CS388: Natural Language Processing

Lecture 18: Pre-training 1: BERT

Greg Durrett

Credit: ???
Project 2 due today (last assignment to use slip days on)

Presentation day announcements next week
Recall: Self-Attention

- Each word forms a “query” which then computes attention over each word

\[ \alpha_{i,j} = \text{softmax}(x_i^\top x_j) \quad \text{scalar} \]

\[ x'_i = \sum_{j=1}^{n} \alpha_{i,j} x_j \quad \text{vector} = \text{sum of scalar} * \text{vector} \]

Vaswani et al. (2017)
Recall: Multi-Head Self Attention

Alammar, *The Illustrated Transformer*

sent len x sent len (attn for each word to each other)

\[ Z = \text{softmax} \left( \frac{Q K^T}{\sqrt{d_k}} \right) \]

sent len x hidden dim

Z is a weighted combination of V rows
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
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
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.

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

Devlin et al. (2019)
How to learn a “deeply bidirectional” model? What happens if we just replace an LSTM with a transformer?

ELMo (Language Modeling)

- visited, Madag, yesterday, ...
- John, visited Madagascar yesterday

BERT

- visited, Madag, yesterday, ...
- John, visited Madagascar yesterday

Transformer LMs have to be “one-sided” (only attend to previous tokens), not what we want.
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?)

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 chunk.
- BERT objective: masked LM + next sentence prediction

Devlin et al. (2019)
- 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

Devlin et al. (2019)
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

Devlin et al. (2019)
What can BERT do?

Entails

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.

[CLS] A boy plays in the snow [SEP] A boy is outside
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
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
## Fine-tuning BERT

<table>
<thead>
<tr>
<th>Pretraining</th>
<th>Adaptation</th>
<th>NER CoNLL 2003</th>
<th>SA SST-2</th>
<th>Nat. lang. inference MNLI</th>
<th>Semantic textual similarity SICK-E</th>
<th>Semantic textual similarity SICK-R</th>
<th>MRPC</th>
<th>STS-B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Skip-thoughts</td>
<td>❄️</td>
<td>-</td>
<td>81.8</td>
<td>62.9</td>
<td>-</td>
<td>86.6</td>
<td>75.8</td>
<td>71.8</td>
</tr>
<tr>
<td>ELMo</td>
<td>❄️</td>
<td>91.7</td>
<td>91.8</td>
<td>79.6</td>
<td>86.3</td>
<td>86.1</td>
<td>76.0</td>
<td>75.9</td>
</tr>
<tr>
<td></td>
<td>🔥</td>
<td>91.9</td>
<td>91.2</td>
<td>76.4</td>
<td>83.3</td>
<td>83.3</td>
<td>74.7</td>
<td>75.5</td>
</tr>
<tr>
<td>Δ=🔥-❄️</td>
<td>0.2</td>
<td>-0.6</td>
<td>-3.2</td>
<td>-3.3</td>
<td>-2.8</td>
<td>-1.3</td>
<td>-0.4</td>
<td></td>
</tr>
<tr>
<td>BERT-base</td>
<td>❄️</td>
<td>92.2</td>
<td>93.0</td>
<td>84.6</td>
<td>84.8</td>
<td>86.4</td>
<td>78.1</td>
<td>82.9</td>
</tr>
<tr>
<td></td>
<td>🔥</td>
<td>92.4</td>
<td>93.5</td>
<td>84.6</td>
<td>85.8</td>
<td>88.7</td>
<td>84.8</td>
<td>87.1</td>
</tr>
<tr>
<td>Δ=🔥-❄️</td>
<td>0.2</td>
<td>0.5</td>
<td>0.0</td>
<td>1.0</td>
<td>2.3</td>
<td>6.7</td>
<td>4.2</td>
<td></td>
</tr>
</tbody>
</table>

- BERT is typically better if the whole network is fine-tuned, unlike ELMo

Peters, Ruder, Smith (2019)
# 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. acc.</td>
<td>misc.</td>
</tr>
<tr>
<td>SST-2</td>
<td>67k</td>
<td>1.8k</td>
<td>sentiment</td>
<td>acc.</td>
<td>movie reviews</td>
</tr>
</tbody>
</table>

**Single-Sentence Tasks**

<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>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. acc.</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>
</tbody>
</table>

**Similarity and Paraphrase Tasks**

<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>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>WNL1</td>
<td>634</td>
<td>146</td>
<td>coreference/NLI</td>
<td>acc.</td>
<td>fiction books</td>
</tr>
</tbody>
</table>

