Attention Mechanism in Neural Networks
Effective Approaches to Attention-based Neural Machine Translation

Minh-Thang Luong, Hieu Pham, Christopher D. Manning

Computer Science Department, Stanford University, Stanford, CA 94305
{lmthang,hyhieu,manning}@stanford.edu

Abstract

An attentional mechanism has lately been used to improve neural machine translation (NMT) by selectively focusing on parts of the source sentence during translation. However, there has been little work exploring useful architectures for attention-based NMT. This paper examines two simple and effective classes of attentional mechanism: a global approach which always attends to all source words and a local one that only looks at a subset of source words at a time. We demonstrate the effectiveness of both approaches on the WMT translation tasks between English and German in both directions. With local attention, we achieve a significant gain of 5.0 BLEU points over non-attentional systems that already incorporate known techniques such as dropout. Our ensemble model using different attention architectures yields a new state-of-the-art result in the WMT'15 English to German translation task with 25.9 BLEU points, an improvement of 1.0 BLEU points over the existing best system backed by NMT and an n-gram reranker.

1 Introduction

Neural Machine Translation (NMT) achieved state-of-the-art performances in large-scale translation tasks such as from English to French (Luong et al., 2015) and English to German (Jean et al., 2015). NMT is appealing since it requires minimal domain knowledge and is conceptually simple. The model by Luong et al. (2015) reads through all the source words until the end-of-sentence symbol \(<\text{eos}>\) is reached. It then starts emitting one target word at a time, as illustrated in Figure 1. NMT is often a large neural network that is trained in an end-to-end fashion and has the ability to generalize well to very long word sequences. This means the model does not have to explicitly store gigantic phrase tables and language models as in the case of standard MT; hence, NMT has a small memory footprint. Lastly, implementing NMT decoders is easy unlike the highly intricate decoders in standard MT (Koehn et al., 2003).

In parallel, the concept of "attention" has gained popularity recently in training neural networks, allowing models to learn alignments between different modalities, e.g., between image objects and agent actions in the dynamic control problem (Mnih et al., 2014), between speech frames and text in the speech recognition task (?), or between visual features of a picture and its text description in the image caption generation task (Xu et al., 2015). In the context of NMT, Bahdanau et al. (2015) has successfully applied such attentional mechanism to jointly translate and align words. To the best of our knowledge, there has not been any other work exploring the use of attention-based architectures for NMT.

In this work, we design, with simplicity and ef-
Encoder-Decoder with Attention
Encoder-Decoder with Attention
Attention Mechanisms

**Global Attention Model**

- Context vector $c_t$
- Global align weights $a_t$

**Local Attention Model**

- Context vector $c_t$
- Aligned position $p_t$
- Local weights $a_t$
Computing Attention Weights

\[ a_i = \frac{\exp(f_{att}(q, k_i))}{\sum_{j=1}^{\|M\|} \exp(f_{att}(q, k_j))} \]
Three ways of attention

Encoder-Decoder Attention

Encoder Self-Attention

MaskedDecoder Self-Attention
Transformer
Transformers

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis non erat sem.
Transformers
Attention Visualization

(a) Step 1
(b) Step 2
(c) Step 3
(d) Step 4
Related Papers

• Attention for Machine Translation

• Transformer
  • https://arxiv.org/abs/1706.03762

• Universal Transformer
  • https://arxiv.org/abs/1807.03819

• Transformer-XL
  • https://arxiv.org/abs/1901.02860
Useful Resources

• Source Codes:
  • https://github.com/tensorflow/tensor2tensor
  • https://github.com/tensorflow/models/tree/master/official/transformer/model

• Visualization of attention heads for BERT:
  • https://github.com/jessevig/bertviz

• Nice blogposts:
  • https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html
  • http://jalammar.github.io/illustrated-transformer/
  • http://nlp.seas.harvard.edu/2018/04/03/attention.html
  • http://jalammar.github.io/illustrated-bert/

