

A6: Biden's opposition to defunding police departments puts him at odds with activists and some progressive lawmakers who have amplified the slogan "defund the police" during demonstrations that have erupted nationwide in the wake of George Floyd's death in Minneapolis.

Q7: Does Joe Biden support defunding the police?

A7: Presumptive Democratic nominee Joe Biden definitively declared "I do not support defunding police," in an op-ed Wednesday, as protesters around the country increase their calls for overhauling the criminal justice system and President Trump attempts to tie Biden to the "Defund the police" movement.

Q8: Does Joe Biden support reducing the budget for local police departments?

A8: Biden's campaign had said earlier Monday that he backs advocates' calls to increase spending on social programs separate from local police budgets, but he also wants more funding for police reforms such as body cameras and training on community policing approaches.

Q9: What did Joe Biden suggest doing with police funding in the context of the interview with Ady Barkan?

A9: In fact, Biden was responding to a question not about defunding the police but about shifting some funding to social service agencies: "But do we agree that we can redirect some of the funding?" progressive activist Ady Barkan asked in a July 8 interview.

Q10: Does the phrase "defund the police" refer to eliminating police departments entirely or revisiting their functions and shifting funding to other services?

A10: While some argue police departments should be eliminated entirely, as PolitiFact National detailed in a June 9, 2020 article, the use of the phrase "defund the police" more typically means to revisit the functions of police departments and shift funding toward, for instance, mental health and social services.

Scores:

Hungarian METEOR	Ev ² R Q-only	Ev ² R Q+A	SemQA
0.375	1	1	1

Table 13: Example 10. Claim: "Biden has pledged to defund the police". All metrics show high score, and indicate an agreement.

Example 121 (Russian vaccine)

Claim: Russia has successfully developed a vaccine for Covid-19 and it has passed all checks.

Gold QA

Q1: Who has developed this vaccine?

A1: 'Sputnik V' has been developed by Moscow-based Gamaleya Research Institute of Epidemiology and Microbiology.

Q2: When was it first registered?

A2: It was registered by the Russian health ministry on August 11 as the first registered Covid-19 vaccine in the market but The certificate mentions that "the vaccine cannot be used widely until 1 January 2021", presumably after larger clinical trials are completed.

Q3: What kind of vaccine is it?

A3: The Russian vaccine is an "adenovirus vector-based vaccine". The idea is to use the weakened common cold virus to stimulate an immune response and trigger the formation of antibodies against Covid-19. These anti-bodies are then ready to fight against Covid-19 in case one catches a coronavirus infection. Even the widely awaited Oxford University vaccine is exploring the same route.

Q4: Has it been thoroughly tested?

A4: No

Q5: Has it been tested for safety?

A5: No

Generated QA

Q1: Has the COVID-19 vaccine developed by Russia completed all necessary clinical trials?

A1: Russia has not yet published any scientific data from its first clinical trials. The WHO's list of vaccine candidates in human testing still lists the Gamaleya product as in Phase 1 trials.

Q2: Has the COVID-19 vaccine developed by Russia completed a phase-three trial before its approval?

A2: But the vaccine's approval by Russian regulators comes before the completion of a larger study involving thousands of people, known as a phase-three trial.

Q3: Has the COVID-19 vaccine developed by Russia been proven to be safe and effective?

A3: Russian President Vladimir Putin announced the approval of a coronavirus vaccine for use on Tuesday, claiming it as a "world first," amid continued concern and unanswered questions over its safety and effectiveness.

Q4: How long did it take for the clinical trials of the Russian vaccine to be completed?

A4: Clinical trials of the Russian vaccine have been completed in less than two months and phase three trials are set to begin shortly, despite the vaccine having already been registered.

Q5: Has the COVID-19 vaccine developed by Russia been proven to be safe and effective?

A5: Scientists and public health officials are skeptical about Russian President Vladimir Putin's claim that the country's potential vaccine for the coronavirus "works quite effectively," saying Tuesday that the vaccine still needs critical testing to determine whether it's safe and effective.

Q6: How long did it take for the clinical trials of the Russian vaccine to be completed?

A6: Clinical trials of the Russian vaccine have been completed in less than two months and phase three trials are set to begin Wednesday, despite the vaccine having already been registered.

Q7: Does Russia have sufficient data to support the safety of its newly approved COVID-19 vaccine?

A7: Russian President Vladimir Putin says his country has become the first in the world to formally approve a new COVID-19 vaccine, despite a lack of data to back up his claims that the drug is safe to administer to humans.

Q8: Has the COVID-19 vaccine developed by the Gamaleya Institute in Moscow been thoroughly tested?

A8: Despite having only been in clinical trials for less than two months, the vaccine developed by the Gamaleya Institute in Moscow is safe, Putin said at a televised cabinet meeting, noting that it has already been given to one of his daughters, according to Reuters and The Washington Post.

Q9: What stages of clinical trials has the Russian Covid-19 vaccine completed?

A9: Few details have been made available about the Russian vaccine so far, but its approval was based on only the first two phases of clinical trials, which are designed to make early assessments on whether a potential vaccine can induce an immune response, and whether it's safe to administer in humans.

Q10: Has Russia's Covid-19 vaccine been proven to be safe and effective?

A10: Scientists are skeptical about Putin's claims on Russia's potential Covid-19 vaccine, saying the vaccine still needs critical data to determine whether it's safe or effective.

