Large Language Models

Part 1: Pre-training

David Samuel

Language Technology Group, University of Oslo



Table of contents



1. Pre-training / fine-tuning paradigm

2. Subword tokenization

3. Decoder language models

4. Encoder language models

5. Encoder-decoder language models

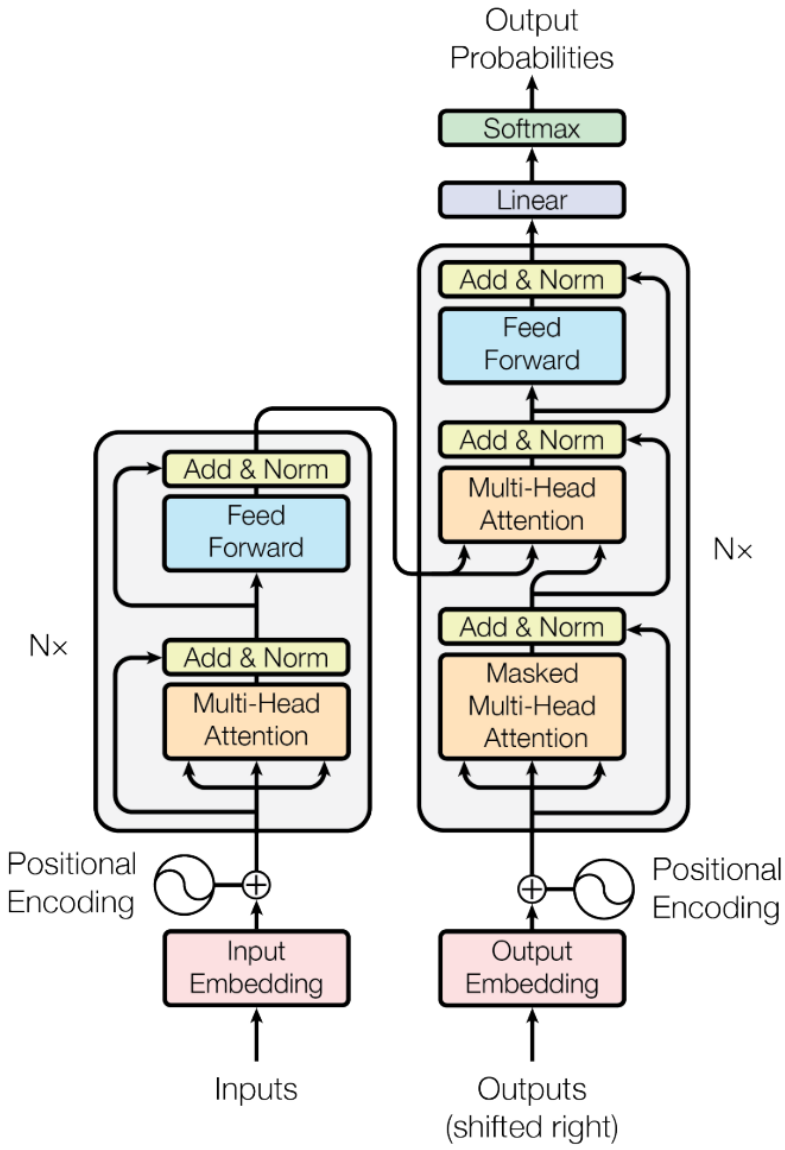
6. References

Pre-training / fine-tuning paradigm

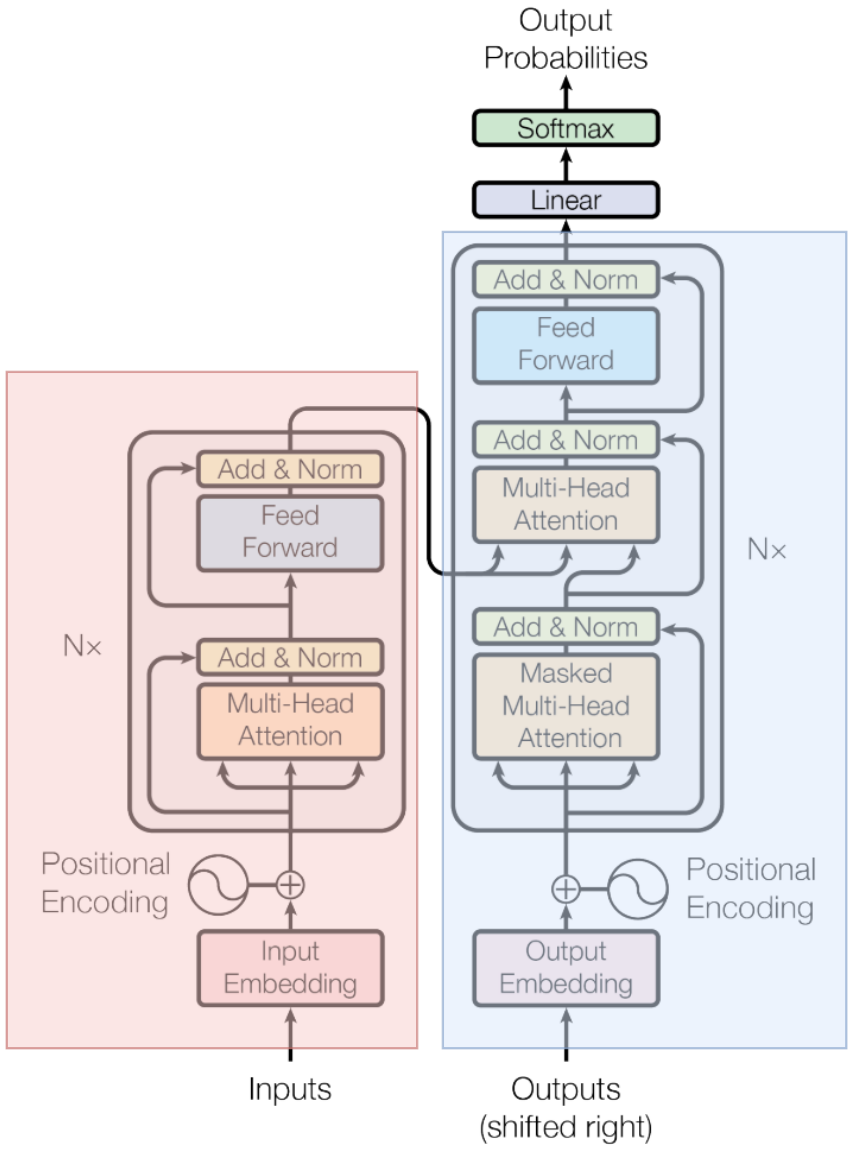
We are building upon concepts introduced in previous lectures

- Contextualized word embeddings
- Representation learning
- Transfer learning
- Transformers
  - **encoder**: *encodes* the source text
  - **decoder**: takes the encoded text and *decodes* the target text token by token