• Blogpost comparing Transformers with Capsule Networks:
  • https://staff.fnwi.uva.nl/s.abnar/?p=108
Bidirectional Encoder Representations from Transformers (BERT)

Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova
Google AI Language

Presented by: Shantanu Chandra
Agenda

- Recap - Traditional Language Models
- Limitations
- BERT to the rescue!
- BERT Architecture and Training
- BERT Results and Discussion
- Summary
- Future work
Traditional Seq2Seq - recap
Traditional Seq2Seq
ELMo

- Bi-directional LSTM
- Lower layers - Syntactic
- Higher layers - Semantic
- Feature based use - feed embeddings to model
- Dependency on task-customized models
OpenAI GPT

- **Multi-layer Transformer architecture**
  (not shallow concat of independently trained LSTMs)

- **Fine tune usage** - fine tune the same base model for all end tasks

- Attention applied left-to-right, hence unidirectional!
BERT to the rescue!

- B - Bidirectional
- E - Encoder
- R - Representation
- T - Transformers
BERT

- **Bidirectional**
  - naturally bidirectional in all the layers. (ELMo is “shallow bidirectional”)

- **General purpose representations:** plug-and-play in real sense

- **Transformer Architecture** (new paradigm!)

- **Novel training methods** - not one, but TWO!!
BERT in numbers

- Pushed the GLUE benchmark to 80.4% (7.6% absolute improvement).
- Pushed MultiNLI accuracy to 86.7% (5.6% absolute improvement).
- Pushed the SQuAD v1.1 question-answering Test F1 to 93.2 (1.5 absolute improvement), outperforming human performance by 2.0.
BERT in numbers

- BERT - Base shattered OpenAI GPT
- BERT - Large beat BERT-Base
BERT to the rescue!

- B - Bidirectional
- E - Encoder
- R - Representation
- T - Transformers
B - Bidirectional

- Actually *non-directional*
Context is everything

- No Context (Word2Vec)
  - River [bank]
  - [Bank] deposit

- Left-to-right context
  - I made a [bank] deposit = I made a [....]

- Bi-directional context
  - I made a [bank] deposit = I made a [....] deposit
BERT to the rescue!

- B - Bidirectional
- E - Encoder
- R - Representation
- T - Transformers
Transformer Block
Transformer: Types of Attention

- **Encoder-Decoder Attention**

- **Encoder-Self Attention**

- **Decoder - Self Attention**
Transformer: in action

Source: Google AI Blog
Transformer: Self-Attention

Source: Tensor2Tensor
Transformer: Self-Attention
Transformer: Self-Attention

Encoder

\[ \theta_0 \rightarrow \theta_1 \rightarrow \theta_2 \rightarrow \theta_3 \rightarrow \theta_4 \rightarrow \theta_5 \rightarrow \theta_6 \]

Decoder

\[ d_0 \rightarrow d_1 \rightarrow d_2 \rightarrow d_3 \]

Knowledge is power <end>
BERT - Architecture

- Transformers - Encoder blocks only
- No weight sharing
- REMEMBER - attention mechanism!
- NOTE - residual connections
BERT - Architecture
**BERT - Input**

- **Input** = \((\text{inp\_len} \times \text{emb\_dim})\)

- **Output** = \((\text{inp\_len} \times \text{emb\_dim})\)

- **NOTE:** padding for equal length of batch

\[
Z = \begin{pmatrix}
\text{Hello} & 123.4 & 0.32 & \cdots & 94 & 32 \\
, & 83 & 34 & \cdots & 77 & 19 \\
\text{how} & 0.2 & 50 & \cdots & 33 & 30 \\
\text{are} & 289 & 432.98 & \cdots & 150 & 92 \\
\text{you} & 80 & 46 & \cdots & 23 & 32 \\
? & 41 & 21 & \cdots & 74 & 33
\end{pmatrix}
\]
BERT - Positional encoding

