L15: Contextualized Representations and Pre-training

Spring 2024
Announcements

• Project proposal feedback on Gradescope by April 12
• Project poster session scheduled on **May 3rd 1:30-3:30pm @Friend Center upper atrium**
• Project Compute: We can reimburse each team one month of Colab Pro for your computing needs or up to $50 of OpenAl/Claude credits (see Ed post!)
• A4 is slightly more challenging - get started early!
• April 12 and April 19: Guest lectures!
This lecture

- Contextualized word embeddings
- Pre-training and fine-tuning
- GPT, ELMo, BERT
• ELMo = Embeddings from Language Models
• GPT = Generative Pre-Training
• BERT = Bidirectional Encoder Representations from Transformers

(ERNIE, Grover, Big Bird, Kermit, RoBERTa, Rosita, …)
Contextualized Word Embeddings
Limitations of word2vec

- One vector for each word type  
  (Aka. “Static word embeddings”)

- Complex characteristics of word use: syntax and semantics

- Polysemous words, e.g., bank, mouse

  \[ v(\text{play}) = \begin{pmatrix} -0.224 \\ 0.130 \\ -0.290 \\ 0.276 \end{pmatrix} \]

  - \( \text{mouse}^1 \): .... a mouse controlling a computer system in 1968.
  - \( \text{mouse}^2 \): .... a quiet animal like a mouse
  - \( \text{bank}^1 \): ...a bank can hold the investments in a custodial account ...
  - \( \text{bank}^2 \): ...as agriculture burgeons on the east bank, the river ...
Contextualized word embeddings

Let’s build a vector for each word conditioned on its **context**!

\[ f : (w_1, w_2, \ldots, w_n) \rightarrow x_1, \ldots, x_n \in \mathbb{R}^d \]
Contextualized word embeddings

Let’s build a vector for each word conditioned on its context!

Hey ELMo, what’s the embedding of the word “stick”?

There are multiple possible embeddings! Use it in a sentence.

Oh, okay. Here: “Let’s stick to improvisation in this skit”

Oh in that case, the embedding is: -0.02, -0.16, 0.12, -0.1 … etc

http://jalammar.github.io/illustrated-bert/
Contextualized word embeddings

Sent #1: Chico Ruiz made a spectacular play on Alusik’s grounder {. . . }

Sent #2: Olivia De Havilland signed to do a Broadway play for Garson {. . . }

Sent #3: Kieffer was commended for his ability to hit in the clutch, as well as his all-round excellent play {. . . }

Sent #4: {. . . } they were actors who had been handed fat roles in a successful play {. . . }

Sent #5: Concepts play an important role in all aspects of cognition {. . . }

$\nu(\text{play}) = ?$
Contextualized word embeddings

Sent #1: Chico Ruiz made a spectacular play on Alusik’s grounder { . . . }

Which of the following ν(play) is expected to have the most similar vector to the first one?

(A) Olivia De Havilland signed to do a Broadway play for Garson { . . . }
(B) Kieffer was commended for his ability to hit in the clutch, as well as his all-round excellent play { . . . }
(C) { . . . } they were actors who had been handed fat roles in a successful play { . . . }
(D) Concepts play an important role in all aspects of cognition { . . . }

(B) is correct.
## Contextualized word embeddings

<table>
<thead>
<tr>
<th>Source</th>
<th>Nearest Neighbors</th>
</tr>
</thead>
<tbody>
<tr>
<td>GloVe</td>
<td>play</td>
</tr>
<tr>
<td></td>
<td>playing, game, games, played, players, plays, player, Play, football, multiplayer</td>
</tr>
<tr>
<td>biLM</td>
<td>Chico Ruiz made a spectacular play on Alusik’s grounder {...}</td>
</tr>
<tr>
<td></td>
<td>Kieffer, the only junior in the group, was commended for his ability to hit in the clutch, as well as his all-round excellent play.</td>
</tr>
<tr>
<td></td>
<td>Olivia De Havilland signed to do a Broadway play for Garson {...}</td>
</tr>
<tr>
<td></td>
<td>{...} they were actors who had been handed fat roles in a successful play, and had talent enough to fill the roles competently, with nice understatement.</td>
</tr>
</tbody>
</table>
The key idea of ELMo:

- Train two stacked LSTM-based language models on a large corpus
- Use the hidden states of the LSTMs for each token to compute a vector representation of each word
How does ELMo work?

