CS388: Natural Language Processing

Lecture 18: Pre-training 1: BERT

Recall: Self-Attention
- Each word forms a “query” which then computes attention over each word
  \[ \alpha_{i,j} = \text{softmax}(x_i^T x_j) \quad \text{scalar} \]
  \[ x_i' = \sum_{j=1}^{n} \alpha_{i,j} x_j \quad \text{vector = sum of scalar \times vector} \]

Recall: Multi-Head Self Attention
- Alammar, *The Illustrated Transformer*
  \[ \text{sent len x sent len (attn for each word to each other)} \]
  \[ \text{softmax} \left( \frac{x_i \times W^Q \times K^T}{\sqrt{d_k}} \right) \]
  \[ = \frac{x_i \times W^Q}{\sqrt{d_k}} \times \begin{bmatrix} V \end{bmatrix} \]
  \[ Z \text{ is a weighted combination of V rows} \]

Greg Durrett

Project 2 due today (last assignment to use slip days on)
Presentation day announcements next week
Recall: Transformers

- Augment word embedding with position embeddings, each dim is a sine/cosine wave of a different frequency. Closer points = higher dot products.
- Works essentially as well as just encoding position as a one-hot vector.

Vaswani et al. (2017)

This Lecture

- BERT
- BERT Results, Extensions
- Analysis/Visualization of BERT
- GPT/GPT2

BERT

- AI2 released ELMo in spring 2018, GPT was released in summer 2018, BERT came out October 2018
- Three major changes compared to ELMo:
  - Transformers instead of LSTMs (transformers in GPT as well)
  - Bidirectional <-> Masked LM objective instead of standard LM
  - Fine-tune instead of freeze at test time
BERT

- ELMo is a unidirectional model (as is GPT): we can concatenate two unidirectional models, but is this the right thing to do?
- ELMo reps look at each direction in isolation; BERT looks at them jointly

"performer"  ELMo

"ballet dancer"  ELMo

A stunning ballet dancer, Copeland is one of the best performers to see live.

"ballet dancer/performer"  BERT

Devlin et al. (2019)

Masked Language Modeling

- How to prevent cheating? Next word prediction fundamentally doesn’t work for bidirectional models, instead do masked language modeling

- BERT formula: take a chunk of text, predict 15% of the tokens
- For 80% (of the 15%), replace the input token with [MASK]
- For 10%, replace w/random
- For 10%, keep same (why?)

Madagascar

Devlin et al. (2019)

Next “Sentence” Prediction

- Input: [CLS] Text chunk 1 [SEP] Text chunk 2
- 50% of the time, take the true next chunk of text, 50% of the time take a random other chunk. Predict whether the next chunk is the “true” next
- BERT objective: masked LM + next sentence prediction

NotNext

Madagascar

enjoyed

like

Transformer

...  

Transformer


Devlin et al. (2019)
**BERT Architecture**

- BERT Base: 12 layers, 768-dim, 12 heads. Total params = 110M
- BERT Large: 24 layers, 1024-dim, 16 heads. Total params = 340M
- Positional embeddings and segment embeddings, 30k word pieces
- This is the model that gets **pre-trained** on a large corpus

**What can BERT do?**

- CLS token is used to provide classification decisions
- Sentence pair tasks (entailment): feed both sentences into BERT
- BERT can also do tagging by predicting tags at each word piece

**What can BERT NOT do?**

- **BERT cannot** generate text (at least not in an obvious way)
- Not an autoregressive model, can do weird things like stick a [MASK] at the end of a string, fill in the mask, and repeat
- Masked language models are intended to be used primarily for “analysis” tasks

**How does BERT model this sentence pair stuff?**

- Transformers can capture interactions between the two sentences, even though the NSP objective doesn’t really cause this to happen
**Fine-tuning BERT**

- Fine-tune for 1-3 epochs, batch size 2-32, learning rate 2e-5 - 5e-5
- Large changes to weights up here (particularly in last layer to route the right information to [CLS])
- Smaller changes to weights lower down in the transformer
- Small LR and short fine-tuning schedule mean weights don't change much
- More complex “triangular learning rate” schemes exist