- Non-sequential input

- Hence Positional Encoding

- Non-learned, pre-determined sinusoidal functions $[-1,1]$

- Thus, same word in different position has different embedding in the SAME sentence.
BERT - Positional encoding

\[ p_{i,j} = \begin{cases} 
\sin \left( \frac{i}{10000^{\frac{j}{d_{emb\_dim}}}} \right) & \text{if } j \text{ is even} \\
\cos \left( \frac{i}{10000^{\frac{j-1}{d_{emb\_dim}}}} \right) & \text{if } j \text{ is odd}
\end{cases} \]

\[ P = \begin{bmatrix} 
\sin \left( \frac{0}{10000^{\frac{0}{d_{emb\_dim}}}} \right) & \cos \left( \frac{0}{10000^{\frac{0}{d_{emb\_dim}}}} \right) & \sin \left( \frac{0}{10000^{\frac{1}{d_{emb\_dim}}}} \right) & \cos \left( \frac{0}{10000^{\frac{1}{d_{emb\_dim}}}} \right) & \cdots \\
\sin \left( \frac{1}{10000^{\frac{0}{d_{emb\_dim}}}} \right) & \cos \left( \frac{1}{10000^{\frac{0}{d_{emb\_dim}}}} \right) & \sin \left( \frac{1}{10000^{\frac{1}{d_{emb\_dim}}}} \right) & \cos \left( \frac{1}{10000^{\frac{1}{d_{emb\_dim}}}} \right) & \cdots \\
\sin \left( \frac{2}{10000^{\frac{0}{d_{emb\_dim}}}} \right) & \cos \left( \frac{2}{10000^{\frac{0}{d_{emb\_dim}}}} \right) & \sin \left( \frac{2}{10000^{\frac{1}{d_{emb\_dim}}}} \right) & \cos \left( \frac{2}{10000^{\frac{1}{d_{emb\_dim}}}} \right) & \cdots \\
\sin \left( \frac{3}{10000^{\frac{0}{d_{emb\_dim}}}} \right) & \cos \left( \frac{3}{10000^{\frac{0}{d_{emb\_dim}}}} \right) & \sin \left( \frac{3}{10000^{\frac{1}{d_{emb\_dim}}}} \right) & \cos \left( \frac{3}{10000^{\frac{1}{d_{emb\_dim}}}} \right) & \cdots \\
\sin \left( \frac{4}{10000^{\frac{0}{d_{emb\_dim}}}} \right) & \cos \left( \frac{4}{10000^{\frac{0}{d_{emb\_dim}}}} \right) & \sin \left( \frac{4}{10000^{\frac{1}{d_{emb\_dim}}}} \right) & \cos \left( \frac{4}{10000^{\frac{1}{d_{emb\_dim}}}} \right) & \cdots \\
\sin \left( \frac{5}{10000^{\frac{0}{d_{emb\_dim}}}} \right) & \cos \left( \frac{5}{10000^{\frac{0}{d_{emb\_dim}}}} \right) & \sin \left( \frac{5}{10000^{\frac{1}{d_{emb\_dim}}}} \right) & \cos \left( \frac{5}{10000^{\frac{1}{d_{emb\_dim}}}} \right) & \cdots 
\end{bmatrix} \]
\[ X = Z + P \]

Input to the encoder block
BERT - Encoder Block

- **Multi-head Attention (h_i) modules:**
  - Multiple times,
  - ..with different weight matrices
  - ..then concat all and pass through linear
BERT - Encoder Block

- Multi-head Attention (h_i) modules:
  - Multiple times,
  - ..with different weight matrices
  - ..finally concat all and pass through linear