Pre-training / fine-tuning paradigm



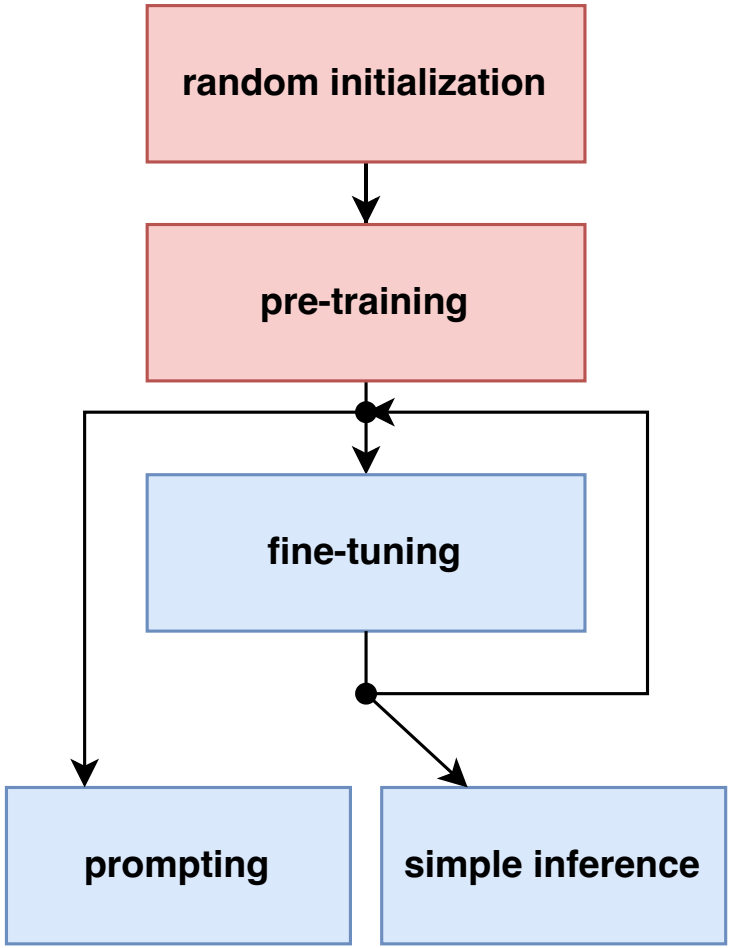
Pre-training / fine-tuning paradigm



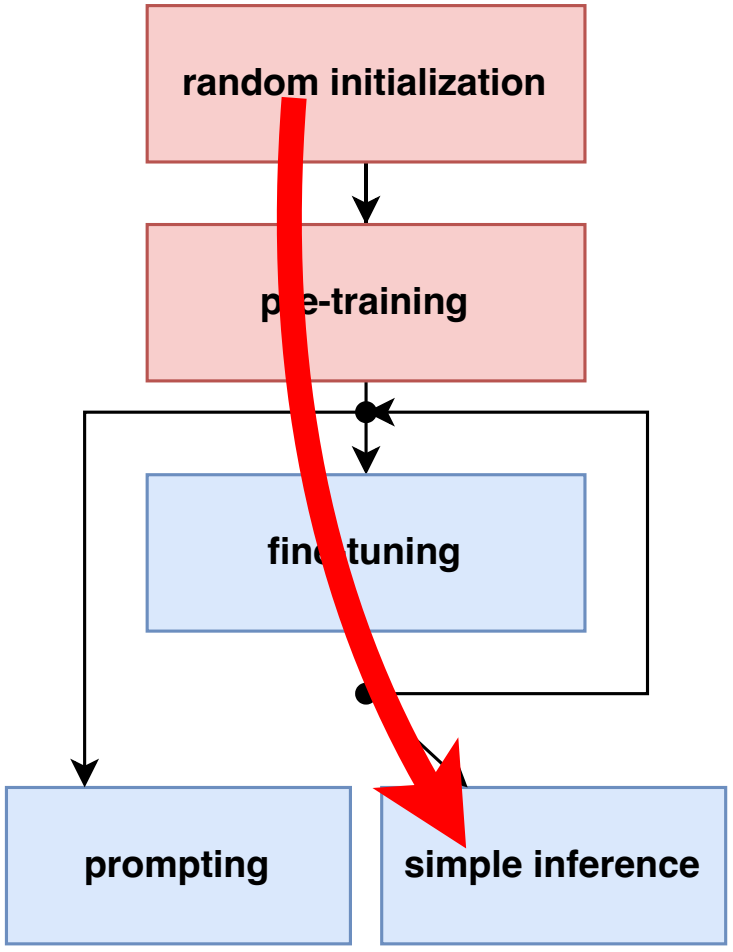
Encoder

Decoder

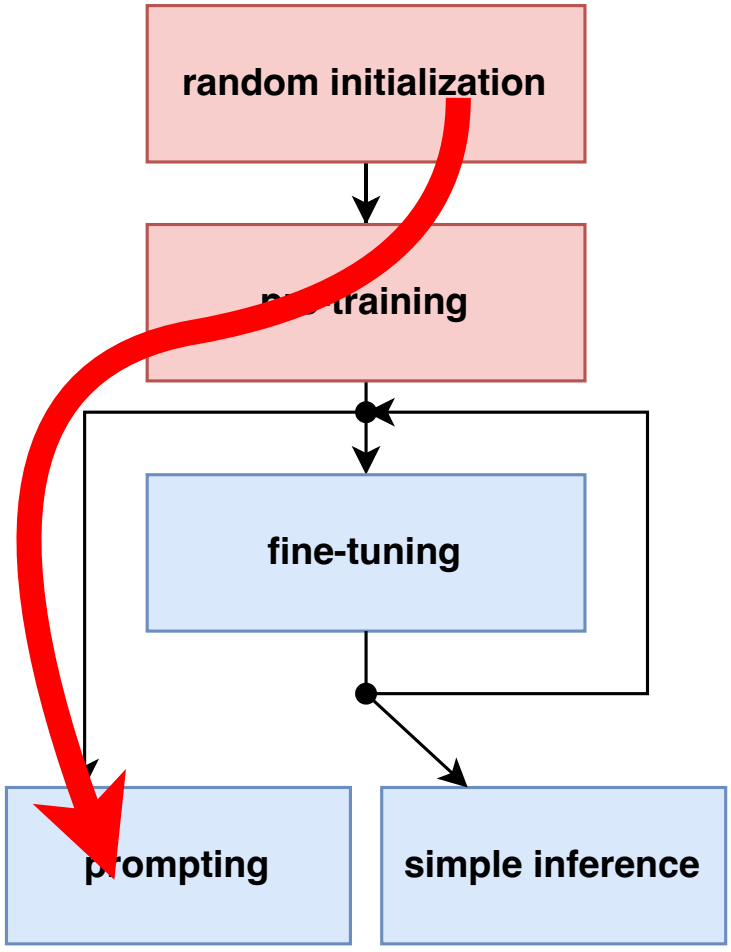
Pre-training / fine-tuning paradigm



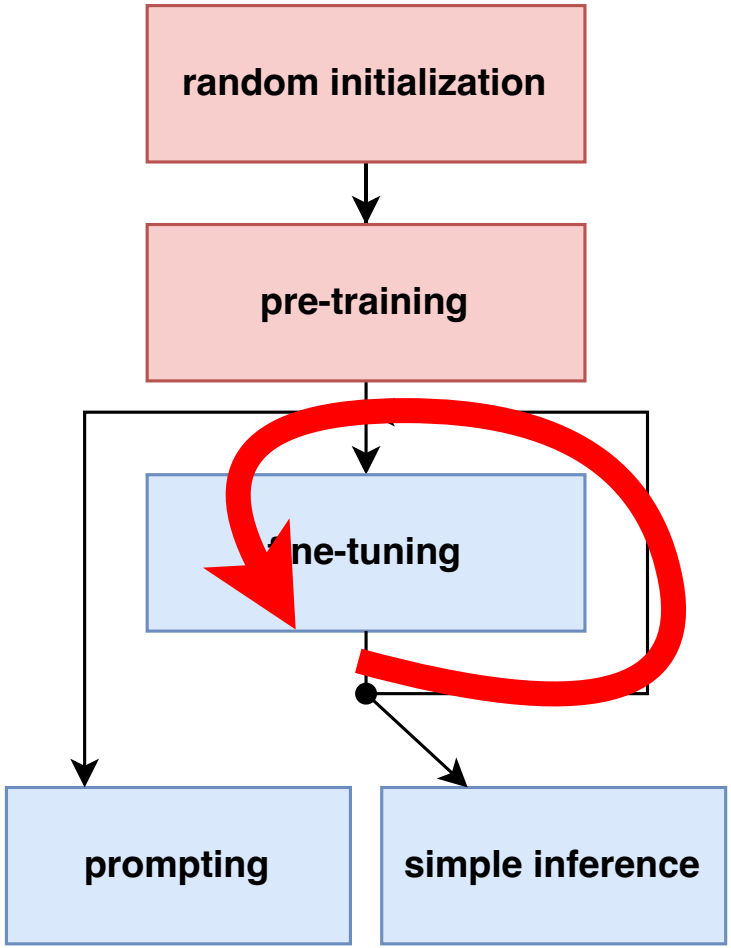
Pre-training / fine-tuning paradigm



Pre-training / fine-tuning paradigm



Pre-training / fine-tuning paradigm



Pre-training / fine-tuning paradigm

1st part:
pre-training

2nd part:
fine-tuning
and inference

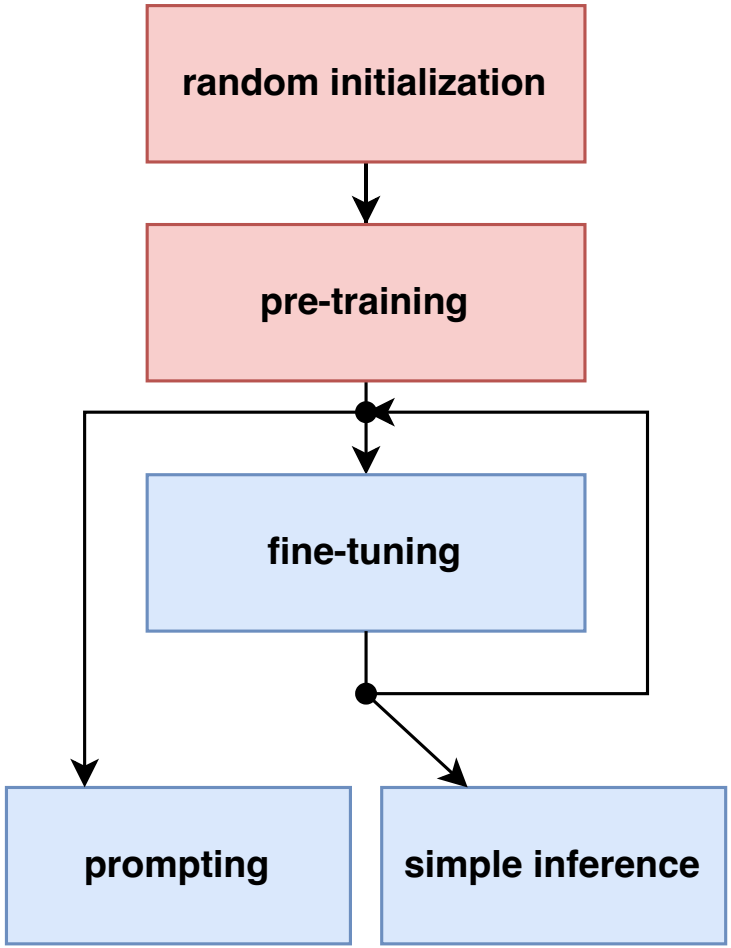


Table of contents



1. Pre-training / fine-tuning paradigm
2. Subword tokenization
3. Decoder language models
4. Encoder language models
5. Encoder-decoder language models
6. References

Subword tokenization

- Neural machine translation has popularized the use of subword tokenization
- Word tokens are no longer used in modern language models
- There is a near-infinite amount of possible word forms
  - Out of vocabulary words are frequent even for static word embeddings with millions of words in their vocabularies
- Subwords require vocabularies with only 10 000s of tokens without introducing any out-of-vocabulary errors

Byte-pair encoding (BPE; Gage, 1994) – training

Text corpus: *"AABCCAABAABCC"*

- s_0: *"A"*
- s_1: *"B"*
- s_2: *"C"*



Byte-pair encoding (BPE; Gage, 1994) – training

Text corpus: 0 0 1 2 2 0 0 1 0 0 1 2 2

- s_0: *"A"*
- s_1: *"B"*
- s_2: *"C"*

Byte-pair encoding (BPE; Gage, 1994) – training

Text corpus: 0 **0 1** 2 2 0 **0 1** 0 **0 1** 2 2

- s_0: *"A"*
- s_1: *"B"*
- s_2: *"C"*
- s_3: $s_0 + s_1$ = *"AB"*

Byte-pair encoding (BPE; Gage, 1994) – training

Text corpus: 0 3 2 2 0 3 0 3 2 2

- s_0: *"A"*
- s_1: *"B"*
- s_2: *"C"*
- s_3: $s_0 + s_1$ = *"AB"*
- s_4: $s_0 + s_3$ = *"AAB"*

Byte-pair encoding (BPE; Gage, 1994) – training

Text corpus: 4 2 2 0 3 4 2 2

- s_0: *"A"*
- s_1: *"B"*
- s_2: *"C"*
- s_3: $s_0 + s_1$ = *"AB"*
- s_4: $s_0 + s_3$ = *"AAB"*

