Nature Language Process Final Project

Team Yellow

Date : 2020/06/19

- Speaker : Noah Wang 0760054
- Team Member : Noah Wang 0760054

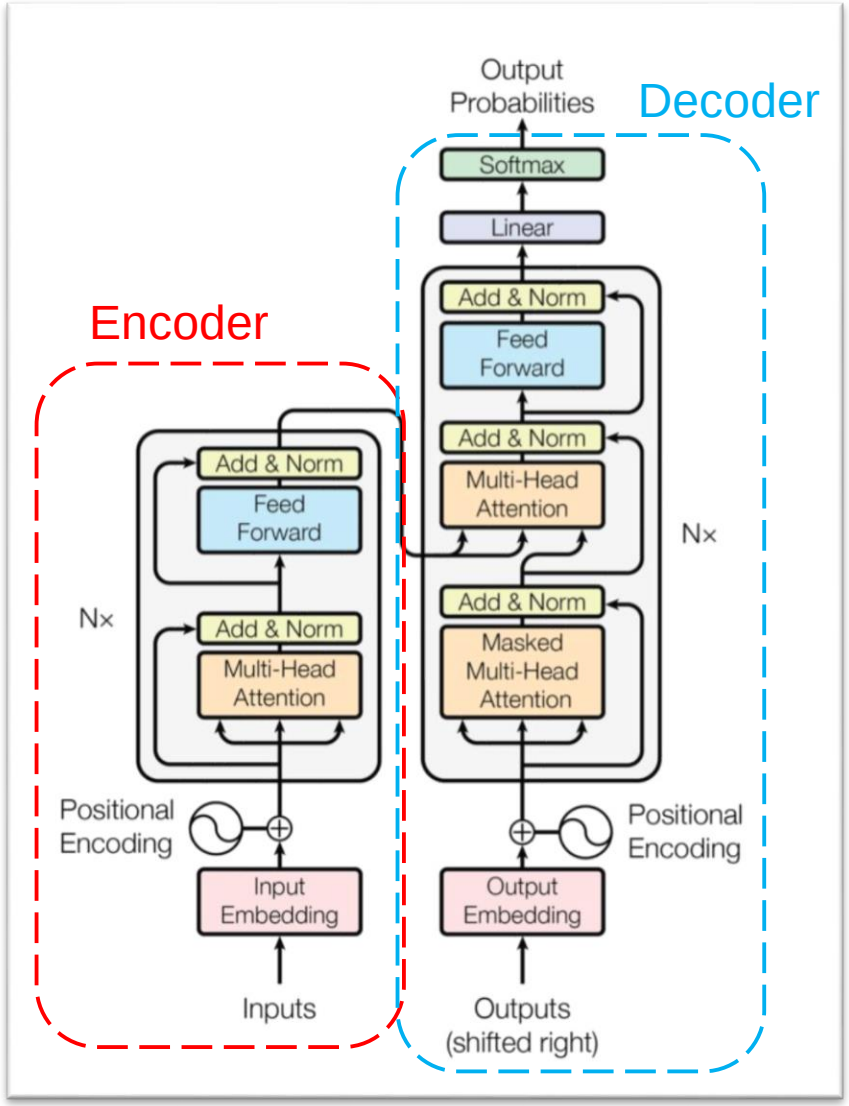
Outline

/01

Transformers

Transformers

- BERT can be consider as an Encoder of Transformers



Sequence to Sequence

- Translate Task
- Summarize Task
- …

Multi-head Self-attention

- Replace RNN Layer

Positional Encoding

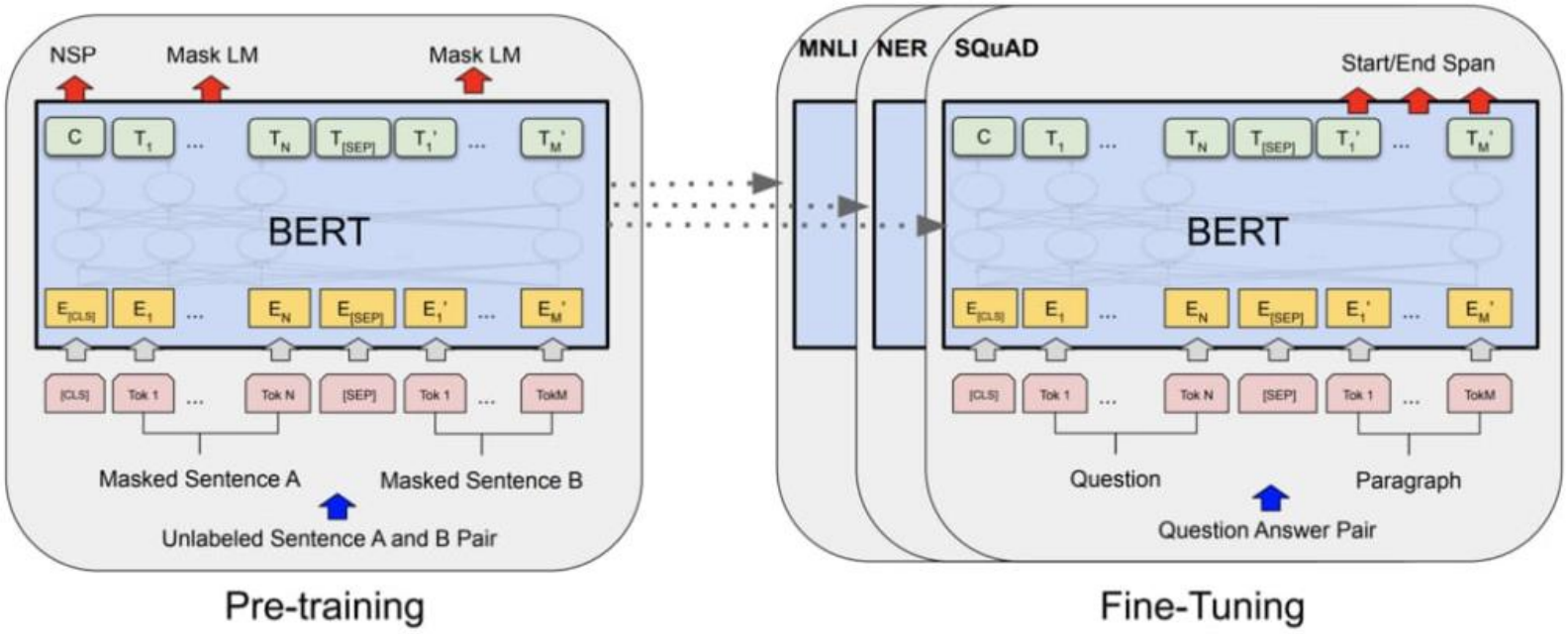
- Add a position information in self-attention