---

**Evaluation: GLUE**

<table>
<thead>
<tr>
<th>Corpus</th>
<th>[Train]</th>
<th>[Test]</th>
<th>Task</th>
<th>Metrics</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoLA</td>
<td>8.5k</td>
<td>1k</td>
<td>acceptability</td>
<td>Matthews corr.</td>
<td>misc.</td>
</tr>
<tr>
<td>SST-2</td>
<td>67k</td>
<td>1.3k</td>
<td>sentiment</td>
<td>acc.</td>
<td>movie reviews</td>
</tr>
<tr>
<td>MRPC</td>
<td>3.7k</td>
<td>1.7k</td>
<td>paraphrase</td>
<td>acc./F1</td>
<td>news</td>
</tr>
<tr>
<td>STS-B</td>
<td>7k</td>
<td>1.4k</td>
<td>sentence similarity</td>
<td>Pearson/Spearman corr.</td>
<td>misc.</td>
</tr>
<tr>
<td>QQP</td>
<td>364k</td>
<td>391k</td>
<td>paraphrase</td>
<td>acc./F1</td>
<td>social QA questions</td>
</tr>
<tr>
<td>MNLI</td>
<td>393k</td>
<td>20k</td>
<td>NLI</td>
<td>matched acc./mismatched acc.</td>
<td>misc.</td>
</tr>
<tr>
<td>QNLI</td>
<td>105k</td>
<td>5.4k</td>
<td>QA/NLI</td>
<td>acc.</td>
<td>Wikipedia</td>
</tr>
<tr>
<td>RTE</td>
<td>2.5k</td>
<td>3k</td>
<td>NLI</td>
<td>acc.</td>
<td>news, Wikipedia</td>
</tr>
<tr>
<td>WNLI</td>
<td>654</td>
<td>146</td>
<td>coreference/NLI</td>
<td>acc.</td>
<td>fiction books</td>
</tr>
</tbody>
</table>

---

**BERT Results, Extensions**

- **Fine-tuning BERT**
  - Fine-tune for 1-3 epochs, batch size 2-32, learning rate 2e-5 - 5e-5
  - Large changes to weights up here (particularly in last layer to route the right information to [CLS])
  - Smaller changes to weights lower down in the transformer
  - Small LR and short fine-tuning schedule mean weights don't change much
  - More complex “triangular learning rate” schemes exist

- **Evaluation: GLUE**

---

**Fine-tuning BERT**

- Fine-tune for 1-3 epochs, batch size 2-32, learning rate 2e-5 - 5e-5
- Large changes to weights up here (particularly in last layer to route the right information to [CLS])
- Smaller changes to weights lower down in the transformer
- Small LR and short fine-tuning schedule mean weights don’t change much
- More complex “triangular learning rate” schemes exist

---

**Evaluation: GLUE**

<table>
<thead>
<tr>
<th>Corpus</th>
<th>[Train]</th>
<th>[Test]</th>
<th>Task</th>
<th>Metrics</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoLA</td>
<td>8.5k</td>
<td>1k</td>
<td>acceptability</td>
<td>Matthews corr.</td>
<td>misc.</td>
</tr>
<tr>
<td>SST-2</td>
<td>67k</td>
<td>1.3k</td>
<td>sentiment</td>
<td>acc.</td>
<td>movie reviews</td>
</tr>
<tr>
<td>MRPC</td>
<td>3.7k</td>
<td>1.7k</td>
<td>paraphrase</td>
<td>acc./F1</td>
<td>news</td>
</tr>
<tr>
<td>STS-B</td>
<td>7k</td>
<td>1.4k</td>
<td>sentence similarity</td>
<td>Pearson/Spearman corr.</td>
<td>misc.</td>
</tr>
<tr>
<td>QQP</td>
<td>364k</td>
<td>391k</td>
<td>paraphrase</td>
<td>acc./F1</td>
<td>social QA questions</td>
</tr>
<tr>
<td>MNLI</td>
<td>393k</td>
<td>20k</td>
<td>NLI</td>
<td>matched acc./mismatched acc.</td>
<td>misc.</td>
</tr>
<tr>
<td>QNLI</td>
<td>105k</td>
<td>5.4k</td>
<td>QA/NLI</td>
<td>acc.</td>
<td>Wikipedia</td>
</tr>
<tr>
<td>RTE</td>
<td>2.5k</td>
<td>3k</td>
<td>NLI</td>
<td>acc.</td>
<td>news, Wikipedia</td>
</tr>
<tr>
<td>WNLI</td>
<td>654</td>
<td>146</td>
<td>coreference/NLI</td>
<td>acc.</td>
<td>fiction books</td>
</tr>
</tbody>
</table>