Byte-pair encoding (BPE; Gage, 1994) – training

Text corpus: 4 2 2 0 3 4 2 2

- s_0: *"A"*
- s_1: *"B"*
- s_2: *"C"*
- s_3: $s_0 + s_1$ = *"AB"*
- s_4: $s_0 + s_3$ = *"AAB"*
- s_5: $s_2 + s_2$ = *"CC"*

Byte-pair encoding (BPE; Gage, 1994) – training

Text corpus: 4 5 0 3 4 5

- s_0: *"A"*
- s_1: *"B"*
- s_2: *"C"*
- s_3: $s_0 + s_1$ = *"AB"*
- s_4: $s_0 + s_3$ = *"AAB"*
- s_5: $s_2 + s_2$ = *"CC"*

Byte-pair encoding (BPE; Gage, 1994) – training

Text corpus: 4 5 0 3 4 5

- s_0: *"A"*
- s_1: *"B"*
- s_2: *"C"*
- s_3: $s_0 + s_1$ = *"AB"*
- s_4: $s_0 + s_3$ = *"AAB"*
- s_5: $s_2 + s_2$ = *"CC"*
- s_6: $s_4 + s_5$ = *"AABCC"*

Byte-pair encoding (BPE; Gage, 1994) – training

Text corpus: 6 0 3 6

- s_0: *"A"*
- s_1: *"B"*
- s_2: *"C"*
- s_3: $s_0 + s_1$ = *"AB"*
- s_4: $s_0 + s_3$ = *"AAB"*
- s_5: $s_2 + s_2$ = *"CC"*
- s_6: $s_4 + s_5$ = *"AABCC"*

Byte-pair encoding (BPE; Gage, 1994)– tokenization

Raw text: *"CCAABCC"*

Vocabulary / merge list

- s_0: *"A"*
- s_1: *"B"*
- s_2: *"C"*
- s_3: $s_0 + s_1$ = *"AB"*
- s_4: $s_0 + s_3$ = *"AAB"*
- s_5: $s_2 + s_2$ = *"CC"*
- s_6: $s_4 + s_5$ = *"AABCC"*

Byte-pair encoding (BPE; Gage, 1994) – tokenization

Raw text: *"CCAABCC"*

- 2 2 0 0 1 2 2

Vocabulary / merge list

- s_0: *"A"*
- s_1: *"B"*
- s_2: *"C"*
- s_3: $s_0 + s_1$ = *"AB"*
- s_4: $s_0 + s_3$ = *"AAB"*
- s_5: $s_2 + s_2$ = *"CC"*
- s_6: $s_4 + s_5$ = *"AABCC"*

Byte-pair encoding (BPE; Gage, 1994) – tokenization

Raw text: *"CCAABCC"*

- 2 2 0 0 1 2 2
- 2 2 0 3 2 2

Vocabulary / merge list

- s_0: *"A"*
- s_1: *"B"*
- s_2: *"C"*
- s_3: $s_0 + s_1$ = *"AB"*
- s_4: $s_0 + s_3$ = *"AAB"*
- s_5: $s_2 + s_2$ = *"CC"*
- s_6: $s_4 + s_5$ = *"AABCC"*

Byte-pair encoding (BPE; Gage, 1994) – tokenization

Raw text: *"CCAABCC"*

- 2 2 0 0 1 2 2
- 2 2 0 3 2 2
- 2 2 4 2 2

Vocabulary / merge list

- s_0: *"A"*
- s_1: *"B"*
- s_2: *"C"*
- s_3: $s_0 + s_1$ = *"AB"*
- s_4: $s_0 + s_3$ = *"AAB"*
- s_5: $s_2 + s_2$ = *"CC"*
- s_6: $s_4 + s_5$ = *"AABCC"*

Byte-pair encoding (BPE; Gage, 1994) – tokenization

Raw text: *"CCAABCC"*

- 2 2 0 0 1 2 2
- 2 2 0 3 2 2
- 2 2 4 2 2
- 5 4 5

Vocabulary / merge list

- s_0: *"A"*
- s_1: *"B"*
- s_2: *"C"*
- s_3: $s_0 + s_1$ = *"AB"*
- s_4: $s_0 + s_3$ = *"AAB"*
- s_5: $s_2 + s_2$ = *"CC"*
- s_6: $s_4 + s_5$ = *"AABCC"*

Byte-pair encoding (BPE; Gage, 1994) – tokenization

Raw text: *"CCAABCC"*

- 2 2 0 0 1 2 2
- 2 2 0 3 2 2
- 2 2 4 2 2
- 5 4 5
- 5 6

→ 2 tokens: *"CC"* + *"AABCC"*

Vocabulary / merge list

- s_0: *"A"*
- s_1: *"B"*
- s_2: *"C"*
- s_3: $s_0 + s_1$ = *"AB"*
- s_4: $s_0 + s_3$ = *"AAB"*
- s_5: $s_2 + s_2$ = *"CC"*
- s_6: $s_4 + s_5$ = *"AABCC"*



Practical considerations

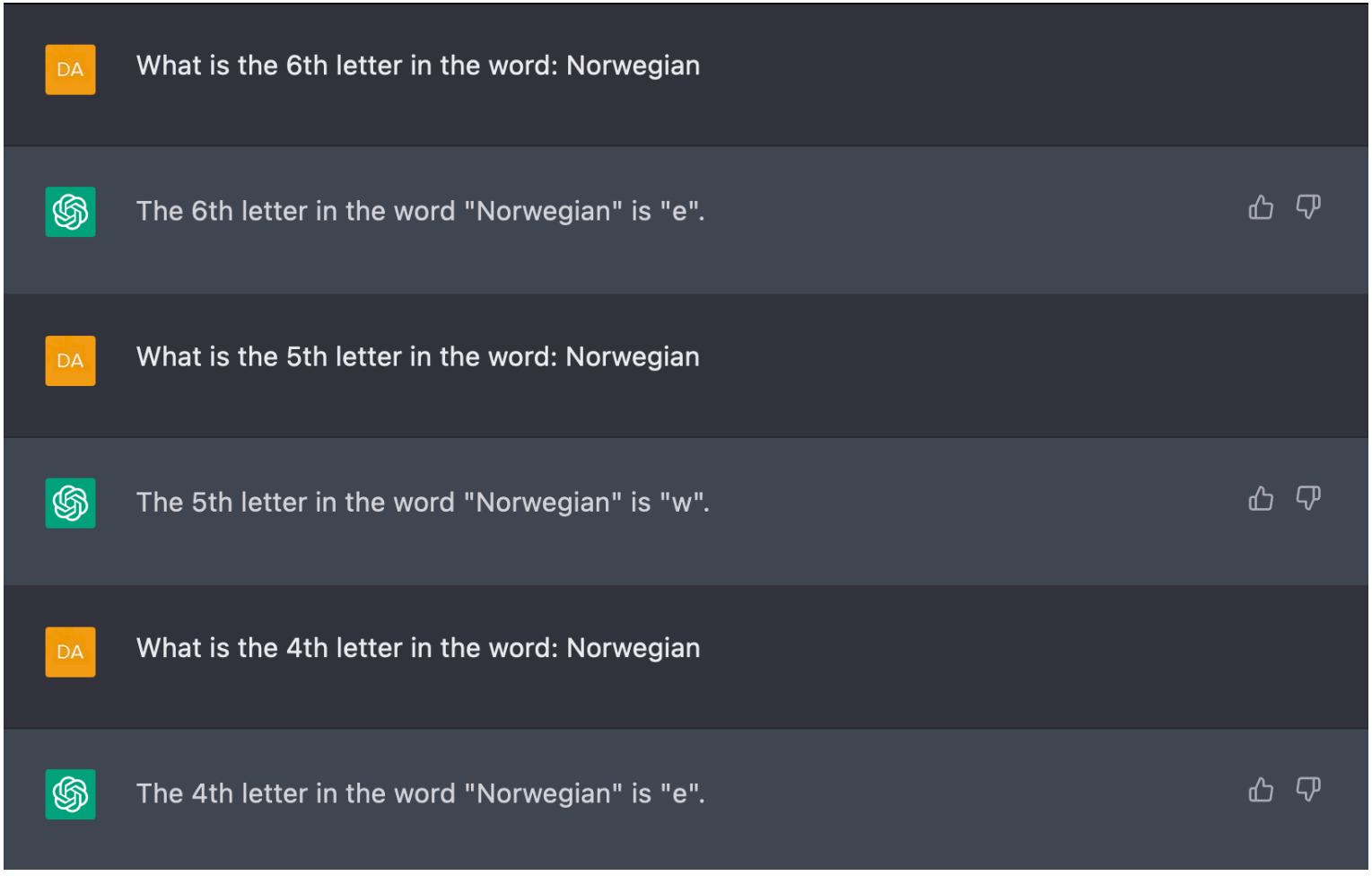
- So-called "pre-tokenization" step (actually our good old word tokenization) is performed first, usually
- We need a mechanism to tell apart splits between words and splits inside words
- Letters are not a good atomic unit (UTF-8 can encode 1 112 064 valid characters) – recent systems use UTF-8 bytes as the atomic unit
- Improved training algorithms: WordPiece (Wu et al., 2016), Unigram tokenizer (Kudo, 2018) or Morfessor systems (Creutz and Lagus, 2007)

Subword tokenization

Examples

- BERT: *“Ex ##ample of token ##ization for IN ##55 ##50 .”*
- XLM-R: *“_Exam ple _of _to ken ization _for _IN 55 50 .”*
- ChatGPT: *“Example Ġof Ġtoken ization Ġfor ĠIN 55 50 .”*

