BERT
Bidirectional Encoder Representations from Transformers

April 7th, 2020  http://adl.miulab.tw
BERT: Bidirectional Encoder Representations from Transformers

- Idea: contextualized word representations
  - Learn word vectors using long contexts using Transformer instead of LSTM

**BERT #1 – Masked Language Model**

- Idea: language understanding is **bidirectional** while LM only uses *left* or *right* context.

  - Use the output of the masked word’s position to predict the masked word.

  - Randomly mask 15% of tokens:
    - Too little: expensive to train
    - Too much: not enough context

  - Possible classes: All English words
    - 0.1%: Aardvark
    - 0.1%: Improvisation
    - 0.1%: Zyzzyva

- [Link](http://jalammar.github.io/illustrated-bert/)
BERT #1 – Masked Language Model

BERT #2 – Next Sentence Prediction

Idea: modeling *relationship* between sentences
- QA, NLI etc. are based on understanding inter-sentence relationship

**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

Idea: modeling relationship between sentences

Predict likelihood that sentence B belongs after sentence A

FFNN + Softmax

Tokenized Input

http://jalammar.github.io/illustrated-bert/
### BERT – Input Representation

- **Input embeddings contain**
  - Word-level token embeddings
  - Sentence-level segment embeddings
  - Position embeddings

#### Diagram

![Diagram showing input representation with BERT components](image_url)

- **Token Embeddings**
  - \( E_{[CLS]} \), \( E_{my} \), \( E_{dog} \), \( E_{is} \), \( E_{cute} \), \( E_{[SEP]} \), \( E_{he} \), \( E_{likes} \), \( E_{play} \), \( E_{##ing} \), \( E_{[SEP]} \)

- **Segment Embeddings**
  - \( E_A \), \( E_A \), \( E_A \), \( E_A \), \( E_A \), \( E_A \), \( E_B \), \( E_B \), \( E_B \), \( E_B \), \( E_B \)

- **Position Embeddings**
  - \( E_0 \), \( E_1 \), \( E_2 \), \( E_3 \), \( E_4 \), \( E_5 \), \( E_6 \), \( E_7 \), \( E_8 \), \( E_9 \), \( E_{10} \)

---

BERT Training

- Training data: Wikipedia + BookCorpus
- 2 BERT models
  - BERT-Base: 12-layer, 768-hidden, 12-head
  - BERT-Large: 24-layer, 1024-hidden, 16-head
**BERT Fine-Tuning for Understanding Tasks**

- Idea: simply learn a classifier/tagger built on the top layer for each target task

**BERT Overview**

1 - **Semi-supervised** training on large amounts of text (books, wikipedia..etc).

The model is trained on a certain task that enables it to grasp patterns in language. By the end of the training process, BERT has language-processing abilities capable of empowering many models we later need to build and train in a supervised way.

**Semi-supervised Learning Step**

**Model:**

**Dataset:**

**Objective:**
Predict the masked word (language modeling)

2 - **Supervised** training on a specific task with a labeled dataset.

**Supervised Learning Step**

**Model:** (pre-trained in step #1)

**Dataset:**

<table>
<thead>
<tr>
<th>Email message</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buy these pills</td>
<td>Spam</td>
</tr>
<tr>
<td>Win cash prizes</td>
<td>Spam</td>
</tr>
<tr>
<td>Dear Mr. Atreides, please find attached...</td>
<td>Not Spam</td>
</tr>
</tbody>
</table>
## BERT Results on NER

<table>
<thead>
<tr>
<th>Model</th>
<th>Description</th>
<th>CONLL 2003 F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>TagLM (Peters+, 2017)</td>
<td>LSTM BiLM in BLSTM Tagger</td>
<td>91.93</td>
</tr>
<tr>
<td>ELMo (Peters+, 2018)</td>
<td>ELMo in BLSTM</td>
<td>92.22</td>
</tr>
<tr>
<td>BERT-Base (Devlin+, 2019)</td>
<td>Transformer LM + fine-tune</td>
<td>92.4</td>
</tr>
<tr>
<td>CVT Clark</td>
<td>Cross-view training + multitask learn</td>
<td>92.61</td>
</tr>
<tr>
<td>BERT-Large (Devlin+, 2019)</td>
<td>Transformer LM + fine-tune</td>
<td>92.8</td>
</tr>
<tr>
<td>Flair</td>
<td>Character-level language model</td>
<td>93.09</td>
</tr>
</tbody>
</table>

BERT Results with Different Model Sizes

- Improving performance by increasing model size

Idea: use pre-trained BERT to get contextualized word embeddings and feed them into the task-specific models.

The output of each encoder layer along each token's path can be used as a feature representing that token.

But which one should we use?
What is the best contextualized embedding for “Help” in that context?
For named-entity recognition task CoNLL-2003 NER

<table>
<thead>
<tr>
<th>Layer Type</th>
<th>Embedding</th>
<th>Dev F1 Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Layer</td>
<td></td>
<td>91.0</td>
</tr>
<tr>
<td>Last Hidden Layer</td>
<td></td>
<td>94.9</td>
</tr>
<tr>
<td>Sum All 12 Layers</td>
<td></td>
<td>95.5</td>
</tr>
<tr>
<td>Second-to-Last Hidden Layer</td>
<td></td>
<td>95.6</td>
</tr>
<tr>
<td>Sum Last Four Hidden</td>
<td></td>
<td>95.9</td>
</tr>
<tr>
<td>Concat Last Four Hidden</td>
<td></td>
<td>96.1</td>
</tr>
</tbody>
</table>

http://jalammar.github.io/illustrated-bert/
**ERNIE: Enhanced Representation through Knowledge Integration**

- BERT models local cooccurrence between tokens, while characters are modeled independently
  - 哈(ha), 爾(er), 濱(bin) instead 哈爾濱(Harbin)
- ERNIE incorporates knowledge by masking semantic units/entities

Diagram showing the difference between what BERT and ERNIE learn.
Concluding Remarks

- Contextualized embeddings learned from masked LM via Transformers provide informative cues for **transfer learning**

- BERT – a general approach for learning contextual representations from Transformers and benefiting language understanding

  ✓ Pre-trained BERT: [https://github.com/google-research/bert](https://github.com/google-research/bert) [https://github.com/huggingface/transformers](https://github.com/huggingface/transformers)