Lecture 10
Contextual word embeddings

LTAT.01.001 – Natural Language Processing
Kairit Sirts (kairit.sirts@ut.ee)
24.04.2020
Plan for today

● Word senses
● Contextual word embeddings
● ELMo
● Transformers
● BERT
Word senses
Word Senses

● One lemma “bank” can have many meanings:

Sense 1: • ...a bank can hold the investments in a custodial account...
Sense 2: • “...as agriculture burgeons on the east bank the river will shrink even more”

● Sense (or word sense)
  • A discrete representation of an aspect of a word’s meaning.

● The lemma bank here has two senses
Homonymy

**Homonyms**: words that share a form but have unrelated, distinct meanings:

- **bank**
  - bank\textsubscript{1}: financial institution
  - bank\textsubscript{2}: sloping land
- **bat**
  - bat\textsubscript{1}: club for hitting a ball
  - bat\textsubscript{2}: nocturnal flying mammal

1. **Homographs**

   - In English: bank/bank, bat/bat
   - In Estonian: tee/tee

2. **Homophones**:

   - In English: write and right, piece and peace
   - In Estonian: baar and paar, ball and pall
Polysemy

1. The **bank** was constructed in 1875 out of local red brick.
2. I withdrew the money from the **bank**
3. Are those the same sense?
   - Sense 2: “A financial institution”
   - Sense 1: “The building belonging to a financial institution”

A **polysemous** word has related meanings
- Most non-rare words have multiple meanings
Sense information in word embeddings

- Embedding systems like Word2Vec or Glove include one embedding per word

- All different meanings are conflated into a single vector

- This can result in semantically unrelated words being pulled together

Source: Camacho-Collados and Pilehvar, 2018.  
From word to sense embeddings: A survey on vector representations of meaning
Solving the sense problem in word embeddings

- Option I: just ignore it
- Option II: Devise methods to learn sense embeddings, i.e. one vector per sense. This is related to various problems:
  - How many senses does each word have?
  - For instance, the word “bass” has 9 senses in English WordNet
  - Wordnet: a hierarchically organized lexical database that includes concepts and their relations (synonyms, antonyms, is-a relation, is-part-of relation etc)
Noun

- **S:** (n) *bass* (the lowest part of the musical range)
- **S:** (n) *bass*, *bass part* (the lowest part in polyphonic music)
- **S:** (n) *bass*, *basso* (an adult male singer with the lowest voice)
- **S:** (n) *sea bass*, *bass* (the lean flesh of a saltwater fish of the family Serranidae)
- **S:** (n) *freshwater bass*, *bass* (any of various North American freshwater fish with lean flesh (especially of the genus Micropterus))
- **S:** (n) *bass*, *bass voice*, *basso* (the lowest adult male singing voice)
- **S:** (n) *bass* (the member with the lowest range of a family of musical instruments)
- **S:** (n) *bass* (nontechnical name for any of numerous edible marine and freshwater spiny-finned fishes)

Adjective

- **S:** (adj) *bass*, *deep* (having or denoting a low vocal or instrumental range) "a deep voice"; "a bass voice is lower than a baritone voice"; "a bass clarinet"
Solving the sense problem in word embeddings

- Option I: just ignore it
- Option II: Devise methods to learn sense embeddings, i.e. one vector per sense. This is related to various problems:
  - How many senses does each word have?
  - For instance, the word “bass” has 9 senses in English WordNet
  - Wordnet: a hierarchically organized lexical database that includes concepts and their relations (synonyms, antonyms, is-a relation, is-part-of relation etc)
  - Can group senses into larger clusters:
    - bass – music related
    - bass – fish related
- Option III: contextual word embeddings
Contextual word embeddings
Contextualized word embeddings

- Essentially large language models
- Pretrained on large amounts of unannotated data
- The representation of each word is constructed based on its context.
- The word “bank” in the following two sentences will have a different representation:
  - ...a **bank** can hold the investments in a custodial account...
  - “…as agriculture burgeons on the east **bank** the river will shrink even more”
- Thus, contextual word embeddings essentially solve the problem of different senses
Contextualized word embeddings

- Recent trend in NLP
- Pre-train deep contextualized word embeddings
ELMo
ELMo – Embeddings from Language Models

• Peters et al., 2018. Deep contextualized word representations that model:
  • Syntax and semantics of the word use
  • How the uses of the same word vary in different contexts (polysemy)

• Main characteristics
  • Contextual
  • Deep
  • Character-based
Language model for ELMo

Possible classes:
- Aardvark (0.1%)
- Improvisation (10%)
- Zyzyva (0%)

