IN5550: Neural Methods in Natural Language Processing
Sub-lecture 7.4
Contextualized embeddings

Andrey Kutuzov

University of Oslo



Contents



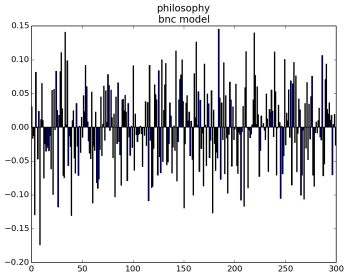
1. Brief Recap
2. Problems of static word embeddings
3. Solution: contextualized embeddings
   - We need to talk about ELMo
4. Using contextualized embeddings
   - Practicalities

Brief Recap

Word embeddings

- Distributional models are based on distributions of word co-occurrences in large training corpora;
- they represent lexical meanings as dense vectors (embeddings);
- the representations are also distributed: meaning is expressed via values of multiple vector entries;
- particular vector entries (features) are not directly related to any particular semantic 'properties', and thus not directly interpretable;
- words occurring in similar contexts have similar vectors.

Important: each word is associated with exactly one dense vector.
Hence, such models are sometimes called 'static embeddings'.



Contents



1. Brief Recap
2. Problems of static word embeddings
3. Solution: contextualized embeddings
   - We need to talk about ELMo
4. Using contextualized embeddings
   - Practicalities

Problems of static word embeddings

Meaning is meaningful only in context

English sentences with the word 'bank' in two different senses:

1. '*She was enjoying her walk down the quiet country lane towards the river bank.*' (**sense 0**)
2. '*The bank obtained its customer payment or reimbursement and released the goods to the customer.*' (**sense 1**)

Problems of static word embeddings

Meaning is meaningful only in context

English sentences with the word 'bank' in two different senses:

1. '*She was enjoying her walk down the quiet country lane towards the river bank.*' (**sense 0**)
2. '*The bank obtained its customer payment or reimbursement and released the goods to the customer.*' (**sense 1**)

Even most perfect 'static' embeddings will always yield the same vector for 'bank' . But in fact the meaning is different!

Problems of static word embeddings

Meaning is meaningful only in context

English sentences with the word 'bank' in two different senses:

1. '*She was enjoying her walk down the quiet country lane towards the river bank.*' (**sense 0**)
2. '*The bank obtained its customer payment or reimbursement and released the goods to the customer.*' (**sense 1**)

Even most perfect 'static' embeddings will always yield the same vector for 'bank' . But in fact the meaning is different!

What can be done?

Problems of static word embeddings

One can represent 'bank' as the average embedding of all the context words [Schütze, 1998].
But:

Problems of static word embeddings

One can represent 'bank' as the average embedding of all the context words [Schütze, 1998].
But:

1. Context words themselves can be ambiguous.

Problems of static word embeddings

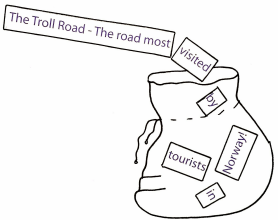
One can represent 'bank' as the average embedding of all the context words [Schütze, 1998].
But:

1. Context words themselves can be ambiguous.
2. Their contextual senses will be lost.

Problems of static word embeddings

One can represent 'bank' as the average embedding of all the context words [Schütze, 1998].
But:

1. Context words themselves can be ambiguous.
2. Their contextual senses will be lost.
3. In this 'bag of embeddings', word order information is also entirely lost.



So, what to do?

Contents



1. Brief Recap
2. Problems of static word embeddings
3. Solution: contextualized embeddings
   - We need to talk about ELMo
4. Using contextualized embeddings
   - Practicalities

Solution: contextualized embeddings

- Idea: at inference time, assign each word a vector which is a function of the whole input phrase! [Melamud et al., 2016, McCann et al., 2017]

Solution: contextualized embeddings

- Idea: at inference time, assign each word a vector which is a function of the whole input phrase! [Melamud et al., 2016, McCann et al., 2017]
- Now our word representations are context-dependent: one and the same word has different vectors in different contexts.