Contextualized word embeddings = The weighted average of input embeddings + all hidden representations

\[
\text{ELMo}^{\text{task}}_k = E(R_k; \Theta^{\text{task}}) = \gamma^{\text{task}} \sum_{j=0}^{L} s_j^{\text{task}} h_{k,j}^{LM}
\]

The weights \( \gamma^{\text{task}} \), \( s_j^{\text{task}} \) are task-dependent and learned

http://jalammar.github.io/illustrated-bert/
How does ELMo work?

1- Concatenate hidden layers

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

3- Sum the (now weighted) vectors

ELMo embedding of “stick” for this task in this context
ELMo: pre-training and the use

- Data: 10 epochs on 1B Word Benchmark (trained on single sentences)
- Training time: 2 weeks on 3 NVIDIA GTX 1080 GPUs

Example use: A BiLSTM model for sentiment classification

\[ e(\text{terribly}); \text{ELMo}^{\text{task}} \]

(Peters et al, 2018): Deep contextualized word representations
ELMo: some take-aways

Q: Why use both forward and backward language models?
Because it is important to model both left and right context!
Bidirectionality is very important in language understanding tasks!

Q: Why use the weighted average of different layers instead of just the top layer?
Because different layers are expected to encode different information.
Pre-training and Fine-tuning
What is pre-training / fine-tuning?

• “Pre-train” a model on a large dataset for task X, then “fine-tune” it on a dataset for task Y

• Key idea: X is somewhat related to Y, so a model that can do X will have some good neural representations for Y as well

• ImageNet pre-training is huge in computer vision: learning generic visual features for recognizing objects

Can we find some task X that can be useful for a wide range of downstream tasks Y?
Feature-based vs fine-tuning approaches

- ELMo is a feature-based approach which only produces word embeddings that can be used as input representations of existing neural models.
Feature-based vs fine-tuning approaches

- GPT / BERT (and most of following models) are **fine-tuning approaches**
  - Almost all model weights will be **re-used**, and only a small number of task-specific will be added for downstream tasks
Generative Pre-Training (GPT)

- Use a **Transformer decoder** (unidirectional; left-to-right) instead of LSTMs
- Use **language modeling** as a pre-training objective
- Trained on longer segments of text (**512 BPE tokens**), not just single sentences

(Radford et al, 2018): Improving Language Understanding by Generative Pre-Training
Generative Pre-Training (GPT) (Released in 2018/6)

- “Fine-tune” the entire set of model parameters on various downstream tasks

(Radford et al, 2018): Improving Language Understanding by Generative Pre-Training
BERT: Bidirectional Encoder Representations from Transformers

- It is a fine-tuning approach based on a deep **bidirectional Transformer encoder** instead of a Transformer decoder
- The key: learn representations based on **bidirectional contexts**

Example #1: we went to the river **bank**.
Example #2: I need to go to **bank** to make a deposit.

- Two new pre-training objectives:
  - **Masked language modeling (MLM)**
  - Next sentence prediction (NSP) - Later work shows that NSP hurts performance though..

(Devlin et al, 2019): BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
Masked Language Modeling (MLM)

- Q: Why can’t we do language modeling with bidirectional models?