Subword tokenization



Subword tokenization



Alternative: character or byte tokenization

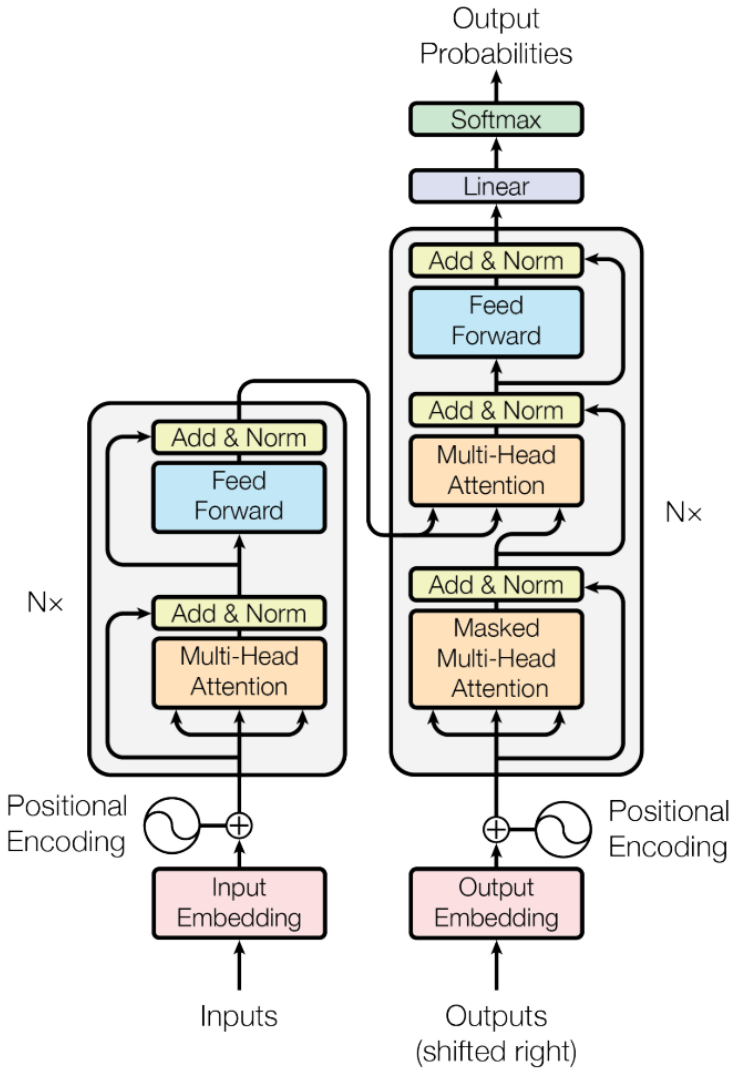
- We lose explicit information about characters inside words
- An alternative is to simply tokenize text by characters (or even by UTF-8 bytes)
- +: small vocabulary and no information loss
- −: long sequences → resource-intesive computation
- Appears in recent language models like CANINE (Clark et al., 2022) or ByT5 (Xue et al., 2022)

Table of contents

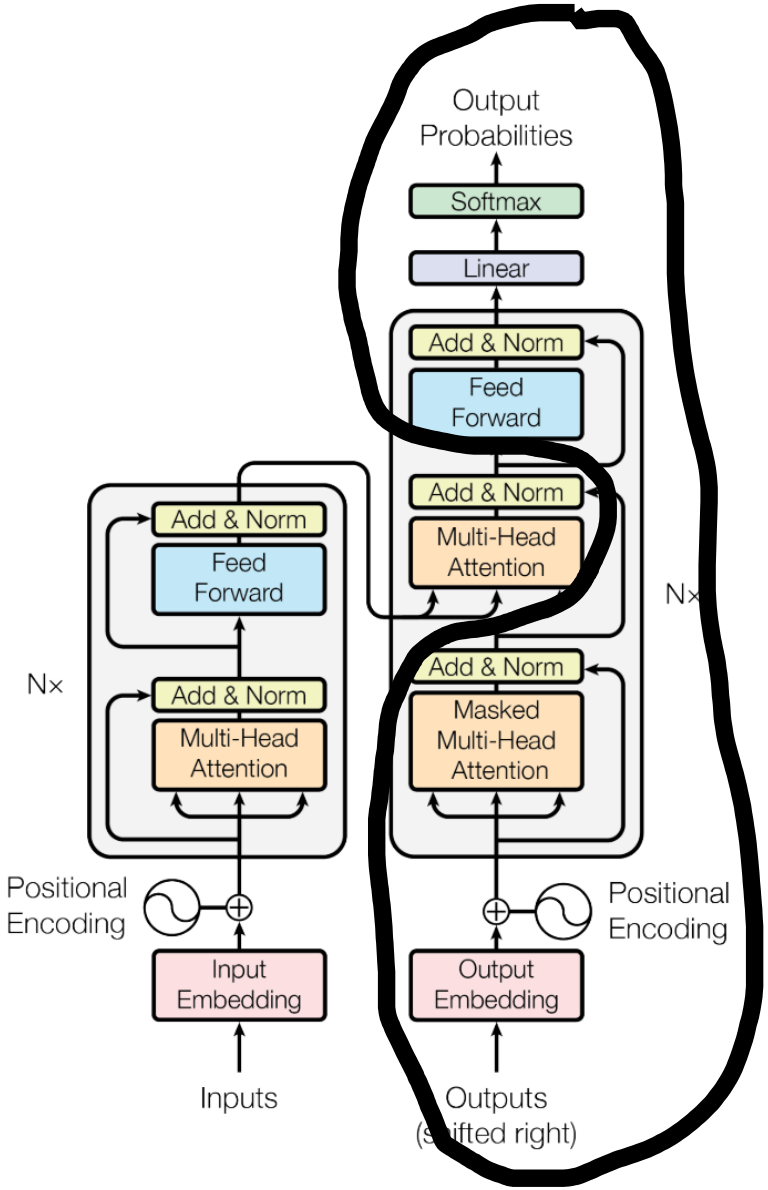


1. Pre-training / fine-tuning paradigm
2. Subword tokenization
3. Decoder language models
4. Encoder language models
5. Encoder-decoder language models
6. References

Decoder language models



Decoder language models



Decoder language models

- Training objective: next word prediction
- Alternative names: causal language models (CLMs), generative LMs, autoregressive LMs, left-to-right LMs, GPT-like LMs (very popular term, unfortunately) or simply “language models”
- Nothing surprising here, these are the traditional language models introduced by Claude E. Shannon (1948)

$$P(w_{0:n}) = \prod_{i=0}^{n} P(w_i | w_{0:i-1})$$

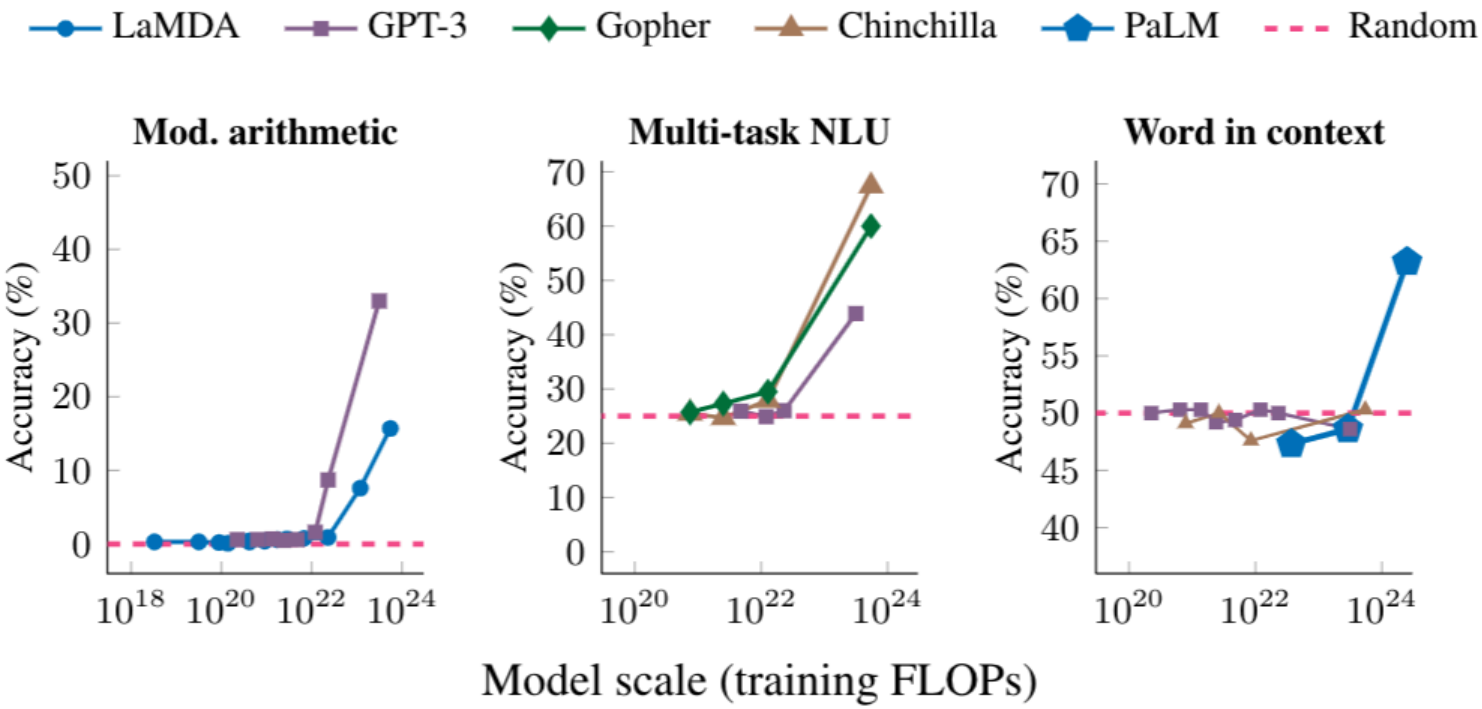
- We want to estimate the probability distribution of sequences of words (or tokens in general)

GPT

- A giant leap forwards in history: Bengio et al. (2001) proposed *"A Neural Probabilistic Language Model"*
- Another huge jump forward – the same thing, but now with transformers: *"Improving Language Understanding by Generative Pre-Training"* by Radford et al. (2018) – the first GPT
  - The idea here is to train a transformer on CLM and then fine-tune the whole thing on a downstream task
- Same architecture in GPT-2, just bigger (Radford et al., 2019)
  - Now the model starts to solve* downstream tasks without any fine-tuning!
    - * better than random prediction