Solution: contextualized embeddings

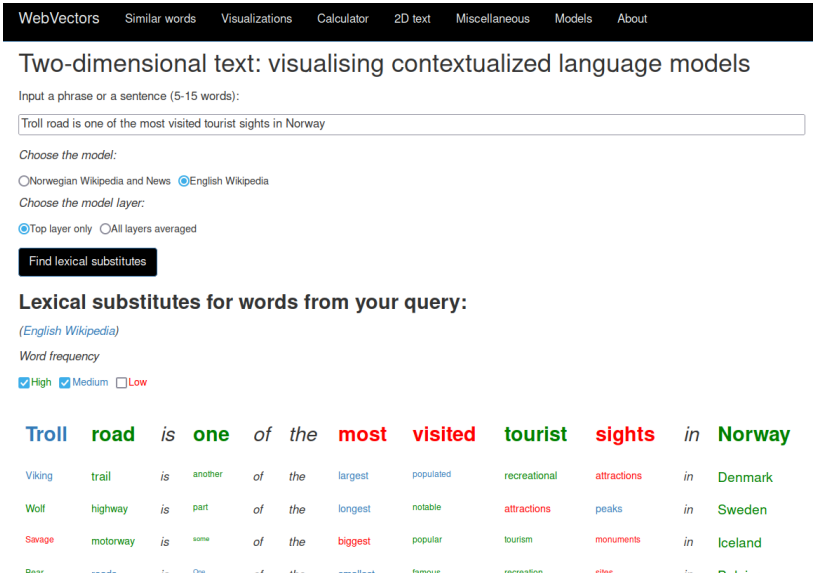
- Idea: at inference time, assign each word a vector which is a function of the whole input phrase! [Melamud et al., 2016, McCann et al., 2017]
- Now our word representations are context-dependent: one and the same word has different vectors in different contexts.
- Token embeddings VS type embeddings.

Solution: contextualized embeddings

- Idea: at inference time, assign each word a vector which is a function of the whole input phrase! [Melamud et al., 2016, McCann et al., 2017]
- Now our word representations are context-dependent: one and the same word has different vectors in different contexts.
- Token embeddings VS type embeddings.
- As an input, our model takes not an isolated word, but a phrase, sentence or text.

Solution: contextualized embeddings

http://vectors.nlpl.eu/explore/embeddings/en/contextual/



Solution: contextualized embeddings

More complex is more expressive

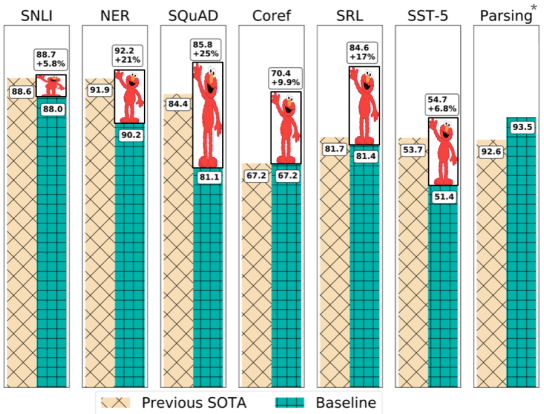
- There is no one-to-one correspondence between a word and its embedding any more.
- Word vectors are not fixed: they are learned functions of the internal states of a deep neural network.
- The architecture itself is no more a simple lookup table:
- it is a full-fledged language model.

Solution: contextualized embeddings

- 2018: ELMo (Embeddings from Language MOdels) conquered almost all NLP tasks
  [Peters et al., 2018a]

Solution: contextualized embeddings

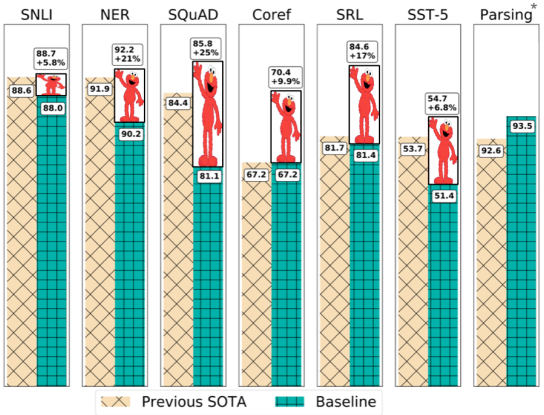
- 2018: ELMo (Embeddings from Language MOdels) conquered almost all NLP tasks
  [Peters et al., 2018a]