/02

BERT Language Model

Bidirectional **E**ncoder **R**epresentation from **T**ransformers

Bidirectional Encoder Representation from Transformers



Pre-training

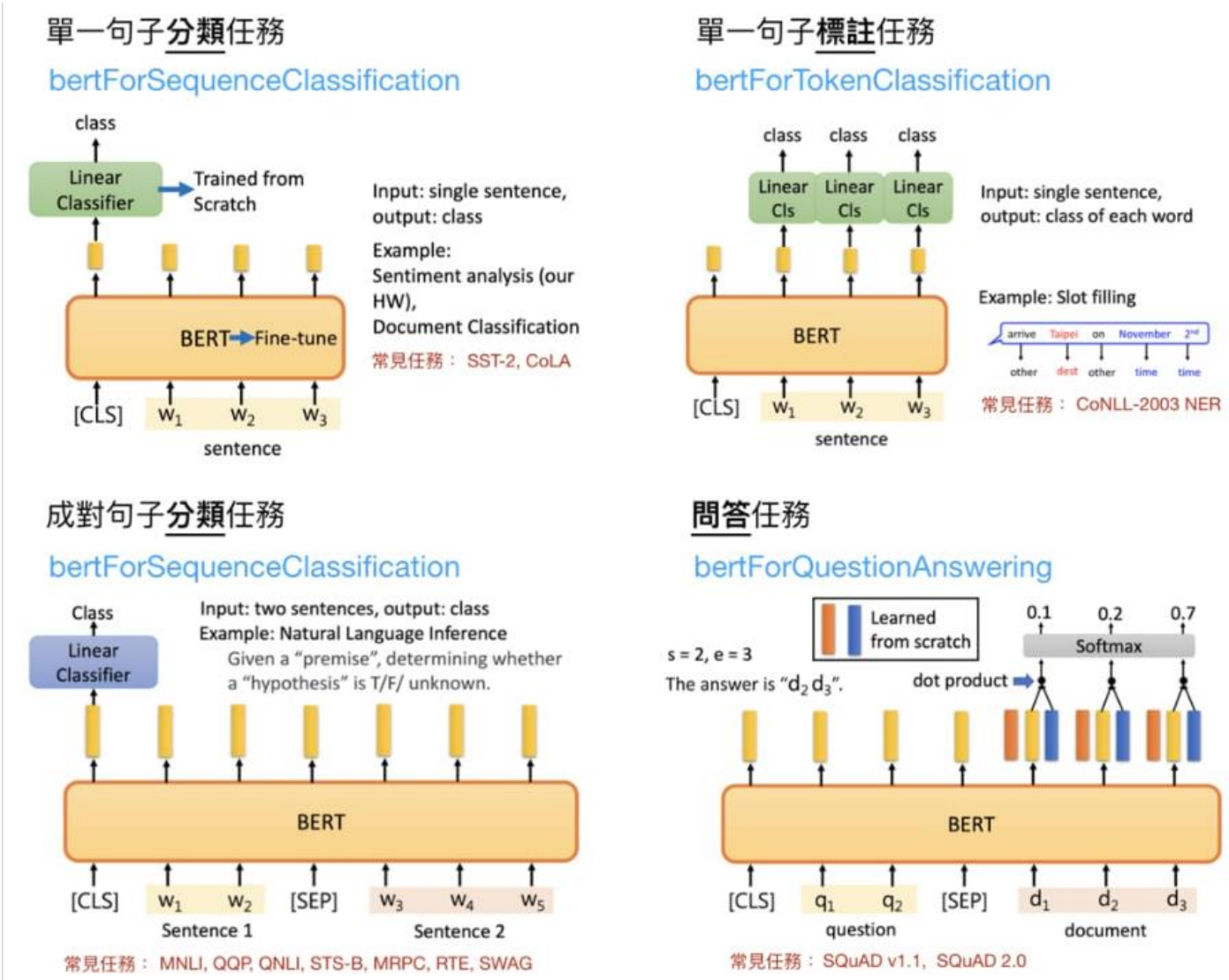
- **BERT-BASE Model** with 12-layer and 110 million parameters could train 4 days using 16 TPU chips.
- There are varies Pre-trained BERT Model been released by the author,
  - BERT-Large, Uncased (Whole Word Masking)
  - BERT-Large, Cased (Whole Word Masking)
  - BERT-Base, Uncased
  - So on….

Fine-Tuning

- Tuning the last few layers to apply on specific task.