<table>
<thead>
<tr>
<th></th>
<th>Hello</th>
<th>,</th>
<th>how</th>
<th>are</th>
<th>you</th>
<th>?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$7.24 \times 10^{-06}$</td>
<td>$1.23 \times 10^{-21}$</td>
<td>$6.51 \times 10^{-40}$</td>
<td>$2.62 \times 10^{-22}$</td>
<td>$9.99 \times 10^{-01}$</td>
<td>$4.30 \times 10^{-08}$</td>
</tr>
<tr>
<td>how</td>
<td>$1.00 \times 10^{+00}$</td>
<td>$7.51 \times 10^{-30}$</td>
<td>$1.54 \times 10^{-17}$</td>
<td>$9.91 \times 10^{-13}$</td>
<td>$8.15 \times 10^{-09}$</td>
<td>$1.09 \times 10^{-30}$</td>
</tr>
<tr>
<td>are</td>
<td>$3.12 \times 10^{-70}$</td>
<td>$2.51 \times 10^{-51}$</td>
<td>$2.72 \times 10^{-21}$</td>
<td>$8.03 \times 10^{-09}$</td>
<td>$1.29 \times 10^{-07}$</td>
<td>$9.99 \times 10^{-01}$</td>
</tr>
<tr>
<td>you</td>
<td>$2.47 \times 10^{-72}$</td>
<td>$5.54 \times 10^{-05}$</td>
<td>$9.80 \times 10^{-01}$</td>
<td>$1.98 \times 10^{-02}$</td>
<td>$2.77 \times 10^{-82}$</td>
<td>$2.58 \times 10^{-08}$</td>
</tr>
<tr>
<td>?</td>
<td>$2.67 \times 10^{-05}$</td>
<td>$1.21 \times 10^{-09}$</td>
<td>$9.75 \times 10^{-07}$</td>
<td>$3.17 \times 10^{-07}$</td>
<td>$9.99 \times 10^{-01}$</td>
<td>$3.64 \times 10^{-28}$</td>
</tr>
<tr>
<td></td>
<td>$8.59 \times 10^{-47}$</td>
<td>$1.05 \times 10^{-35}$</td>
<td>$9.99 \times 10^{-01}$</td>
<td>$2.38 \times 10^{-15}$</td>
<td>$4.21 \times 10^{-27}$</td>
<td>$4.07 \times 10^{-06}$</td>
</tr>
</tbody>
</table>

= 1

= 1

= 1

= 1

= 1

= 1
BERT - Encoder Block

- **Feed Forward module:**
  
  ![Feed Forward Diagram](image)

  - FC linear layer: Input: `emb_dim`, Output: `hidden_dim`
  - RELU
  - FC linear layer: Input: `hidden_dim`, Output: `emb_dim`

- **Add, norm, dropout layer**

  ![Add, norm, dropout Diagram](image)
BERT - Training Tasks

- **Novel method 1: Masked Language model**
  - 15% input words *masked*.
    - 80% replaced by <MASK>
      - Eg: “My dog is <MASK>”
    - 10% replaced with random words
      - Eg: “My dog is **hotdog**”
  - 10% left intact
    - Eg: “My dog is **hairy**”
BERT - Training

- **Novel method 1: Masked Language model**
  - Network trained not to predict all the context words, but only the masked tokens.

- **Design Decision:**
  - Longer training time than other context predicting models (?)
    - Not really due to performance boost from attention module
BERT - Training

- Novel method 1: Masked Language model

  Design Decision:
  - Why replace with random words?
    - Would have only learned a contextual representation of ‘<MASK>’
  - Why not change/mask words for 1.5% of the time?
    - Bias the representation towards actual observed token
BERT - Training

- **Novel method 2**: Next sentence prediction

  - Feed pair of sentences separated by **[SEP]**
    - *50% times the following sentence*
    - *50% times random sentence*

```
Input = [CLS] the man went to [MASK] store [SEP]
       he bought a gallon [MASK] milk [SEP]
Label = IsNext

Input = [CLS] the man [MASK] to the store [SEP]
       penguin [MASK] are flight ##less birds [SEP]
Label = NotNext
```
BERT - Training

- **Novel method 2:** Next sentence prediction

  - **WHY?**
    - To make representation versatile (e.g., sentence level tasks)
    - Shows it helps
BERT - Corpus

- BookCorpus - 800M words
- English Wikipedia - 2,500M words - The ENTIRE WikiPedia!!
- Tokens tokenized using 37,000 WordPiece tokens
Random samples in batches of two (50% of the time adjacent to each other)