*Kitaev and Klein, ACL 2018 (see also Joshi et al., ACL 2018)

Solution: contextualized embeddings

- 2018: ELMo (Embeddings from Language MOdels) conquered almost all NLP tasks
  [Peters et al., 2018a]



*Kitaev and Klein, ACL 2018 (see also Joshi et al., ACL 2018)

- 2019: BERT (Bidirectional Encoder Representations from Transformer) did the same
  [Devlin et al., 2019]

Solution: contextualized embeddings

Both architectures use deep neural networks

- ELMo employs for language modeling bidirectional LSTMs.
- BERT employs transformers with self-attention (next week).
- Many other Sesame street characters made it into NLP since then!

Deep Neural Networks Are Our Friends



Wang Ling Google DeepMind

We need to talk about ELMo

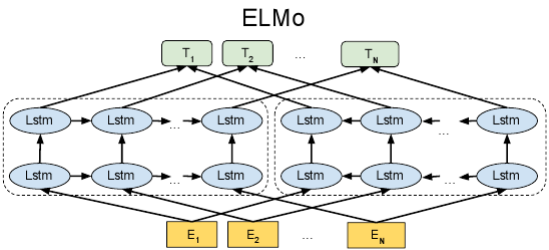
Embeddings from Language MOdels

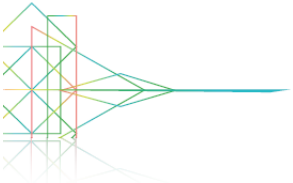
- Contextualized ELMo models are trained on raw text, optimizing for the language modeling task (next word prediction).

We need to talk about ELMo

Embeddings from Language MOdels

- Contextualized ELMo models are trained on raw text, optimizing for the language modeling task (next word prediction).
- Two BiLSTM layers over one layer of character-based CNN.
- Takes sequences of characters as an input.
  - ...actually, they are UTF-8 code units (bytes), not characters *per se*.
- LSTM output states for each word are the ELMo embeddings.





Method

ELMo represents a word t_k as a linear combination of corresponding hidden layers (inc. its embedding)

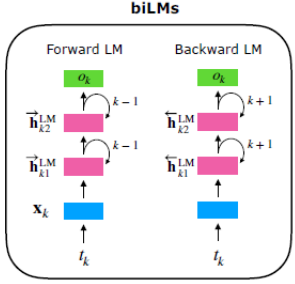
ELMo is a task specific representation. A down-stream task learns weighting parameters

$$\mathbf{ELMo}_k^{\text{task}} = \gamma^{\text{task}} \times \sum \left\{ s_2^{\text{task}} \times \mathbf{h}_{k2}^{\text{LM}},\ s_1^{\text{task}} \times \mathbf{h}_{k1}^{\text{LM}},\ s_0^{\text{task}} \times \mathbf{h}_{k0}^{\text{LM}} \right.$$

$$\left. ([\mathbf{x}_k ; \mathbf{x}_k]) \right.$$

Concatenate hidden layers

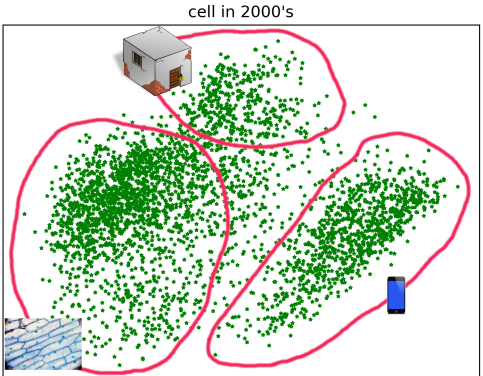
$$[\overrightarrow{\mathbf{h}}_{kj}^{\text{LM}} ; \overleftarrow{\mathbf{h}}_{kj}^{\text{LM}}]$$



Unlike usual word embeddings, ELMo is assigned to every *token* instead of a *type*

[Peters et al., 2018a]

We need to talk about ELMo



- 2-dimensional PCA projections of ELMo embeddings for each occurrence of *'cell'* in the Corpus of Historical American English (2000-2010).
- Left clusters: biological and prison senses.
- Large cluster to the right: 'mobile phone' sense.