---

**Fine-tuning BERT**

- Fine-tune for 1-3 epochs, batch size 2-32, learning rate 2e-5 - 5e-5
- Large changes to weights up here (particularly in last layer to route the right information to [CLS])
- Smaller changes to weights lower down in the transformer
- Small LR and short fine-tuning schedule mean weights don’t change much
- More complex “triangular learning rate” schemes exist

---

**Evaluation: GLUE**

<table>
<thead>
<tr>
<th>Corpus</th>
<th>[Train]</th>
<th>[Test]</th>
<th>Task</th>
<th>Metrics</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoLA</td>
<td>8.5k</td>
<td>1k</td>
<td>acceptability</td>
<td>Matthews corr.</td>
<td>misc.</td>
</tr>
<tr>
<td>SST-2</td>
<td>67k</td>
<td>1.3k</td>
<td>sentiment</td>
<td>acc.</td>
<td>movie reviews</td>
</tr>
<tr>
<td>MRPC</td>
<td>3.7k</td>
<td>1.7k</td>
<td>paraphrase</td>
<td>acc./F1</td>
<td>news</td>
</tr>
<tr>
<td>STS-B</td>
<td>7k</td>
<td>1.4k</td>
<td>sentence similarity</td>
<td>Pearson/Spearman corr.</td>
<td>misc.</td>
</tr>
<tr>
<td>QQP</td>
<td>364k</td>
<td>391k</td>
<td>paraphrase</td>
<td>acc./F1</td>
<td>social QA questions</td>
</tr>
<tr>
<td>MNLI</td>
<td>393k</td>
<td>20k</td>
<td>NLI</td>
<td>matched acc./mismatched acc.</td>
<td>misc.</td>
</tr>
<tr>
<td>QNLI</td>
<td>105k</td>
<td>5.4k</td>
<td>QA/NLI</td>
<td>acc.</td>
<td>Wikipedia</td>
</tr>
<tr>
<td>RTE</td>
<td>2.5k</td>
<td>3k</td>
<td>NLI</td>
<td>acc.</td>
<td>news, Wikipedia</td>
</tr>
<tr>
<td>WNLI</td>
<td>654</td>
<td>146</td>
<td>coreference/NLI</td>
<td>acc.</td>
<td>fiction books</td>
</tr>
</tbody>
</table>
Results

<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>79.9</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>88.1</td>
<td>91.3</td>
<td>45.4</td>
<td>80.0</td>
<td>82.3</td>
<td>56.0</td>
<td>75.2</td>
</tr>
<tr>
<td>BERT BASE</td>
<td>84.6/83.4</td>
<td>71.2</td>
<td>90.1</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 LARGE</td>
<td>86.7/85.9</td>
<td>72.1</td>
<td>91.1</td>
<td>94.9</td>
<td>60.5</td>
<td>86.5</td>
<td>89.3</td>
<td>70.1</td>
<td>81.9</td>
</tr>
</tbody>
</table>

- Huge improvements over prior work (even compared to ELMo)
- Effective at “sentence pair” tasks: textual entailment (does sentence A imply sentence B), paraphrase detection
  
Devlin et al. (2018)

Subsequent Improvements to BERT

- Dynamic masking: standard BERT uses the same MASK scheme for every epoch, RoBERTa recomputes them
  
  epoch 2
  epoch 1
  
  ... John visited Madagascar yesterday ...