- Solution: Mask out k% of the input words, and then predict the masked words

  The man went to [MASK] to buy a [MASK] of milk

  $k = 15\%$ in practice
Masked Language Modeling (MLM)

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

Possible classes:
All English words

- Aardvark: 0.1%
- Improvisation: 10%
- Zzyzyva: 0%

Randomly mask 15% of tokens

Input

[CLS] Let's stick to [MASK] in this skit

[CLS] Let's stick to improvisation in this skit

http://jalammar.github.io/illustrated-bert/
MLM: 80-10-10 corruption

For the 15% predicted words,

- 80% of the time, they replace it with [MASK] token
  
  went to the store → went to the [MASK]

- 10% of the time, they replace it with a random word in the vocabulary
  
  went to the store → went to the running

- 10% of the time, they keep it unchanged
  
  went to the store → went to the store

Why? Because [MASK] tokens are never seen during fine-tuning
(See Table 8 of the paper for an ablation study)
Next Sentence Prediction (NSP)

- Motivation: many NLP downstream tasks require understanding the relationship between two sentences (natural language inference, paraphrase detection, QA)
- NSP is designed to reduce the gap between pre-training and fine-tuning

```
[CLS]: a special token always at the beginning
[SEP]: a special token used to separate two segments

Input = [CLS] the man went to [MASK] store [SEP]
       he bought a gallon [MASK] milk [SEP]
Label = IsNext

Input = [CLS] the man [MASK] to the store [SEP]
       penguin [MASK] are flight # less birds [SEP]
Label = NotNext
```

They sample two contiguous segments for 50% of the time and another random segment from the corpus for 50% of the time.
BERT pre-training

- Vocabulary size: 30,000 wordpieces (common sub-word units) (Wu et al., 2016)

- Input embeddings:

![Image: Stanford CS224N]
BERT pre-training

- BERT-base: 12 layers, 768 hidden size, 12 attention heads, 110M parameters
  - Same as OpenAI GPT

- BERT-large: 24 layers, 1024 hidden size, 16 attention heads, 340M parameters
  - OpenAI GPT was trained on BooksCorpus only!

- Training corpus: Wikipedia (2.5B) + BooksCorpus (0.8B)
- Max sequence size: 512 wordpiece tokens (roughly 256 and 256 for two non-contiguous sequences)
- Trained for 1M steps, batch size 128k
BERT pre-training

- MLM and NSP are trained together
- [CLS] is pre-trained for NSP
- Other token representations are trained for MLM
BERT fine-tuning

“Pretrain once, finetune many times.”

sentence-level tasks

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

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

“Pretrain once, finetune many times.”

token-level tasks

(c) Question Answering Tasks:
SQuAD v1.1

(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER
Example: sentiment classification

We just need to introduce $C \times h$ parameters for classification tasks ($C =$ # of classes, $h =$ hidden size)!

$$P(y = k) = \text{softmax}_k(W_o h_{[CLS]})$$

$W_o \in \mathbb{R}^{C \times h}$

All the parameters will be learned together (original BERT parameters + new classifier parameters)
Example: named entity recognition (NER)

We just need to introduce $C \times h$ parameters for classification tasks ($C =$ # of classes, $h =$ hidden size)!

$$P(y_i = k) = \text{softmax}_k(W_o h_i)$$

$W_o \in \mathbb{R}^{C \times h}$
## Experimental results: GLUE

<table>
<thead>
<tr>
<th>System</th>
<th>MNLI-(m/mm)</th>
<th>QQP</th>
<th>QNLI</th>
<th>SST-2</th>
<th>CoLA</th>
<th>STS-B</th>
<th>MRPC</th>
<th>RTE</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-OpenAI SOTA</td>
<td>80.6/80.1</td>
<td>66.1</td>
<td>82.3</td>
<td>93.2</td>
<td>35.0</td>
<td>81.0</td>
<td>86.0</td>
<td>61.7</td>
<td>74.0</td>
</tr>
<tr>
<td>BiLSTM+ELMo+Attn</td>
<td>76.4/76.1</td>
<td>64.8</td>
<td>79.8</td>
<td>90.4</td>
<td>36.0</td>
<td>73.3</td>
<td>84.9</td>
<td>56.8</td>
<td>71.0</td>
</tr>
<tr>
<td>OpenAI GPT</td>
<td>82.1/81.4</td>
<td>70.3</td>
<td>87.4</td>
<td>91.3</td>
<td>45.4</td>
<td>80.0</td>
<td>82.3</td>
<td>56.0</td>
<td>75.1</td>
</tr>
<tr>
<td>BERT\text{BASE}</td>
<td>84.6/83.4</td>
<td>71.2</td>
<td>90.5</td>
<td>93.5</td>
<td>52.1</td>
<td>85.8</td>
<td>88.9</td>
<td>66.4</td>
<td>79.6</td>
</tr>
<tr>
<td>BERT\text{LARGE}</td>
<td>86.7/85.9</td>
<td>72.1</td>
<td>92.7</td>
<td>94.9</td>
<td>60.5</td>
<td>86.5</td>
<td>89.3</td>
<td>70.1</td>
<td>82.1</td>
</tr>
</tbody>
</table>
## Experimental results: SQuAD