Emergent abilities

- The same thing repeated with GPT-3 (Brown et al., 2020)
  - When the size of the model grows even more (0.1B → 1.5B → 175B), the model starts to be really good in zero-shot and few-shot
  - So-called “emergent abilities” (Wei et al., 2022)



Emergent abilities

- Now, we have GPT-4 (OpenAI, 2023)
  - but: *"Given both the competitive landscape and the safety implications of large-scale models like GPT-4, this report contains no further details about the architecture (including model size), hardware, training compute, dataset construction, training method, or similar."*
  - we can't really say anything about that model, this is no longer scientific research ¯_(ツ)_/¯

Emergent abilities

GPT-4 Technical Report

OpenAI*

Abstract

we used python



Emergent abilities

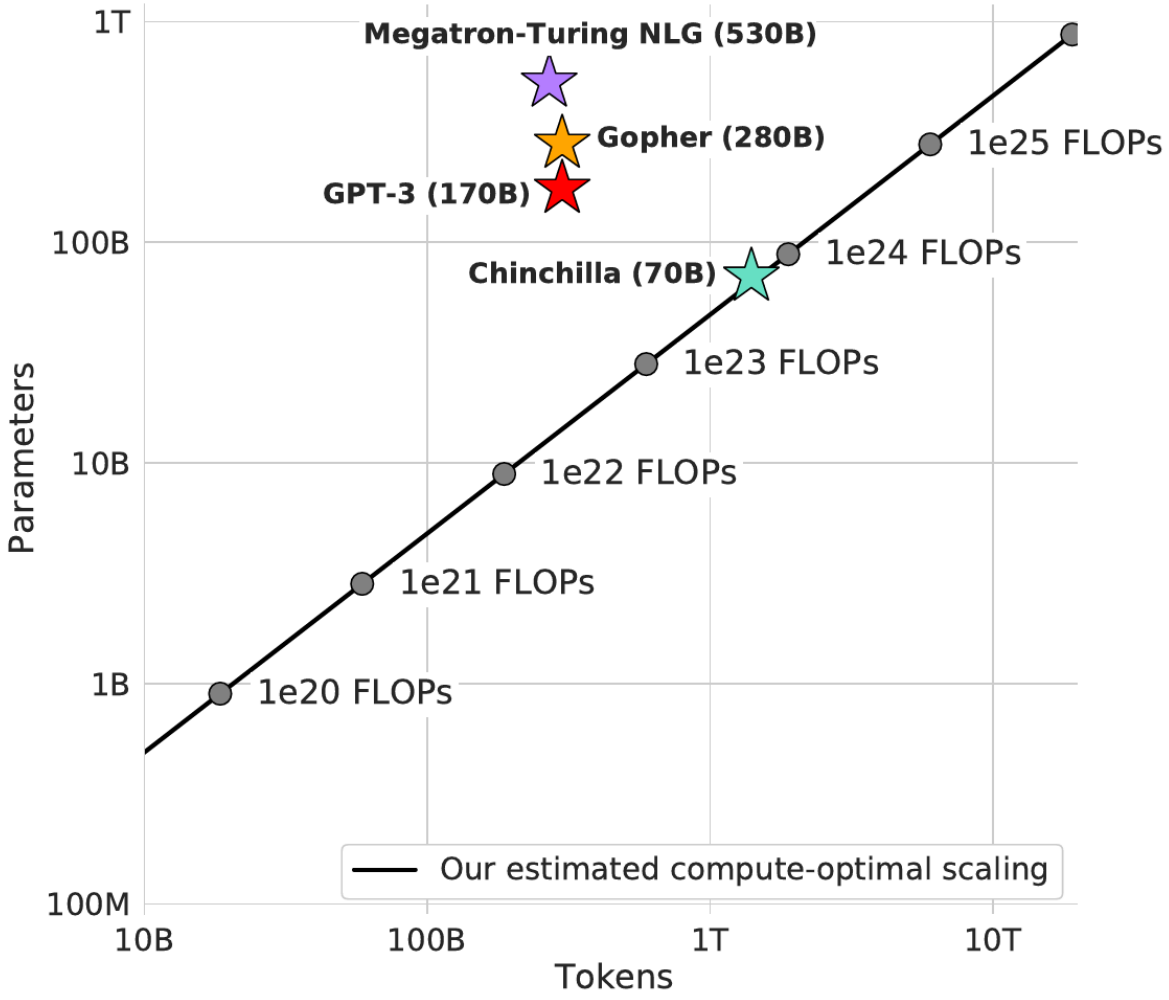
- A double-edged sword for decoder language models
- Their main advantage, otherwise they perform poorly in the pre-training / fine-tuning paradigm
- Therefore they have to be huge and expensive to be somewhat useful
- In practice: is it cheaper to annotate a dataset or to infer from a huge decoder LM?
  - A small fine-tuned LM usually performs better than a huge non-fine-tuned decoder LM (Bang et al., 2023)



GPT and its friends

- A lot of media attention in this field so big-tech invests a lot to train these huge models
- PaLM (Chowdhery et al., 2022): Ha, we trained a 540B LM and it's slightly better!
- Chinchilla (Hoffmann et al., 2022): Ha, we can do better with just 70B parameters!
  - Not only that, they showed the compute-optimal training settings
  - Turns out we are bottlenecked by data, not by model size!
  - Sadly (?), we will not get GAI just by scaling up text transformers, the Internet simply doesn't contain enough text

GPT and its friends



GPT and its open friends

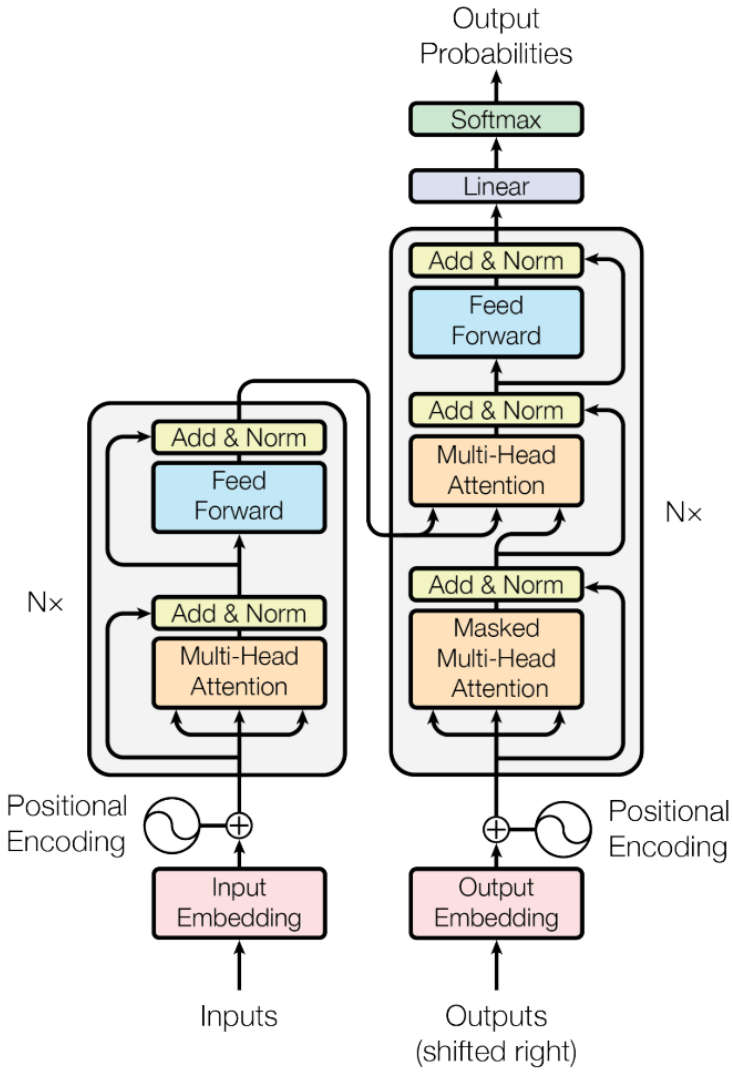
- The scientific contributions of these models are questionable
  - the papers are usually not peer-reviewed or even don't exist at all (ChatGPT)
  - the models are not released and it's thus impossible to reproduce and verify the results
- Still, there are some initiatives and research labs that try to change that
  - GPT-J (Wang and Komatsuzaki, 2021)
  - OPT (Zhang et al., 2022)
  - LLaMA (Touvron et al., 2023)
  - ...and many more

Table of contents

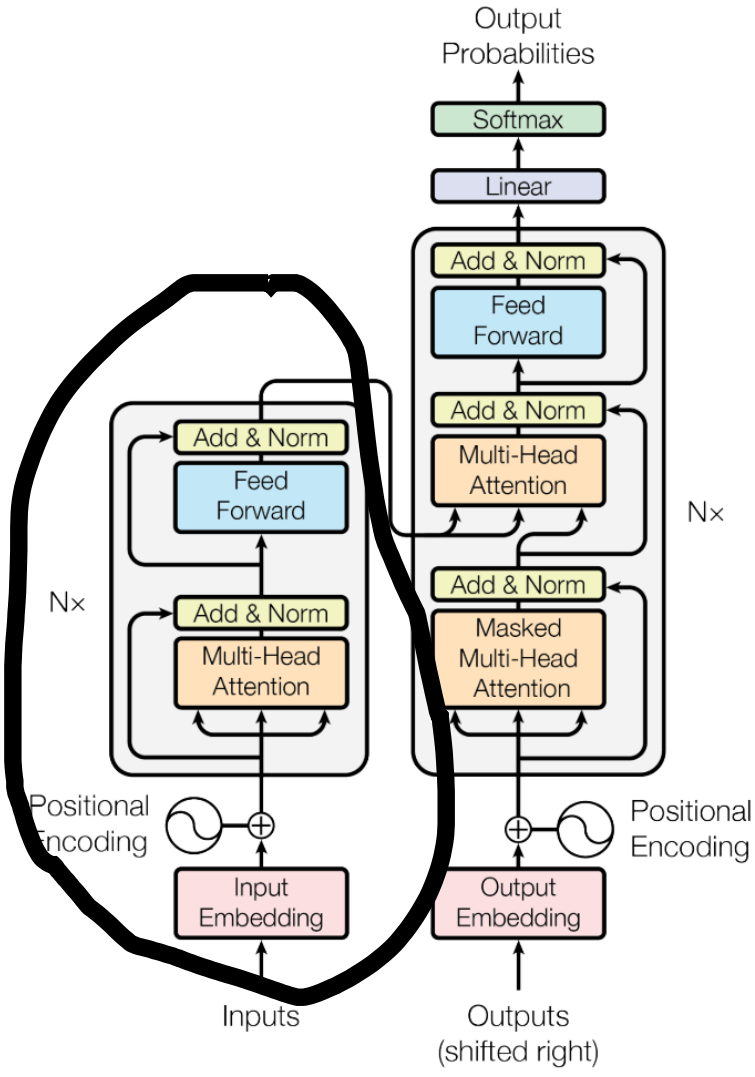


1. Pre-training / fine-tuning paradigm
2. Subword tokenization
3. Decoder language models
4. Encoder language models
5. Encoder-decoder language models
6. References

