Recall: Self-Attention

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

- Multiple “heads” analogous to different convolutional filters. Use parameters \( W_k \) and \( V_k \) to get different attention values + transform vectors
  \[ \alpha_{k,i,j} = \text{softmax}(x_i^T W_k x_j) \]  
  \[ x_{k,i}' = \sum_{j=1}^{n} \alpha_{k,i,j} V_k x_j \]  

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
- GPT/GPT2
- Analysis/Visualization

---

BERT

- 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

---

Devlin et al. (2019)

- 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.
How to learn a “deeply bidirectional” model? What happens if we just replace an LSTM with a transformer?

ELMo (Language Modeling)

 Bergen, Oslo, Copenhagen ...

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

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 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)
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)

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

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

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

Evaluation: GLUE

Results

RoBERTa

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

Results

RoBERTa

“Robustly optimized BERT”

Huge improvements over prior work (even compared to ELMo)

Effective at “sentence pair” tasks: textual entailment (does sentence A imply sentence B), paraphrase detection

Peters, Ruder, Smith (2019)

Wang et al. (2019)

Devlin et al. (2018)

Liu et al. (2019)
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 GPT2

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Layers</th>
<th>(d_{\text{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>

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? (OpenAI has only released 774M param model, not 1.5B yet)
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

Zellers et al. (2019)

Pre-Training Cost (with Google/AWS)

- BERT: Base $500, Large $7000
- Grover-MEGA: $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)


Pushing the Limits

- 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)

Google T5

- Colossal Cleaned Common Crawl: 750 GB of text
- We still haven't hit the limit of bigger data being useful

Raffel et al. (October 23, 2019)
BART

- Sequence-to-sequence BERT variant: permute/make/delete tokens, then predict full sequence autoregressively
- For downstream tasks: feed document into both encoder + decoder, use decoder hidden state as output
- Good results on dialogue, summarization tasks

Lewis et al. (October 30, 2019)

Analysis

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)
**Probing BERT**

- Try to predict POS, etc. from each layer.
  
  Learn mixing weights
  
  \[ h_{i,\tau} = \gamma_{\tau} \sum_{l=0}^{L} s(l) h_{i}^{(l)} \]
  
  representation of wordpiece \( i \) for task \( \tau \)

- Plot shows \( s \) weights (blue) and performance deltas when an additional layer is incorporated (purple)

- BERT “rediscovered the classical NLP pipeline”: first syntactic tasks then semantic ones

---

**Compressing BERT**

- Remove 60+% of BERT’s heads with minimal drop in performance

- DistilBERT (Sanh et al., 2019): nearly as good with half the parameters of BERT (via knowledge distillation)

---

**Open Questions**

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

- These techniques are here to stay, unclear what form will win out

- Role of academia vs. industry: no major pretrained model has come purely from academia

- Cost/carbon footprint: a single model costs $10,000+ to train (though this cost should come down)