Output Layer

LSTM Layer #2

LSTM Layer #1

Embedding

Let's, stick, to
Language model for ELMo

**Forward Language Model**

- **LSTM Layer #2**
- **LSTM Layer #1**
- **Embedding**

**Backward Language Model**

- **Let's**
- **stick**
- **to**
Training the language model

- The loss is the sum for forward and backward probabilities of all words in the sentence:
  \[
  \sum_{i=1}^{n} \left( \log p(w_i|w_1, ..., w_{i-1}) + \log p(w_i|w_{i+1}, ..., w_n) \right)
  \]

- The word representation and output layer parameters are the same for both forward and backward LSTMs.
- The parameters of the forward and backward LSTMs are different.
Character-based input

- The input representations for all words are composed with CNN
- For the same word (e.g. bank) this representation is always the same, regardless of possible different meanings
- In a sense, after training this layer conforms to static word embeddings
- Most analogous to FastText
- The layer is quite large (at least in the original paper)
  - 2048 n-gram filters
  - Two highway layers
  - Then projected down to 512-dimensional vectors
What is ELMo?

- Task-specific combination of the language model layer representations
- For each token, there are $2L+1$ representations in a L-layer language model
- ELMo constructs a single token vector from all these representations
- Possible ways of doing it:
  - Take the concatenation of forward and backward representations from the last layer
  - Task-specific weighting of all language model layers
1- Concatenate hidden layers

\[ h_2 \]

\[ h_1 \]

\[ h_0 \]

2- Multiply each vector by a weight based on the task

\[ \times s_2 \]

\[ \times s_1 \]

\[ \times s_0 \]

3- Sum the (now weighted) vectors

\[ h = \gamma (s_0 h_0 + s_1 h_1 + s_2 h_2) \]

\[ \gamma, s_i - \text{task-specific parameters} \]

\[ s_i \text{ are softmax normalized} \]

ELMo embedding of “stick” for this task in this context
Transformer
Attention is all you need  
Vaswani et al., 2017

- A sequence to sequence model based on attention

- Strong results on standard machine translation benchmark datasets

- Fast because it does not include recurrent networks
Attention tricks

- **Self-attention**: each layer combines words with others

- **Multi-headed attention**: 8 attention heads learn independently

- **Normalized dot-product attention**: remove bias in dot product when using large networks

- **Positional encodings**: as the model does not use an RNN, a mechanism is needed to distinguish different input positions
Self-attention

- Each layer combines words with others with scaled dot product attention

\[
a(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V
\]
Multi-head attention

- Project keys, queries and values into several different sub-spaces
  \[ Q_i = QW_i^Q, K_i = KW_i^K, V_i = VW_i^V \]

- For each attention \( i \) compute a head
  \[ \text{head}_i = a(Q_i, K_i, V_i) \]

\[ \text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O \]
Positional encodings

- Positional encodings are added to input embeddings in both encoder and decoder
  - The embeddings and positional encoding are summed

- Positions are encoded with sine and cosine functions of different frequencies

\[
PE_{pos, 2i} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right)
\]

\[
PE_{pos, 2i+1} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right)
\]
Masking for training

- Because there is no recurrence many operations can be done in parallel
- However, we cannot “know” the future before it has arrived.
- To simulate that, the self-attention layer in the decoder is masked

```
kono  eiga  ga  kirai  I  hate  this  movie  </s>
```

Source: Attention lecture by Graham Neubig
BERT
BERT: Bidirectional Encoder Representations from Transformer

- Devlin et al., 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

- Main characteristics
  - Bidirectional
  - Very deep

- Key components
  - Masked language model
  - Auxiliary “next sentence prediction” task
Input to the BERT

<table>
<thead>
<tr>
<th>Input</th>
<th>[CLS]</th>
<th>my</th>
<th>dog</th>
<th>is</th>
<th>cute</th>
<th>[SEP]</th>
<th>he</th>
<th>likes</th>
<th>play</th>
<th>#ing</th>
<th>[SEP]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Token Embeddings</strong></td>
<td>(E_{[CLS]})</td>
<td>(E_{my})</td>
<td>(E_{dog})</td>
<td>(E_{is})</td>
<td>(E_{cute})</td>
<td>(E_{[SEP]})</td>
<td>(E_{he})</td>
<td>(E_{likes})</td>
<td>(E_{play})</td>
<td>(E_{#ing})</td>
<td>(E_{[SEP]})</td>
</tr>
<tr>
<td><strong>Segment Embeddings</strong></td>
<td>(E_A)</td>
<td>(E_A)</td>
<td>(E_A)</td>
<td>(E_A)</td>
<td>(E_A)</td>
<td>(E_A)</td>
<td>(E_B)</td>
<td>(E_B)</td>
<td>(E_B)</td>
<td>(E_B)</td>
<td>(E_B)</td>
</tr>
<tr>
<td><strong>Position Embeddings</strong></td>
<td>(E_0)</td>
<td>(E_1)</td>
<td>(E_2)</td>
<td>(E_3)</td>
<td>(E_4)</td>
<td>(E_5)</td>
<td>(E_6)</td>
<td>(E_7)</td>
<td>(E_8)</td>
<td>(E_9)</td>
<td>(E_{10})</td>
</tr>
</tbody>
</table>
Masked language model