Encoder language models



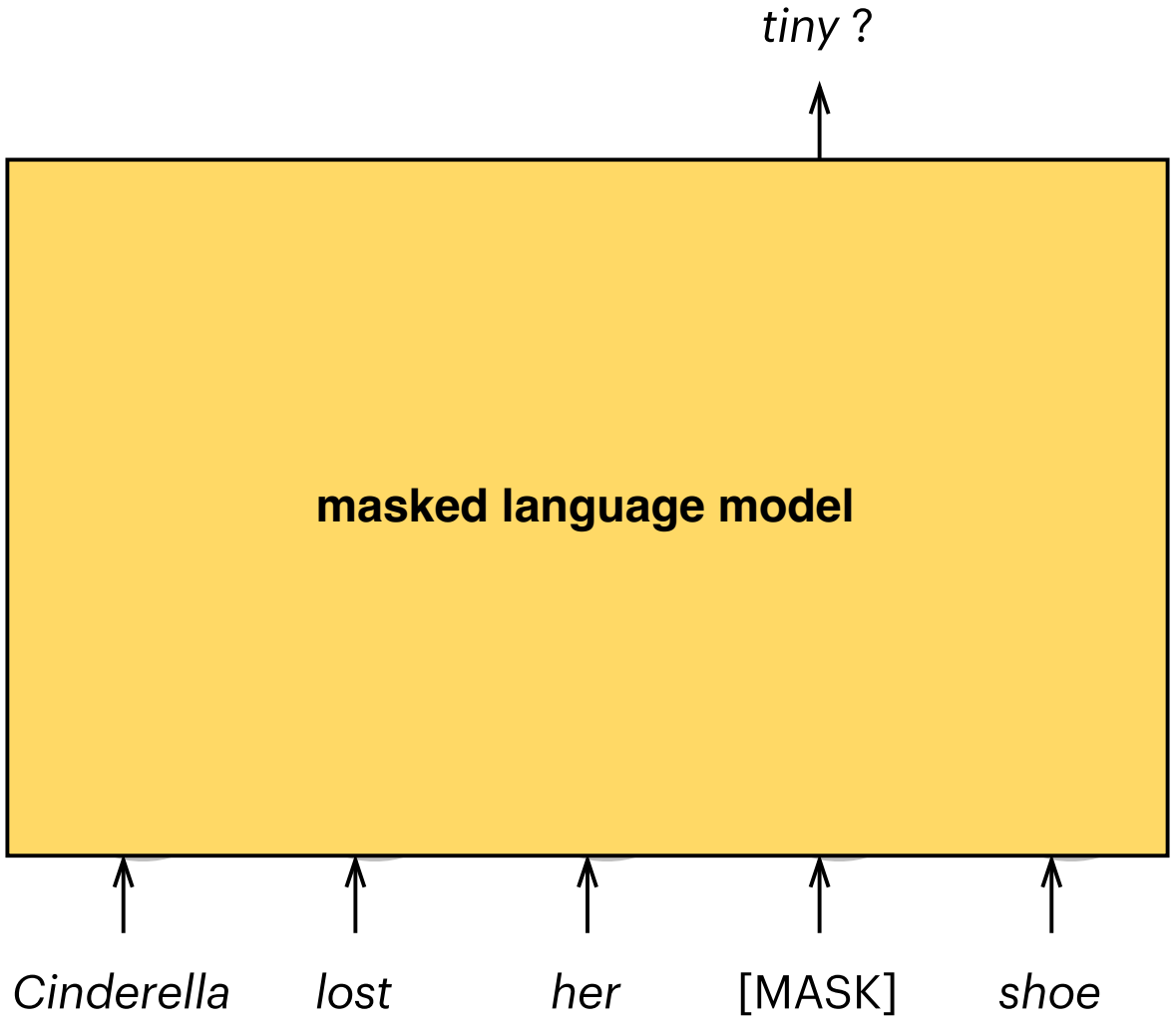
Encoder language models



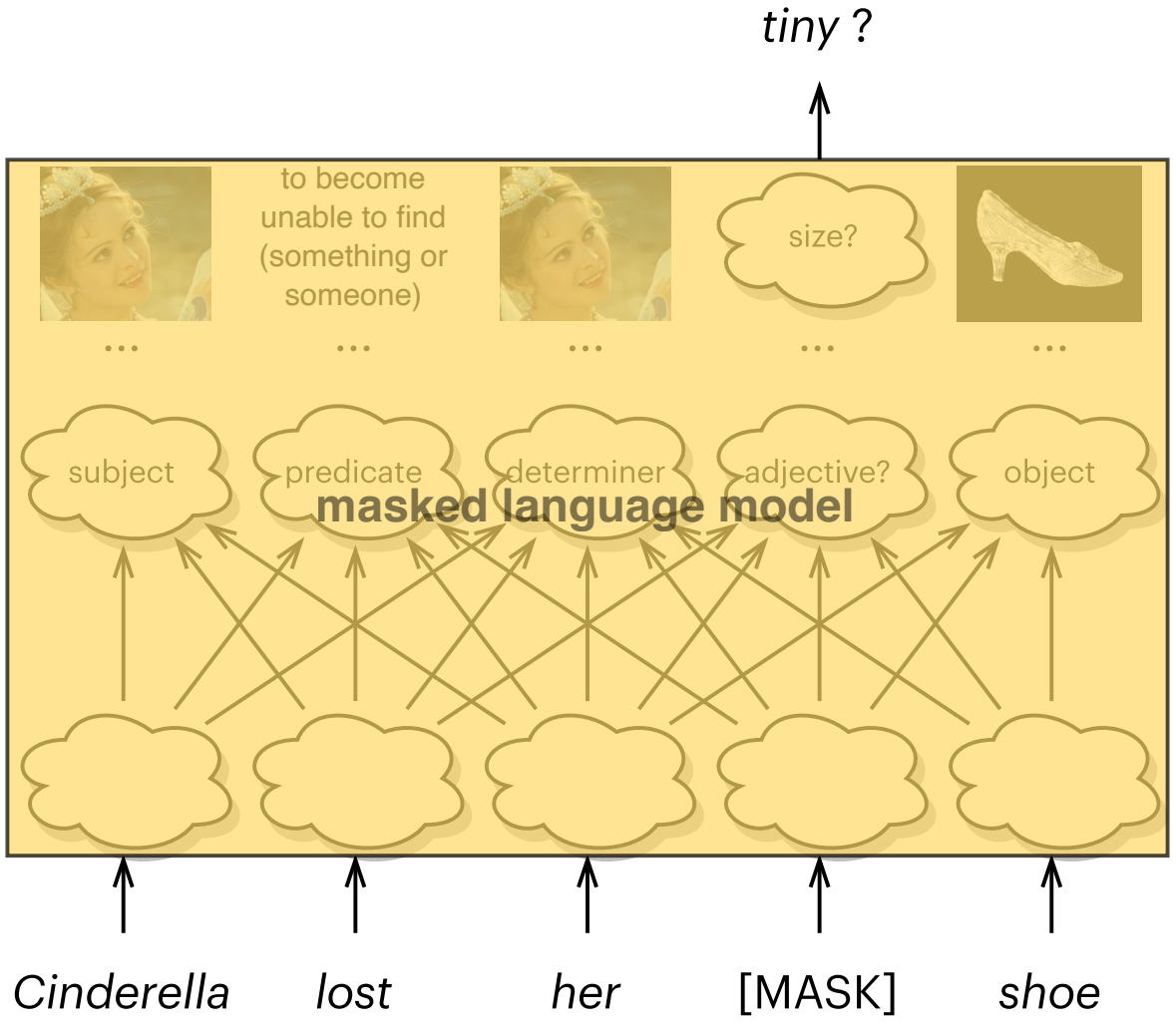
Encoder language models

- Training objective: fill in the blanks
  - "*Cinderella lost her ____ shoe.*"
- Alternative names: masked language models (MLMs), bidirectional LMs or BERT-like LMs
- We want to process text in both directions
- This results in strong language representation by the encoder
  - Then the encoder can be easily fine-tuned and generally performs better than a decoder language model
- However, the encoder doesn't directly estimate $P(w_{0:n})$ and doesn't generate text out of the box
  - Generating is possible but very impractical: Wang and Cho (2019)