- Whole word masking: don’t mask out parts of words
  
  ... _John _visited _Madagascar gas car yesterday ...
  
Liu et al. (2019)

RoBERTa

- “Robustly optimized BERT” incorporating some of these tricks
- 160GB of data instead of 16 GB
- New training + more data = better performance

Liu et al. (2019)

ALBERT

- Factorized embedding matrix to save parameters, model context-independent words with fewer parameters
  
  Ordinarily $|V| \times H$ — $|V|$ is 30k-90k, $H$ is >1000

  Factor into two matrices with a low-rank approximation
  
  Now: $|V| \times E$ and $E \times H$ — $E$ is 128 in their implementation

  Additional cross-layer parameter sharing

Lan et al. (2020)
### ELECTRA

- No need to necessarily have a generative model (predicting words)
- This objective is more computationally efficient (trains faster) than the standard BERT objective

---

### BERT/MLMs

- There are lots of ways to train these models!
- Key factors:
  - Big enough model
  - Big enough data
  - Well-designed “self-supervised” objective (something like language modeling). Needs to be a hard enough problem!

---

### Analysis/Visualization of BERT

**BERTology**

1. How can we probe syntactic + semantic knowledge of BERT? What does BERT “know” in its representations?

2. What can we learn from looking at attention heads?

3. What can we learn about training BERT (more efficiently, etc.)?

---

*Clark et al. (2020)*

*Rogers et al. (2020)*
**BERTology: Probing**

(1) In general: set up some “probing” task to try to determine syntactic features from BERT’s hidden states

E.g.: Words with syntactic relations have a higher impact on one another during MLM prediction

---

**BERTology**

(2) What’s going inside attention heads?

---

**What does BERT learn?**

- Heads on transformers learn interesting and diverse things: content heads (attend based on content), positional heads (based on position), etc.

---

**What does BERT learn?**

- Still way worse than what supervised systems can do, but interesting that this is learned organically
Compressing BERT

- Remove 60+% of BERT’s heads post-training with minimal drop in performance
- DistilBERT (Sanh et al., 2019): nearly as good with half the parameters of BERT (via knowledge distillation)

GPT/GPT2

- “ELMo with transformers” (works better than ELMo)
- Train a single unidirectional transformer LM on long contexts
- GPT2: trained on 40GB of text collected from upvoted links from reddit
- 1.5B parameters — by far the largest of these models trained as of March 2019
- Because it’s a language model, we can generate from it

OpenAI GPT/GPT2

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Layers</th>
<th>$d_{model}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>117M</td>
<td>12</td>
<td>768</td>
</tr>
<tr>
<td>345M</td>
<td>24</td>
<td>1024</td>
</tr>
<tr>
<td>762M</td>
<td>36</td>
<td>1280</td>
</tr>
<tr>
<td>1542M</td>
<td>48</td>
<td>1600</td>
</tr>
</tbody>
</table>

Michel et al. (2019)

OpenAI GPT2

- Slide credit: OpenAI

Radford et al. (2019)
Open Questions

1) How novel is the stuff being generated? (Is it just doing nearest neighbors on a large corpus?)
2) How do we understand and distill what is learned in this model?
3) How do we harness these priors for conditional generation tasks (summarization, generate a report of a basketball game, etc.)
4) Is this technology dangerous?

Grover

» Sample from a large language model conditioned on a domain, date, authors, and headline

» Humans rank Grover-generated propaganda as more realistic than real “fake news”

» Fine-tuned Grover can detect Grover propaganda easily — authors argue for releasing it for this reason

» NOTE: Not a GAN, discriminator trained separately from the generator

Takeaways

» BERT-based systems are state-of-the-art for nearly every major text analysis task

» Transformers + lots of data + self-supervision seems to do very well

» Lots of work studying and analyzing these, but few “deep” conclusions have emerged

» Next time: modifications of these (BART/T5, GPT-3, etc.)