Use the output of the masked word’s position to predict the masked word

Randomly mask 15% of tokens
Masked language model

- Randomly choose 15% of training tokens for masking
- From those:
  - Mask the token 80% of the time
  - replace the token with another random word 10% of the time
  - Leave the token unchanged 10% of the time
- This masking procedure is done because there is a mismatch between pre-training and later fine-tuning as there is no MASK token during finetuning
Next sentence prediction task

● In order to facilitate tasks that rely on two input sequences:
  ● question answering
  ● textual entailment

● Sentences are chosen into pretraining so that 50% of the time the sequence B is an actual next sentence to the sequence A

● 50% of the time sequence B is not an actual sequence to sentence A, but a random sentence from the input corpus

● The sequence-initial CLS token representation is used to train a binary classifier to discriminate between these two cases
Next sentence prediction task

Predict likelihood that sentence B belongs after sentence A

1%  IsNext
99%  NotNext

FFNN + Softmax

Tokenized Input

Input

[CLS] the man [MASK] to the store [SEP] penguin [MASK] are flightless birds [SEP]

Sentence A

Sentence B
Validation on external tasks

- During training the masked LM and next sentence classification accuracy can be monitored on an evaluation set.
- Masked LM and next sentence prediction tasks are auxiliary training objectives.
- The goal is to train models that would provide good input representations for subsequent tasks.
- It is a good idea to also monitor the model training on external tasks:
  - text classification
  - sequence tagging
- While training the BERT, take subsequent checkpoints and use as basis for training the external tasks.
- Monitor the test performance of the external tasks over the sequence of BERT checkpoints.
Task-specific fine-tuning: input

- Plug in task-specific inputs
- Two input sequences for tasks like:
  - question answering
  - textual entailment
  - paraphrase pairs etc
- One input sequence + plus empty second sequence for:
  - text classification
  - morphological/syntactic tasks
  - NER etc
Task-specific fine-tuning: output

- For classification tasks:
  - Fit a dense classification layer + softmax on top of the CLS representation

- For sequence tagging tasks:
  - Fit a dense classification layer + softmax on top of each token representation
  - Use the first piece of each token to represent the word (remember: the input was subword tokenized)
Task-specific fine-tuning

(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(b) Single Sentence Classification Tasks: SST-2, CoLA
Task-specific fine-tuning

(c) Question Answering Tasks: SQuAD v1.1

(d) Single Sentence Tagging Tasks: CoNLL-2003 NER
Available BERT models

- Huggingface transformers library offers good access to available BERT models: https://huggingface.co/transformers/
- Cased and uncased models
- Smaller and larger models:

<table>
<thead>
<tr>
<th>BERT base</th>
<th>BERT large</th>
</tr>
</thead>
<tbody>
<tr>
<td>12 layers</td>
<td>24 layers</td>
</tr>
<tr>
<td>hidden size 768</td>
<td>hidden size 1024</td>
</tr>
<tr>
<td>12 heads</td>
<td>16 heads</td>
</tr>
<tr>
<td>110M parameters</td>
<td>340M parameters</td>
</tr>
</tbody>
</table>

- First models masked out sub-word tokens, now available models that mask out whole words
Non-English BERT models

- BERT models are available for several languages, including:
  - Chinese
  - German
  - Japanese
  - Finnish
  - Dutch
- These models are also available via huggingface transformers library
- We are currently training a model also for Estonian
Multilingual BERT models

Several multilingual BERT models are available via huggingface transformers library

- Bert base multilingual cased:
  - trained on 104 largest Wikipedia languages
  - 110 shared WordPiece vocabulary
  - includes languages like Estonian, Russian, Ukraine

- There are also other BERT-like models:
  - XLM masked LM trained on 100 languages
  - DistilmBERT distilled from multilingual BERT model
  - XLM RoBERTa large trained on 100 languages
In conclusion

- Contextual word embeddings seem to be a very promising general component for many NLP tasks.
- They help to improve the models especially in low resource setting.
- Major results in NLP during the last decade:
  - static word embeddings
  - recurrent neural networks
  - encoder-decoder networks
  - attention mechanism
  - Transformer architecture
  - Pre-trained contextual word embedding models
- Opens an area for lots of research and experiments for many languages and many tasks.