BERT
AI2 made ELMo in spring 2018, GPT (transformer-based ELMo) was released in summer 2018, BERT came out October 2018

Four major changes compared to ELMo:

- Transformers instead of LSTMs
- Bidirectional model with “Masked LM” objective instead of standard LM
- Fine-tune instead of freeze at test time
- Operates over word pieces (byte pair encoding)
ELMo is a unidirectional model (as is GPT): we can concatenate two unidirectional models, but is this the right thing to do?

ELMo reprs 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)

- John visited Madagascar yesterday
- \( \text{visited} \quad \text{Madag.} \quad \text{yesterday} \quad \ldots \)

BERT

- \( \text{visited} \quad \text{Madag.} \quad \text{yesterday} \quad \ldots \)
- You could do this with a “one-sided” transformer, but this “two-sided” model can cheat

- John visited Madagascar yesterday

- \( \text{John} \quad \text{visited Madagascar yesterday} \)
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, mask out 15% of the tokens, and try to predict them.

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


Devlin et al. (2019)
BERT Architecture

- BERT Base: 12 layers, 768-dim per wordpiece token, 12 heads. Total params = 110M
- BERT Large: 24 layers, 1024-dim per wordpiece token, 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)
BERT can also do tagging by predicting tags at each word piece.
What can BERT do?

Entails  (first sentence implies second is true)

- 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
What can BERT NOT do?

- BERT cannot generate text (at least not in an obvious way)
  - Can fill in MASK tokens, but can’t generate left-to-right (well, you could put MASK at the end repeatedly, but this is slow)
- Masked language models are intended to be used primarily for “analysis” tasks
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

(b) Single Sentence Classification Tasks: SST-2, CoLA
### 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>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=.ov.</td>
<td></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>
</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=ov.</td>
<td></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>
</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></td>
<td></td>
<td></td>
<td>Single-Sentence Tasks</td>
<td></td>
<td></td>
</tr>
<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>
<tr>
<td></td>
<td></td>
<td></td>
<td>Similarity and Paraphrase Tasks</td>
<td></td>
<td></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. acc./F1</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></td>
<td></td>
<td></td>
<td>Inference Tasks</td>
<td></td>
<td></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>WNL1</td>
<td>634</td>
<td>146</td>
<td>coreference/NLI</td>
<td>acc.</td>
<td>fiction books</td>
</tr>
</tbody>
</table>

Wang et al. (2019)
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></td>
<td>392k</td>
<td>363k</td>
<td>108k</td>
<td>67k</td>
<td>8.5k</td>
<td>5.7k</td>
<td>3.5k</td>
<td>2.5k</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&lt;sub&gt;BASE&lt;/sub&gt;</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&lt;sub&gt;LARGE&lt;/sub&gt;</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)
RoBERTa

- “Robustly optimized BERT”
- 160GB of data instead of 16 GB
- Dynamic masking: standard BERT uses the same MASK scheme for every epoch, RoBERTa recomputes them
- 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 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)
Using BERT

- Huggingface Transformers: big open-source library with most pre-trained architectures implemented, weights available

- Lots of standard models...

  Model architectures

  - Transformers currently provides the following NLU/NLG architectures:
    1. **BERT** (from Google) released with the paper *BERT: Pre-training of Deep Understanding* by Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova.
    2. **GPT** (from OpenAI) released with the paper *Improving Language Under Radford, Karthik Narasimhan, Tim Salimans and Ilya Sutskever.
    4. **Transformer-XL** (from Google/CMU) released with the paper *Transformer Fixed-Length Context* by Zihang Dai*, Zhilin Yang*, Yiming Yang, and Jaime Carbonell.
    5. **XLNet** (from Google/CMU) released with the paper *XLNet: Generalized Autoregressive Pre-training* by Zhilin Yang*, Zihang Dai*, Yiming Yang, Jaime Carbonell.
    6. **XLM** (from Facebook) released together with the paper *Cross-lingual Language Model* and Alexis Conneau.
    7. **RoBERTa** (from Facebook), released together with the paper *Robustly Optimized Pre-trained Empirical Model Basing*...
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 when it came out in March 2019

<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>

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? (OpenAI pursued a “staged release” strategy and didn’t release biggest model)
Pre-Training Cost (with Google/AWS)

- BERT: Base $500, Large $7000
- Grover-MEGA (GPT-2 variant): $25,000
- XLNet (BERT variant): $30,000 — $60,000 (unclear)
- This is for a single pre-training run...developing new pre-training techniques may require many runs
  
  *Fine-tuning* these models can typically be done with a single GPU (but may take 1-3 days for medium-sized datasets)
NVIDIA: trained 8.3B parameter GPT model (5.6x the size of GPT-2)

Arguable these models are still underfit: larger models still get better held-out perplexities

NVIDIA blog (Narasimhan, August 2019)