...such that combined length ≤ 512 tokens

15% masked from each sequence
BERT - Procedure

- Batch size = 256 sequences
- Each sequence of length 512 tokens
- Hence $256 \times 512 = 128,000$ tokens per batch
- NOTE: sequences can have more than 2 sentences.
The loss was calculated as:

\[ \mu_{\text{masked}_\text{LM}} + \mu_{\text{sent\_pred\_likelihood}} \]
BERT - Experiments

- Hyperparameters:
  - Batch-size: 16, 32
  - Learning rate (ADAM): 5e-5, 3e-5, 2e-5
  - Number of epochs: 3, 4
  - GELU activation
BERT - Experiments (GLUE)

- Evaluation tasks:
  - MNLI
  - QQP
  - QNLI
  - MRPC
  - WNLI
  - SST-2
  - CoLA
  - STS-B
  - RTE
BERT – Experiments (GLUE)

- Model used:

(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

(b) Single Sentence Classification Tasks: SST-2, CoLA
BERT - Results (GLUE)

Glue Scores

<table>
<thead>
<tr>
<th>Evaluation Task</th>
<th>Pre-OpenAI</th>
<th>ELMo</th>
<th>OpenAI GPT</th>
<th>BERT(B)</th>
<th>BERT(L)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNLI</td>
<td>80.6</td>
<td>82.4</td>
<td>84.8</td>
<td>86.7</td>
<td>88.5</td>
</tr>
<tr>
<td>QQP</td>
<td>66.3</td>
<td>70.7</td>
<td>72.1</td>
<td>74.1</td>
<td>77.3</td>
</tr>
<tr>
<td>QNLI</td>
<td>82.3</td>
<td>88.2</td>
<td>91.1</td>
<td>93.2</td>
<td>94.3</td>
</tr>
<tr>
<td>SST-2</td>
<td>35.3</td>
<td>45.2</td>
<td>52.9</td>
<td>60.5</td>
<td>64.1</td>
</tr>
<tr>
<td>CoLA</td>
<td>81.0</td>
<td>80.0</td>
<td>85.8</td>
<td>88.6</td>
<td>89.3</td>
</tr>
<tr>
<td>STS-b</td>
<td>74.7</td>
<td>75.3</td>
<td>77.2</td>
<td>79.3</td>
<td>81.9</td>
</tr>
<tr>
<td>MRPC</td>
<td>74.7</td>
<td>75.3</td>
<td>77.2</td>
<td>79.3</td>
<td>81.9</td>
</tr>
<tr>
<td>RTE</td>
<td>74.7</td>
<td>75.3</td>
<td>77.2</td>
<td>79.3</td>
<td>81.9</td>
</tr>
<tr>
<td>Avg</td>
<td>74.7</td>
<td>75.3</td>
<td>77.2</td>
<td>79.3</td>
<td>81.9</td>
</tr>
</tbody>
</table>
BERT - Experiments (SQuAD)

• Input Question:
  Where do water droplets collide with ice crystals to form precipitation?

• Input Paragraph:
  ... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals within a cloud. ... 

• Output Answer:
  within a cloud
BERT - Experiments (SQuAD)

- Model used:
BERT - Experiments (SQuAD)

- S: start vector of answer span
- $T_i$: Output at step 'i'