- Cloze Task :
  - ➢ This **[MASK]** a book. → This **is** a book.
- Prediction Task :
  - ➢ **[CLS]** How are you doing ? **[SEP]** Great !
    - → **[CLS] = True**

Fine-Tuning BERT Model

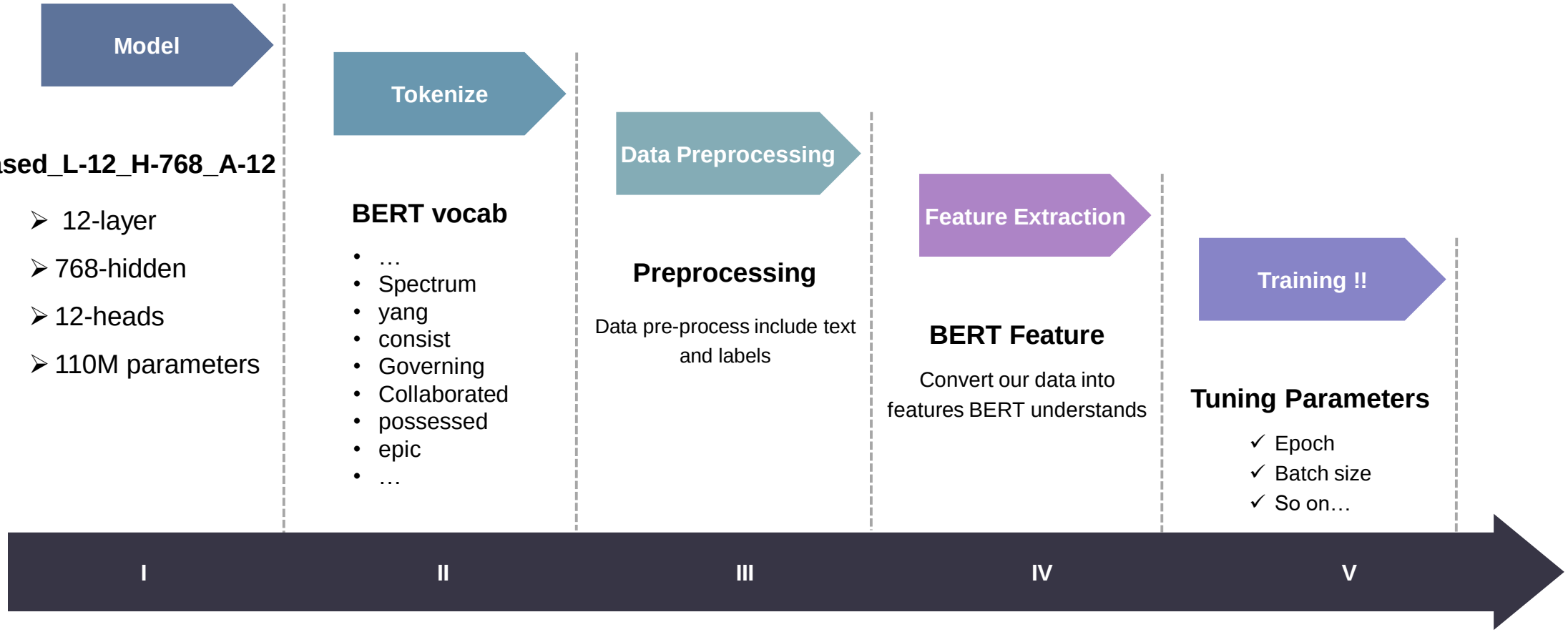
- **Basic :**
  - bertModel
  - bertTokenizer
- **Pre-trained :**
  - bertForMaskedLM
  - bertForNextSentencePrediction
  - bertForPreTraining
- **Fine-tuning :**
  - bertForSequenceClassification
  - bertForTokenClassification
  - bertForQuestionAnswering
  - bertForMultipleChoice