**Inference Tasks**

Wang et al. (2019)
<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></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<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 1
    
    \[...\quad \text{John} \quad \text{visited} \quad \text{Madagascar} \quad \text{yesterday} \quad ...\]

  - Epoch 2

- Whole word masking: don’t mask out parts of words.

  \[...\quad \_\text{John} \quad \_\text{visited} \quad \_\text{Madagascar} \quad \text{yesterday} \quad ...\]

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

<table>
<thead>
<tr>
<th>Model</th>
<th>data</th>
<th>bsz</th>
<th>steps</th>
<th>SQuAD (v1.1/2.0)</th>
<th>MNLI-m</th>
<th>SST-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>RoBERTa</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>with BOOKS + WIKI</td>
<td>16GB</td>
<td>8K</td>
<td>100K</td>
<td>93.6/87.3</td>
<td>89.0</td>
<td>95.3</td>
</tr>
<tr>
<td>+ additional data (§3.2)</td>
<td>160GB</td>
<td>8K</td>
<td>100K</td>
<td>94.0/87.7</td>
<td>89.3</td>
<td>95.6</td>
</tr>
<tr>
<td>+ pretrain longer</td>
<td>160GB</td>
<td>8K</td>
<td>300K</td>
<td>94.4/88.7</td>
<td>90.0</td>
<td>96.1</td>
</tr>
<tr>
<td>+ pretrain even longer</td>
<td>160GB</td>
<td>8K</td>
<td>500K</td>
<td>94.6/89.4</td>
<td>90.2</td>
<td>96.4</td>
</tr>
<tr>
<td>BERT\textsubscript{LARGE}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>with BOOKS + WIKI</td>
<td>13GB</td>
<td>256</td>
<td>1M</td>
<td>90.9/81.8</td>
<td>86.6</td>
<td>93.7</td>
</tr>
</tbody>
</table>

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)
No need to necessarily have a generative model (predicting words)

This objective is more computationally efficient (trains faster) than the standard BERT objective

Clark et al. (2020)
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
(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.)?
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.

Rogers et al. (2020)
(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.

Clark et al. (2019)
What does BERT learn?

- **Direct objects** attend to their verbs
  - 86.8% accuracy at the `dobj` relation

- **Noun modifiers** (e.g., determiners) attend to their noun
  - 94.3% accuracy at the `det` relation

- **Coreferent** mentions attend to their antecedents
  - 65.1% accuracy at linking the head of a coreferent mention to the head of an antecedent

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

Clark et al. (2019)
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)

(b) Evolution of accuracy on the MultiNLI-matched validation set when heads are pruned from BERT according to $I_h$ (solid blue) and accuracy difference (dashed green).

Michel et al. (2019)
GPT/GPT2
OpenAI 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

---

Radford et al. (2019)
Miley Cyrus was caught shoplifting from Abercrombie and Fitch on Hollywood Boulevard today.

The 19-year-old singer was caught on camera being escorted out of the store by security guards.

The singer was wearing a black hoodie with the label ‘Blurred Lines’ on the front and ‘Fashion Police’ on the back.

Scroll down for video

Shoplifting: Miley Cyrus was caught shoplifting from Abercrombie and Fitch on Hollywood Boulevard today (pictured)

The singer was wearing a black hoodie with the label ‘Blurred Lines’ on the front and ‘Fashion Police’ on the back

The singer was also wearing a pair of black-rimmed glasses, a black jacket, black jeans and black sandals.

She was carrying a pair of black and white striped gloves and a small black bag.
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

---

<table>
<thead>
<tr>
<th>Discriminator size</th>
<th>Unpaired Accuracy</th>
<th>Paired Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Generator size</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.5B</td>
<td>355M</td>
</tr>
<tr>
<td>Chance</td>
<td>50.0</td>
<td></td>
</tr>
<tr>
<td>Grover-Mega</td>
<td>92.0</td>
<td>98.5</td>
</tr>
<tr>
<td>Grover-Large</td>
<td>80.8</td>
<td>91.2</td>
</tr>
<tr>
<td>BERT-Large</td>
<td>73.1</td>
<td>75.9</td>
</tr>
<tr>
<td>GPT2</td>
<td>70.1</td>
<td>78.0</td>
</tr>
<tr>
<td>Grover-Base</td>
<td>70.1</td>
<td>80.0</td>
</tr>
<tr>
<td>BERT-Base</td>
<td>67.2</td>
<td>76.6</td>
</tr>
<tr>
<td>GPT2</td>
<td>66.2</td>
<td>71.9</td>
</tr>
<tr>
<td>11M FastText</td>
<td>63.8</td>
<td>65.6</td>
</tr>
</tbody>
</table>

Zellers et al. (2019)
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.)