<table>
<thead>
<tr>
<th>System</th>
<th>Dev EM</th>
<th>Dev F1</th>
<th>Test EM</th>
<th>Test F1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Top Leaderboard Systems</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Human</td>
<td>-</td>
<td>-</td>
<td>82.3</td>
<td>91.2</td>
</tr>
<tr>
<td>#1 Ensemble - nlnet</td>
<td>-</td>
<td>-</td>
<td>86.0</td>
<td>91.7</td>
</tr>
<tr>
<td>#2 Ensemble - QANet</td>
<td>-</td>
<td>-</td>
<td>84.5</td>
<td>90.5</td>
</tr>
<tr>
<td><strong>Published</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BiDAF+ELMo (Single)</td>
<td>-</td>
<td>85.6</td>
<td>-</td>
<td>85.8</td>
</tr>
<tr>
<td>R.M. Reader (Ensemble)</td>
<td>81.2</td>
<td>87.9</td>
<td>82.3</td>
<td>88.5</td>
</tr>
<tr>
<td><strong>Ours</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BERT_BASE (Single)</td>
<td>80.8</td>
<td>88.5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BERT_LARGE (Single)</td>
<td>84.1</td>
<td>90.9</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BERT_LARGE (Ensemble)</td>
<td>85.8</td>
<td>91.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BERT_LARGE (Sgl.+TriviaQA)</td>
<td><strong>84.2</strong></td>
<td><strong>91.1</strong></td>
<td><strong>85.1</strong></td>
<td><strong>91.8</strong></td>
</tr>
<tr>
<td>BERT_LARGE (Ens.+TriviaQA)</td>
<td><strong>86.2</strong></td>
<td><strong>92.2</strong></td>
<td><strong>87.4</strong></td>
<td><strong>93.2</strong></td>
</tr>
</tbody>
</table>

SQuAD = Stanford Question Answering dataset
Ablation study: pre-training tasks

- MLM >> left-to-right LMs
- NSP improves on some tasks
- Note: later work (Joshi et al., 2020; Liu et al., 2019) argued that NSP is not useful
Ablation study: model sizes

The bigger, the better!

<table>
<thead>
<tr>
<th>Hyperparams</th>
<th>Dev Set Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>#L</td>
<td>#H</td>
</tr>
<tr>
<td>3</td>
<td>768</td>
</tr>
<tr>
<td>6</td>
<td>768</td>
</tr>
<tr>
<td>6</td>
<td>768</td>
</tr>
<tr>
<td>12</td>
<td>768</td>
</tr>
<tr>
<td>12</td>
<td>1024</td>
</tr>
<tr>
<td>24</td>
<td>1024</td>
</tr>
</tbody>
</table>
Ablation study: training efficiency

MLM takes slightly longer to converge because it only predicts 15% of tokens.
What does BERT learn?

(Clark et al., 2019) What Does BERT Look At? An Analysis of BERT’s Attention
ELMo vs GPT vs BERT

Which of the following statements is INCORRECT?

(A) BERT was trained on more data than ELMo

(B) BERT builds on Transformer encoder, and GPT builds on Transformer decoder

(C) ELMo requires different model architectures for different tasks

(D) BERT was trained on data with longer contexts compared to GPT

(D) is correct.