CS 839: Foundation Models
Models II
Fred Sala
University of Wisconsin-Madison
Sept. 21, 2023
Announcements

• Logistics:
  • Check out some of the posted papers!
  • Homework will start next week

• Class roadmap:

<table>
<thead>
<tr>
<th>Thursday Sept. 21</th>
<th>Models II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tuesday Sept. 23</td>
<td>Prompting I</td>
</tr>
<tr>
<td>Thursday Sept. 28</td>
<td>Prompting II</td>
</tr>
<tr>
<td>Tuesday Oct. 3</td>
<td>Reasoning &amp; Chain-of-Thought</td>
</tr>
<tr>
<td>Thursday Oct. 5</td>
<td>In-Context Learning: Practice and Theory</td>
</tr>
</tbody>
</table>
Outline

• From Last Time: Encoder-only Models
  • Example: BERT, architecture, multitask training, fine-tuning

• Decoder-only Models
  • Example: GPT, architecture, basic functionality

• Variations and Advancements
  • Scaling, upgrades to positional encodings, etc
Questions/Clarifications From Last Time

1. **BERT initialization**
   • Random

2. **Differences** between decoders in encoder-decoder models and decoder-only models
   • Gets rid of cross-attention, still use masked self-attention

3. **Weight-tying** input and output embeddings
   • Trick to help performance (better usage of information, # parameters)
Outline

• From Last Time: Encoder-only Models
  • Example: BERT, architecture, multitask training, fine-tuning

• Decoder-only Models
  • Example: GPT, architecture, basic functionality

• Variations and Advancements
  • Scaling, upgrades to positional encodings, etc
**Why Encoder-Decoder?**

Wanted two things for translation:

- 1) Outputs in natural language
- 2) Tight alignment with input

What happens if we relax these?

1. Encoder-only models
2. Decoder-only models
Encoder-Only Models: BERT

Let’s get rid of the first part

• **1) Outputs** in natural language
  • **2) Tight alignment with input**

• Rip away decoders
  • Just stack encoders

![Diagram of BERT models]

**BERT Base**

**BERT Large**
Interlude: Word Embeddings

Q: Why is it called “BERT”?  
• A: In a sense, follows up ELMo

• Story:  
  • 2013: “Dense” word embeddings (Word2Vec, Glove)  
  • Capture some information  
  • Surprising properties!

https://nlp.stanford.edu/projects/glove/
Interlude: Word Embeddings

Q: Why is it called “BERT”?  
• A: In a sense, follows up ELMo

• Story:
  • 2013: “Dense” word embeddings (Word2Vec, Glove)
  • Downside: fixed representations per word
    • “Bank”: building or riverside?
  • Capturing context---one direction not sufficient!
    • I went to the bank to deposit a check
    • I went to the bank by the riverside

https://nlp.stanford.edu/projects/glove/
Interlude: Word Embeddings

Q: Why is it called “BERT”?  
A: In a sense, follows up ELMo

• Story:  
  • 2013: “Dense” word embeddings (Word2Vec, Glove)  
  • Downside: fixed representations per word  
    • “Bank”: building or riverside?  
  • Need: contextual representations  
    • Bidirectional!  
  • 2018: ELMo, BERT  
  • ELMo: uses LSTMs, BERT uses transformers

https://nlp.stanford.edu/projects/glove/
Interlude: Back to Self-Supervised Learning

• These representations aren’t for a *particular* task
• Once we fix a task, we can
  • Fine-tune them
  • Freeze them and build on top of them
**BERT: Forward Pass**

**BERT architecture**

- Rip away decoders
  - Just stack encoders
BERT: Using It

How to use it for e.g., classification?

• Special token/word [CLS]
  • Output representation here can be trained on (add classifier!)
BERT: Training

Training is more interesting!

• Pretraining. Then fine-tuning on task of interest

• Back to **self-supervised learning!**

• Two tasks for **pretraining.**

1. Masked Language Modeling
2. Next Sentence Prediction
**BERT: Training Task 1**

Masked Language Modeling Task

• Use [MASK] token for word to be predicted

• Which words to mask?
  • Original paper: 15% of words at random
  • But... of these
    • 10% of the time, no [MASK], flip word randomly
    • 10% of the time leave word unchanged
BERT: Fine-Tuning

Training is more interesting,
• Pretraining. Then fine-tuning on task of interest
BERT: Variations

Lots of work!

• Examples:
  • **RoBERTa**: better trained, better performance
    • 10x more data, no next-sentence prediction pretraining task
  • **SpanBERT**: masking *spans*, not just tokens!
  • **ALBERT**: parameter reduction (but same or better perf.)
    • How? Parameter tying/sharing cross layer, factorization

• Specializations: multilingual, domain-specific
  • BioBERT etc.
Break & Questions
Outline

• From Last Time: Encoder-only Models
  • Example: BERT, architecture, multitask training, fine-tuning

• Decoder-only Models
  • Example: GPT, architecture, basic functionality

• Variations and Advancements
  • Scaling, upgrades to positional encodings, etc
Decoder-Only Models: GPT

Let’s get rid of the first part
  • 1) Outputs in natural language
  • 2) Tight alignment with input

• Rip away encoders
  • Just stack decoders
Decoder-Only Models: GPT

Rip away encoders
- Just stack decoders
- Use causal masking! NB: not a *mask token* like in BERT
Decoder-Only Models: GPT

Rip away encoders
- Just stack decoders
- Decoders: get rid of **encoder** aspects (masked self-attention only)
From GPT2 to GPT3

Mainly make things larger!
- 96 decoder blocks (getting very tall)
- Context size: **2048**
- 175 billion parameters in total (800GB!)

**Training data:**

<table>
<thead>
<tr>
<th>Dataset</th>
<th># tokens</th>
<th>Proportion within training</th>
</tr>
</thead>
<tbody>
<tr>
<td>Common Crawl</td>
<td>410 billion</td>
<td>60%</td>
</tr>
<tr>
<td>WebText2</td>
<td>19 billion</td>
<td>22%</td>
</tr>
<tr>
<td>Books1</td>
<td>12 billion</td>
<td>8%</td>
</tr>
<tr>
<td>Books2</td>
<td>55 billion</td>
<td>8%</td>
</tr>
<tr>
<td>Wikipedia</td>
<td>3 billion</td>
<td>3%</td>
</tr>
</tbody>
</table>

https://en.wikipedia.org/wiki/GPT-3
Break & Questions
Outline

• From Last Time: Encoder-only Models
  • Example: BERT, architecture, multitask training, fine-tuning

• Decoder-only Models
  • Example: GPT, architecture, basic functionality

• Variations and Advancements
  • Scaling, upgrades to positional encodings, etc
Variations

Lots of new large language models
• Same basic idea for the architecture
• Example: PALM
  • Different activations, position embeddings, no biases
  • Scales!

<table>
<thead>
<tr>
<th>Model</th>
<th>Layers</th>
<th># of Heads</th>
<th>$d_{model}$</th>
<th># of Parameters (in billions)</th>
<th>Batch Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>PaLM 8B</td>
<td>32</td>
<td>16</td>
<td>4096</td>
<td>8.63</td>
<td>256 → 512</td>
</tr>
<tr>
<td>PaLM 62B</td>
<td>64</td>
<td>32</td>
<td>8192</td>
<td>62.50</td>
<td>512 → 1024</td>
</tr>
<tr>
<td>PaLM 540B</td>
<td>118</td>
<td>48</td>
<td>18432</td>
<td>540.35</td>
<td>512 → 1024 → 2048</td>
</tr>
</tbody>
</table>