NB: each dot represents an embedding of *'cell'* in a particular sentence.

We need to talk about ELMo

Norwegian ELMo in word sense disambiguation (PCA projection of '*rett*' embeddings):



Contents



1. Brief Recap
2. Problems of static word embeddings
3. Solution: contextualized embeddings
   - We need to talk about ELMo
4. Using contextualized embeddings
   - Practicalities

Using contextualized embeddings

How one uses contextualized embeddings?

1. As 'feature extractors': pre-trained contextualized representations are fed to the target task (e.g., document classification)

Using contextualized embeddings

How one uses contextualized embeddings?

1. As 'feature extractors': pre-trained contextualized representations are fed to the target task (e.g., document classification)
   - conceptually, the same workflow as with 'static' word embeddings

Using contextualized embeddings

How one uses contextualized embeddings?

1. As 'feature extractors': pre-trained contextualized representations are fed to the target task (e.g., document classification)
   - conceptually, the same workflow as with 'static' word embeddings
2. Fine-tuning: the whole model undergoes additional training on the target task data

Using contextualized embeddings

How one uses contextualized embeddings?

1. As 'feature extractors': pre-trained contextualized representations are fed to the target task (e.g., document classification)
   - conceptually, the same workflow as with 'static' word embeddings
2. Fine-tuning: the whole model undergoes additional training on the target task data
   - Potentially more powerful, but computationally expensive. More on that later.

Using contextualized embeddings

How one uses contextualized embeddings?

1. As 'feature extractors': pre-trained contextualized representations are fed to the target task (e.g., document classification)
   - conceptually, the same workflow as with 'static' word embeddings
2. Fine-tuning: the whole model undergoes additional training on the target task data
   - Potentially more powerful, but computationally expensive. More on that later.

Conditions			Guidelines
Pretrain	Adapt.	Task	
Any	❄	Any	Add many task parameters
Any	🔥	Any	Add minimal task parameters ⚠ Hyper-parameters
Any	Any	Seq. / clas.	❄ and 🔥 have similar performance
ELMo	Any	Sent. pair	use ❄
BERT	Any	Sent. pair	use 🔥

Table 1: This paper's guidelines for using feature extraction (❄) and fine-tuning (🔥) with ELMo and BERT. Seq.: sequence labeling. Clas.: classification. Sent. pair: sentence pair tasks.

Recommendations from [Peters et al., 2019]

References I

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.

McCann, B., Bradbury, J., Xiong, C., and Socher, R. (2017).
Learned in translation: Contextualized word vectors.
In Advances in Neural Information Processing Systems, pages 6294–6305.

Melamud, O., Goldberger, J., and Dagan, I. (2016).
context2vec: Learning generic context embedding with bidirectional LSTM.
In Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, pages 51–61, Berlin, Germany. Association for Computational Linguistics.

References II


Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018a).
Deep contextualized word representations.
In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 2227–2237. Association for Computational Linguistics.


Peters, M. E., Neumann, M., Zettlemoyer, L., and Yih, W.-t. (2018b).
Dissecting contextual word embeddings: Architecture and representation.
In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 1499–1509, Brussels, Belgium. Association for Computational Linguistics.

References III

Peters, M. E., Ruder, S., and Smith, N. A. (2019).
To tune or not to tune? adapting pretrained representations to diverse tasks.
In Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019), pages 7–14, Florence, Italy. Association for Computational Linguistics.

Schütze, H. (1998).
Automatic word sense discrimination.
Computational Linguistics, 24(1):97–123.