Scores:

Hungarian METEOR	Ev ² R Q-only	Ev ² R Q+A	SemQA
0.294	0.6	0.5	0.476

Table 14: Example 121. Claim: "Russia has successfully developed a vaccine for Covid-19 and it has passed all checks.". SemQA aligns closely with Ev²r, while Hungarian METEOR is much lower.

B Human Evaluation form for SemQA Evaluation

SemQA Human Evaluation

Evaluation set 1 of metric for automated fact checking systems

Thank you!

First of all, thank you for your participation! 🙏 Your answers in this questionnaire are valuable to us and will be used to enhance our analysis in the paper we are currently writing as part of our exam in Neural Methods for Natural Language Processing.

Context

The project is related to the use of LLMs for fact verification. When an LLM generates a response that is incorrectly decoded, not based on training data, or not following identifiable patterns, the response can be false or misleading. Fact-checking LLM outputs is time-consuming for humans, so automated fact-checking (AFC) systems were created to efficiently process large volumes of information and detect hallucinations. The metrics used to evaluate these AFC systems can be computationally intensive both in cost and time. We have investigated the benefits and drawbacks of these evaluation frameworks, and used this insight to create a new evaluation framework for AFC systems that is less computationally intensive. We call this metric SemQA (Semantic Question and Answer).

Why am I here?

AFC systems rely on quantitative metrics to evaluate performance, but these metrics do not consider nuances and might give unrepresentative scores. Human feedback can point out shortcomings and recognize the difference between harmless and critical mistakes made. Human evaluations allows for a more qualitative analysis of the performance of our SemQA metric.

How does it work?

In this project we have produced a dataset of claims, labels (supported/refuted/not enough evidence/etc...), generated/retrieved evidence, referenced evidence, and different types of metrics. SemQA calculates a score from 0-1 based on how semantically aligned the retrieved evidence is with the referenced evidence (semantically aligned in this context means how similar are the two texts). The more similar it is, the higher the score.

What will I do?

You will be given 10 questions. For each question you will be shown a set of retrieved evidence, referenced evidence, and the SemQA score. You will evaluate how accurate the SemQA metric is (in your opinion) on a scale from 1-7, where 1 = score should be much lower, 4 = score is accurate, 7 = score should be much higher 😊

Figure 3: Instructions to participants for human evaluations

Question 4

Referenced Evidence	Retrieved Evidence	SemQA Score
True, FBI statistics and a national database show that checks out: In 2019, police killed 999 people, and 48 officers were killed by a criminal act in the line of duty. We rate Larson's claim True.	The 2019 data from the Federal Bureau of Investigations shows that the ratio of law enforcement officers killed by criminals is not 20.8 times more likely to kill than be killed by a criminal.	0.55

Is the SemQA score accurate? *



Figure 4: Example question in evaluation set for human evaluations

[ID: 13] SemQA vs. Human Eval for Claim: Joe Biden voted for the Iraq War and he supported wars in Serbia, Syria, and Libya.

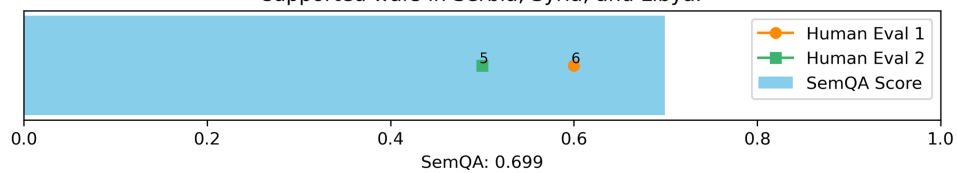


Figure 5: Comparison of SemQA score and human evaluations. Human evaluations of 5 and 6 means the SemQA score of 0.69 should have been slightly to moderately higher according to the annotators.

[ID: 5] SemQA vs. Human Eval for Claim: The Common Law Admission Test (CLAT) 2020 will not be conducted on September 7, 2020, as planned

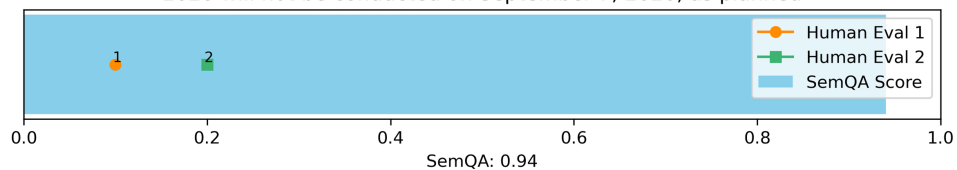


Figure 6: Comparison of SemQA score and human evaluations. Human evaluations of 1 and 2 means the SemQA score of 0.94 should have been far lower according to the annotators.

[ID: 4] SemQA vs. Human Eval for Claim: After the police shooting of Jacob Blake, Gov. Tony Evers & Lt. Gov. Mandela Barnes did not call for peace or encourage calm.

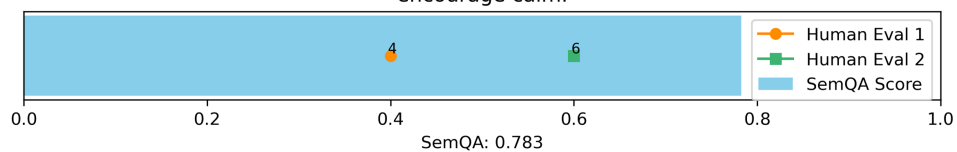


Figure 7: Comparison of SemQA score and human evaluations. Human evaluations of 4 and 6 means the annotators disagree; one marks the SemQA score of 0.78 as accurate, while the other marks the score as too low.