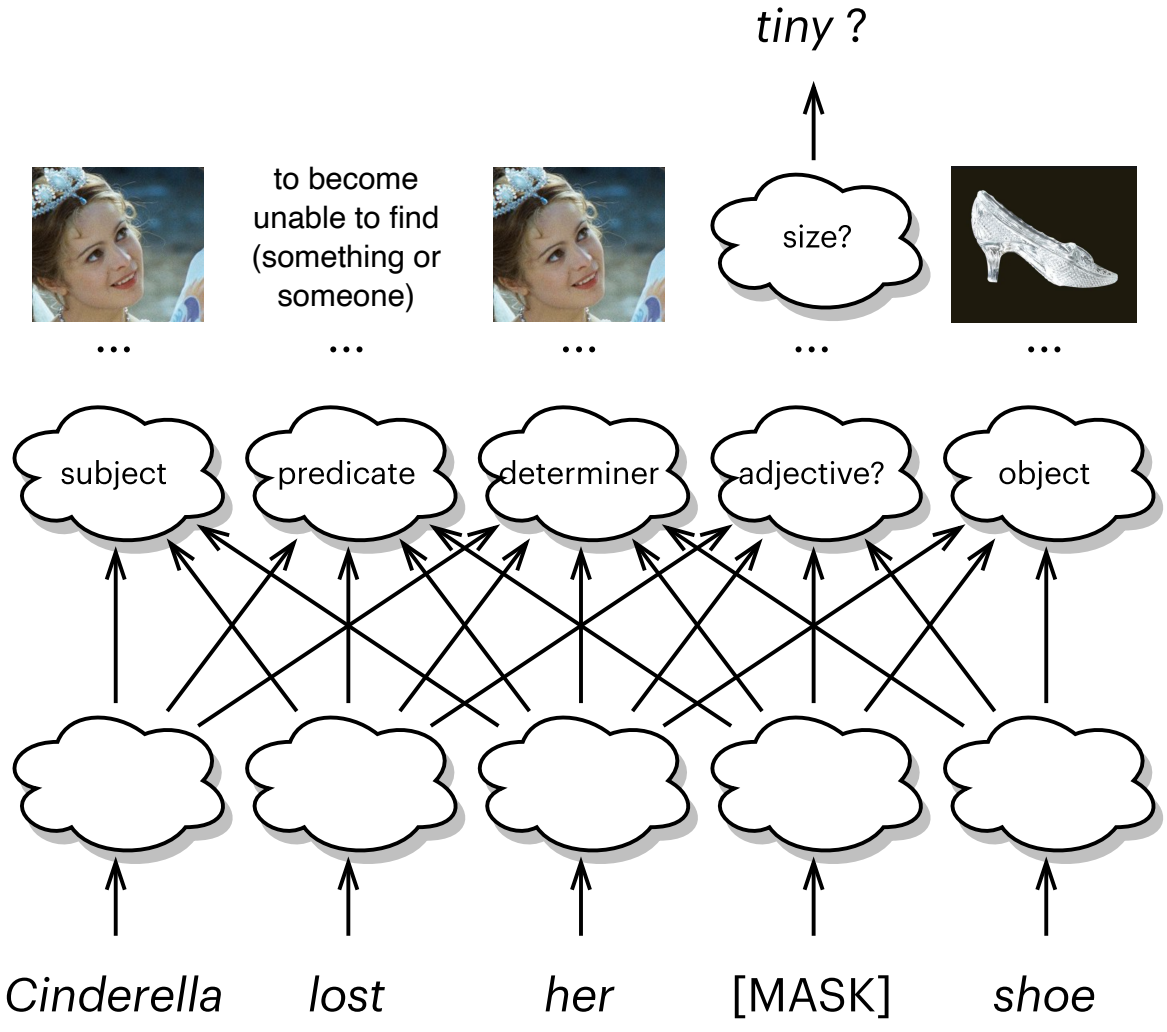
Representation learning with encoder language models



Representation learning with encoder language models



Representation learning with encoder language models





Representation learning with encoder language models

- BERT Rediscovers the Classical NLP Pipeline (Tenney et al., 2019)
  - MLMs work, layer-by-layer, similarly to a traditional NLP pipeline - first they processes the morphology, then the syntax, then the semantics
- A whole field of BERTology (Rogers et al., 2020), which investigates the inner workings of LMs – almost like a neuroscientist who studies brain



BERT and its friends

- Bidirectional Encoder Representations from Transformers (BERT; Devlin et al., 2019)
- Unlike ELMo, BERT is *deeply* bidirectional and transformer-based language model
  - This improvement led to unprecedented downstream performance back in 2018

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

BERT and its friends

- BERT is pre-trained on two objectives at the same time: masked language modeling (MLM) and next sentence prediction (NSP)
  - The input: *[CLS] text$_1$… [SEP] text$_2$… [SEP]*
  - The task: predict if text$_2$ follows text$_1$ or not, using the last hidden representation of the *[CLS]* token
  - The idea is to learn a good sentence representation, not just token representations
- BERT is pre-trained on a corpus of English books and on English Wikipedia, a relatively modest amount of text in today's standards



BERT and its friends

- RoBERTa (Liu et al., 2019) polished BERT and put it on another level
  - Lots of small improvements, nothing groundbreaking
  - Main findings: BERT is underfitted, it should be trained on much more data for many more steps, and next sentence prediction is unnecessary
  - Pre-trained on English CommonCrawl (filtered), in addition to the BERT corpora
  - Still a very strong baseline, many years after its publication!
- Many more *BERTs followed, bringing another improvements here and there; for example DeBERTa (He et al., 2020), which improves BERT by using relative positional embeddings



BERT and its friends

- So far, we only covered English MLMs, but there are lots of multilingual and non-English monolingual models
- The first one is mBERT, multilingual BERT pre-trained on 104 languages
  - Noteworthy because it showed that it is possible to learn on this many languages
  - Otherwise, it is a rather weak baseline
- XLM-R (Conneau et al., 2020), multilingual RoBERTa, is a much better model – often outperforming monolingual MLMs
- Monolingual MLMs now cover many languages, including Norwegian :)
  - For example NorBERT (Kutuzov et al., 2021) trained by our group

BERT and its friends

- ELECTRA (Clark et al., 2020) is one of the more interesting alternatives to MLMs
  - a more efficient pre-training by using two encoders: generator and discriminator
  - the generator tries to fool the discriminator
  - Difficult to set up properly in practice

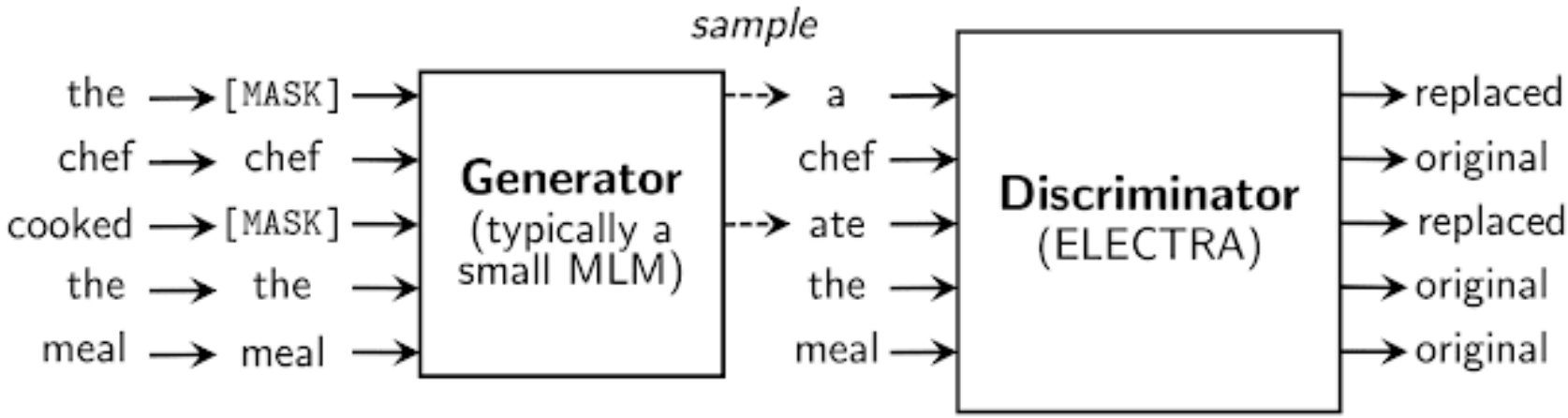
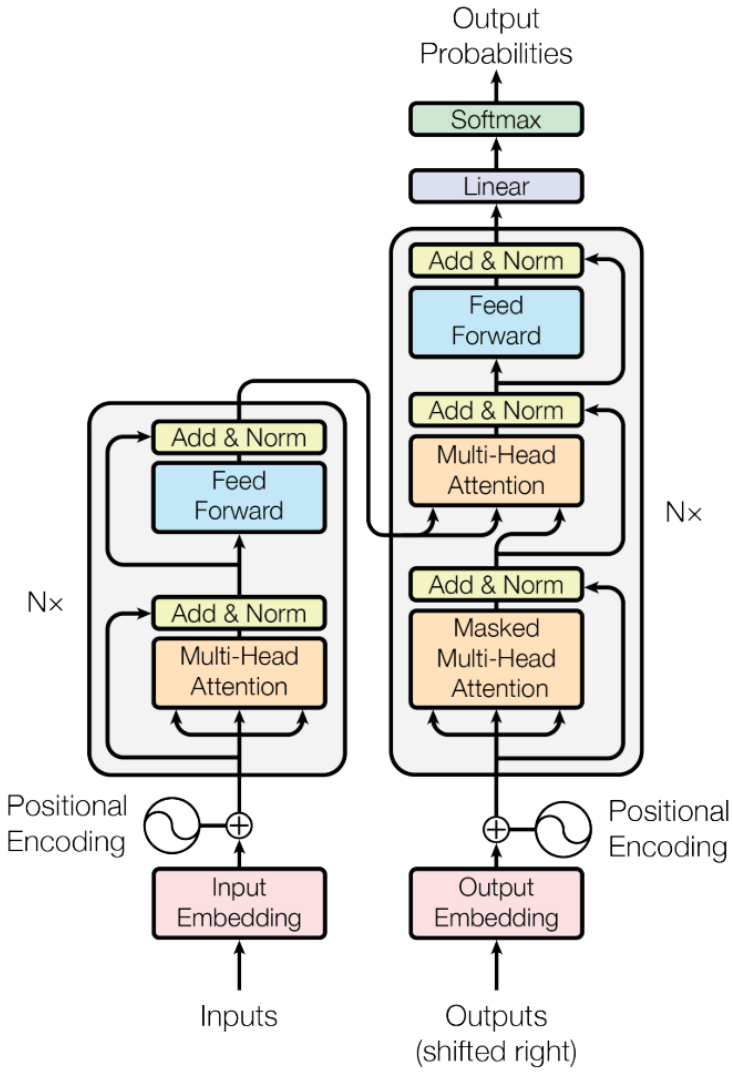