$$P_i = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}}$$

- Same for stop
<table>
<thead>
<tr>
<th>System</th>
<th>Dev EM</th>
<th>Dev F1</th>
<th>Test EM</th>
<th>Test F1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Leaderboard (Oct 8th, 2018)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Human</td>
<td>-</td>
<td>-</td>
<td>82.3</td>
<td>91.2</td>
</tr>
<tr>
<td>#1 Ensemble - nlnet</td>
<td>-</td>
<td>-</td>
<td>86.0</td>
<td>91.7</td>
</tr>
<tr>
<td>#2 Ensemble - QANet</td>
<td>-</td>
<td>-</td>
<td>84.5</td>
<td>90.5</td>
</tr>
<tr>
<td>#1 Single - nlnet</td>
<td>-</td>
<td>-</td>
<td>83.5</td>
<td>90.1</td>
</tr>
<tr>
<td>#2 Single - QANet</td>
<td>-</td>
<td>-</td>
<td>82.5</td>
<td>89.3</td>
</tr>
<tr>
<td><strong>Published</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BiDAF+ELMo (Single)</td>
<td>-</td>
<td>85.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>R.M. Reader (Single)</td>
<td>78.9</td>
<td>86.3</td>
<td>79.5</td>
<td>86.6</td>
</tr>
<tr>
<td>R.M. Reader (Ensemble)</td>
<td>81.2</td>
<td>87.9</td>
<td>82.3</td>
<td>88.5</td>
</tr>
<tr>
<td><strong>Ours</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BERT_{BASE} (Single)</td>
<td>80.8</td>
<td>88.5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BERT_{LARGE} (Single)</td>
<td>84.1</td>
<td>90.9</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BERT_{LARGE} (Ensemble)</td>
<td>85.8</td>
<td>91.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BERT_{LARGE} (Sgl.+TriviaQA)</td>
<td>84.2</td>
<td>91.1</td>
<td>85.1</td>
<td>91.8</td>
</tr>
<tr>
<td>BERT_{LARGE} (Ens.+TriviaQA)</td>
<td>86.2</td>
<td>92.2</td>
<td>87.4</td>
<td>93.2</td>
</tr>
</tbody>
</table>

Table 2: SQuAD results. The BERT ensemble is 7x systems which use different pre-training checkpoints and fine-tuning seeds.
## BERT - Results (Named Entity Recognition)

<table>
<thead>
<tr>
<th>System</th>
<th>Dev F1</th>
<th>Test F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELMo+BiLSTM+CRF</td>
<td>95.7</td>
<td>92.2</td>
</tr>
<tr>
<td>CVT+Multi (Clark et al., 2018)</td>
<td>-</td>
<td>92.6</td>
</tr>
<tr>
<td>BERT_{BASE}</td>
<td>96.4</td>
<td>92.4</td>
</tr>
<tr>
<td>BERT_{LARGE}</td>
<td>96.6</td>
<td>92.8</td>
</tr>
</tbody>
</table>
A girl is going across a set of monkey bars. She
(i) jumps up across the monkey bars.
(ii) struggles onto the bars to grab her head.
(iii) gets to the end and stands on a wooden plank.
(iv) jumps up and does a back flip.
## BERT - Results (SWAG)

<table>
<thead>
<tr>
<th>System</th>
<th>Dev</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESIM+GloVe</td>
<td>51.9</td>
<td>52.7</td>
</tr>
<tr>
<td>ESIM+ELMo</td>
<td>59.1</td>
<td>59.2</td>
</tr>
<tr>
<td>BERT&lt;sub&gt;BASE&lt;/sub&gt;</td>
<td>81.6</td>
<td>-</td>
</tr>
<tr>
<td>BERT&lt;sub&gt;LARGE&lt;/sub&gt;</td>
<td><strong>86.6</strong></td>
<td><strong>86.3</strong></td>
</tr>
<tr>
<td>Human (expert)&lt;sup&gt;†&lt;/sup&gt;</td>
<td>-</td>
<td>85.0</td>
</tr>
<tr>
<td>Human (5 annotations)&lt;sup&gt;†&lt;/sup&gt;</td>
<td>-</td>
<td>88.0</td>
</tr>
</tbody>
</table>

Table 4: SWAG Dev and Test accuracies. Test results were scored against the hidden labels by the SWAG authors. <sup>†</sup>Human performance is measured with 100 samples, as reported in the SWAG paper.
BERT - Ablation Studies

- Pre-training tasks
  - MLM and no NSP (effect of NSP)
  - LTR and no NSP (effect of MLM)
    - i.e, OpenAI GPT architecture
## BERT - Ablation Studies