Table of contents

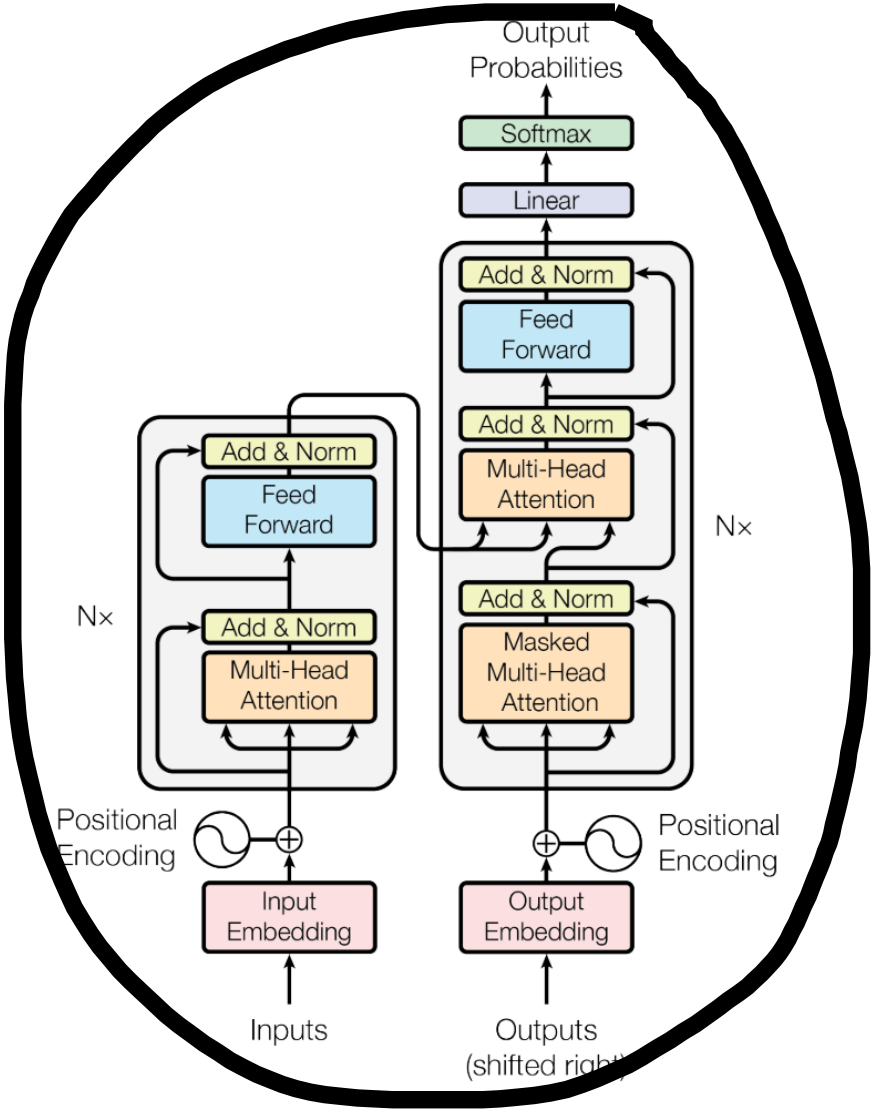


1. Pre-training / fine-tuning paradigm
2. Subword tokenization
3. Decoder language models
4. Encoder language models
5. Encoder-decoder language models
6. References

Encoder-decoder language models

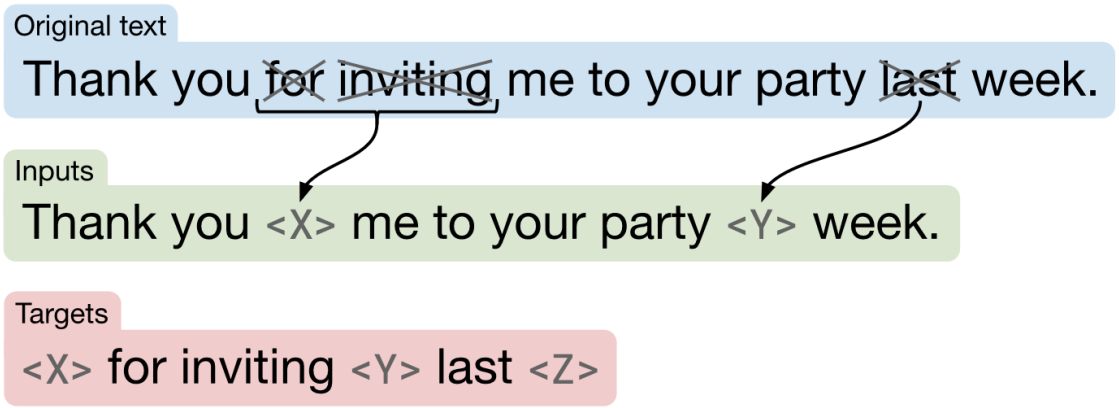


Encoder-decoder language models



Encoder-decoder language models

- Training objective: fill in the blanks, autoregressively



- The training objective combines MLM and CLM together
- We mask a span of tokens, replace it with a single MASK token, put the text into encoder and then let decoder generate all tokens in the masked span
- The resulting LM can be used as a strong representation learner as well as a generative model



T5 family

- Text-to-text Transfer Transformers (T5; Raffel et al., 2019)
  - Also multilingual mT5 (Xue et al., 2021) and byte-level ByT5 (Xue et al., 2022)
- A rule of a thumb: encoder-decoder models are worse than MLMs on fine-tuned tasks and better than CLMs on generative tasks, when comparing models of similar scale
  - However, the encoder-decoder architecture is more complicated to train and to scale up
  - Thus, CLMs beat encoder-decoder models by sheer amount of parameters



Alternative training objectives

- Because this type of models is more versatile, more training options exist
- BART (Lewis et al., 2020) is trained with an alternative take on autoregressive MLM
- UL2: Unifying Language Learning Paradigms by Tay et al. (2022) further merges CLM with autoregressive MLM, leading to better zero-shot performance

Table of contents



1. Pre-training / fine-tuning paradigm
2. Subword tokenization
3. Decoder language models
4. Encoder language models
5. Encoder-decoder language models
6. References

References I

Bang, Y., Cahyawijaya, S., Lee, N., Dai, W., Su, D., Wilie, B., Lovenia, H., Ji, Z., Yu, T., Chung, W., Do, Q. V., Xu, Y., and Fung, P. (2023). A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity.

Bengio, Y., Ducharme, R., and Vincent, P. (2001). A neural probabilistic language model. In Leen, T., Dietterich, T., and Tresp, V., editors, *Advances in Neural Information Processing Systems*, volume 13. MIT Press.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

References II

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., Reif, E., Du, N., Hutchinson, B., Pope, R., Bradbury, J., Austin, J., Isard, M., Gur-Ari, G., Yin, P., Duke, T., Levskaya, A., Ghemawat, S., Dev, S., Michalewski, H., Garcia, X., Misra, V., Robinson, K., Fedus, L., Zhou, D., Ippolito, D., Luan, D., Lim, H., Zoph, B., Spiridonov, A., Sepassi, R., Dohan, D., Agrawal, S., Omernick, M., Dai, A. M., Pillai, T. S., Pellat, M., Lewkowycz, A., Moreira, E., Child, R., Polozov, O., Lee, K., Zhou, Z., Wang, X., Saeta, B., Diaz, M., Firat, O., Catasta, M., Wei, J., Meier-Hellstern, K., Eck, D., Dean, J., Petrov, S., and Fiedel, N. (2022). Palm: Scaling language modeling with pathways.

Clark, J. H., Garrette, D., Turc, I., and Wieting, J. (2022). Canine: Pre-training an efficient tokenization-free encoder for language representation. *Transactions of the Association for Computational Linguistics*, 10:73–91.

Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Claude E. Shannon (1948). A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. (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.

References III

Creutz, M. and Lagus, K. (2007). Unsupervised models for morpheme segmentation and morphology learning. *ACM Trans. Speech Lang. Process.*, 4(1).

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (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.

Gage, P. (1994). A new algorithm for data compression. *C Users J.*, 12(2):23–38.

He, P., Liu, X., Gao, J., and Chen, W. (2020). Deberta: Decoding-enhanced bert with disentangled attention.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D. d. L., Hendricks, L. A., Welbl, J., Clark, A., Hennigan, T., Noland, E., Millican, K., Driessche, G. v. d., Damoc, B., Guy, A., Osindero, S., Simonyan, K., Elsen, E., Rae, J. W., Vinyals, O., and Sifre, L. (2022). Training compute-optimal large language models.

Kudo, T. (2018). Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

References iv

Kutuzov, A., Barnes, J., Velldal, E., Øvrelid, L., and Oepen, S. (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.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach.

OpenAI (2023). Gpt-4 technical report. Accessed on March 13, 2023.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. (2018). Improving language understanding by generative pre-training.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer.

Rogers, A., Kovaleva, O., and Rumshisky, A. (2020). A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

References v



Tay, Y., Dehghani, M., Tran, V. Q., Garcia, X., Wei, J., Wang, X., Chung, H. W., Shakeri, S., Bahri, D., Schuster, T., Zheng, H. S., Zhou, D., Houlsby, N., and Metzler, D. (2022). Ul2: Unifying language learning paradigms.

Tenney, I., Das, D., and Pavlick, E. (2019). BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. (2023). Llama: Open and efficient foundation language models.

Wang, A. and Cho, K. (2019). Bert has a mouth, and it must speak: Bert as a markov random field language model.

Wang, B. and Komatsuzaki, A. (2021). GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. *https://github.com/kingoflolz/mesh-transformer-jax*.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., and Fedus, W. (2022). Emergent abilities of large language models. *Transactions on Machine Learning Research*. Survey Certification.

References VI

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, ?., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., and Dean, J. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation.

Xue, L., Barua, A., Constant, N., Al-Rfou, R., Narang, S., Kale, M., Roberts, A., and Raffel, C. (2022). ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., Barua, A., and Raffel, C. (2021). mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., Mihaylov, T., Ott, M., Shleifer, S., Shuster, K., Simig, D., Koura, P. S., Sridhar, A., Wang, T., and Zettlemoyer, L. (2022). Opt: Open pre-trained transformer language models.