### Pre-training tasks

- **MLM and no NSP (effect of NSP)**

<table>
<thead>
<tr>
<th>Tasks</th>
<th>MNLI-m (Acc)</th>
<th>QNLI (Acc)</th>
<th>Dev Set</th>
<th>SST-2 (Acc)</th>
<th>SQuAD (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT&lt;sub&gt;BASE&lt;/sub&gt;</td>
<td>84.4</td>
<td>88.4</td>
<td>86.7</td>
<td>92.7</td>
<td>88.5</td>
</tr>
<tr>
<td>No NSP</td>
<td>83.9</td>
<td>84.9</td>
<td>86.5</td>
<td>92.6</td>
<td>87.9</td>
</tr>
<tr>
<td>LTR &amp; No NSP</td>
<td>82.1</td>
<td>84.3</td>
<td>77.5</td>
<td>92.1</td>
<td>77.8</td>
</tr>
<tr>
<td>+ BiLSTM</td>
<td>82.1</td>
<td>84.1</td>
<td>75.7</td>
<td>91.6</td>
<td>84.9</td>
</tr>
</tbody>
</table>
## BERT - Ablation Studies

- **Pre-training tasks**
  - **Effect of MLM**

<table>
<thead>
<tr>
<th>Tasks</th>
<th>MNLI-m (Acc)</th>
<th>QNLI (Acc)</th>
<th>Dev Set MRPC (Acc)</th>
<th>SST-2 (Acc)</th>
<th>SQuAD (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT_BASE</td>
<td>84.4</td>
<td>88.4</td>
<td>86.7</td>
<td>92.7</td>
<td>88.5</td>
</tr>
<tr>
<td>No NSP</td>
<td>83.9</td>
<td>84.9</td>
<td>86.5</td>
<td>92.6</td>
<td>87.9</td>
</tr>
<tr>
<td>LTR &amp; No NSP</td>
<td>82.1</td>
<td>84.3</td>
<td>77.5</td>
<td>92.1</td>
<td>77.8</td>
</tr>
<tr>
<td>+ BiLSTM</td>
<td>82.1</td>
<td>84.1</td>
<td>75.7</td>
<td>91.6</td>
<td>84.9</td>
</tr>
</tbody>
</table>
BERT - Ablation Studies

- **Model Size**
  - Less data, Huge model - still works

<table>
<thead>
<tr>
<th>Hyperparams</th>
<th>Dev Set Accuracy</th>
<th>MNLI-m</th>
<th>MRPC</th>
<th>SST-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>#L  #H  #A  LM (ppl)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3   768  12  5.84</td>
<td>77.9</td>
<td>79.8</td>
<td>88.4</td>
<td></td>
</tr>
<tr>
<td>6   768  3   5.24</td>
<td>80.6</td>
<td>82.2</td>
<td>90.7</td>
<td></td>
</tr>
<tr>
<td>6   768  12  4.68</td>
<td>81.9</td>
<td>84.8</td>
<td>91.3</td>
<td></td>
</tr>
<tr>
<td>12  768  12  3.99</td>
<td>84.4</td>
<td>86.7</td>
<td>92.9</td>
<td></td>
</tr>
<tr>
<td>12  1024 16  3.54</td>
<td>85.7</td>
<td>86.9</td>
<td>93.3</td>
<td></td>
</tr>
<tr>
<td>24  1024 16  3.23</td>
<td>86.6</td>
<td>87.8</td>
<td>93.7</td>
<td></td>
</tr>
</tbody>
</table>
BERT - Ablation Studies

- **Training Steps**
  - YES, more steps = higher performance
  - Converges *slower*, but performance *HIGHER*
BERT - Final Remarks and Salient Points

- Bi-Directional learning

- 2 Novel pre-training tasks (PROVED to be better)

- New Input representation

- Very comprehensive and FAIR comparisons/experiments
Extending BERT

- Multi-Task Learning - natural extension of BERT
- Studying what do different layers learn? (cue: ELMo)
- What if syntactic information added?
Thank you and have a Happy Easter!
Q and A