/03

Proposed Structure

Proposed Structure



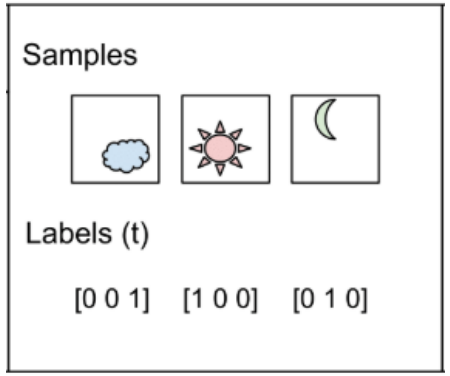
Model

- Refer to works from *Kaggle's Toxic Comment Classification Challenge*.
- Each data has 1 to 6 labels among 43 labels.
- **[CLS]** text + reply

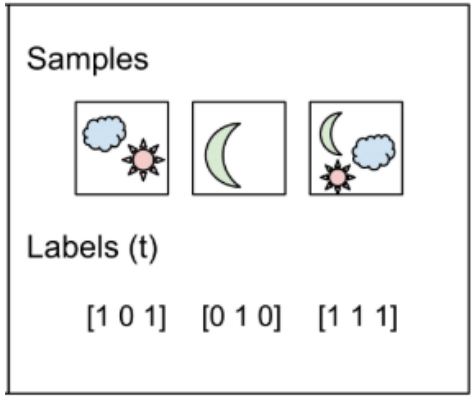
Uncased_L-12_H-768_A-12

- 12-layer
- 768-hidden
- 12-heads
- 110M parameters

Multi-class classification



Multi-label classification



Tokenize

- Text normalization

  John Johanson's, → john johanson's,

- Punctuation splitting

  john johanson's, → john johanson ' s ,

- WordPiece tokenization

  john johanson ' s , → john johan ##son ' s ,

BERT vocab
(about 30,000 words)

….
Visit
33
Evening
Search
Grant
Effort
Solo
Treatment
Buried
Republican
Primarily
Bottom
Owner
1970s
….

```
print(text_list[0])
tokenizer.tokenize(text_list[0])
```

```
we can all agree that any song by Niall Horan.
['we',
 'can',
 'all',
 'agree',
 'that',
 'any',
 'song',
 'by',
 'niall',
 'ho',
 '##ran',
 '.']
```

Data Preprocessing

Data Merging

In order to utilize information from reply, we simply merge “text” and “reply” as “text_reply” with same idx.

Categories Binarizing

For categories, we list out all the categories with each data and binarize them into 1 or 0.

Labels Definition

We define our label as a list with 43 elements which are simply binary information, i.e. label = [1,0,1,…,1].

Feature Extraction

Using BERT to extract fixed feature vectors.

Feature Extraction

Zero Padding

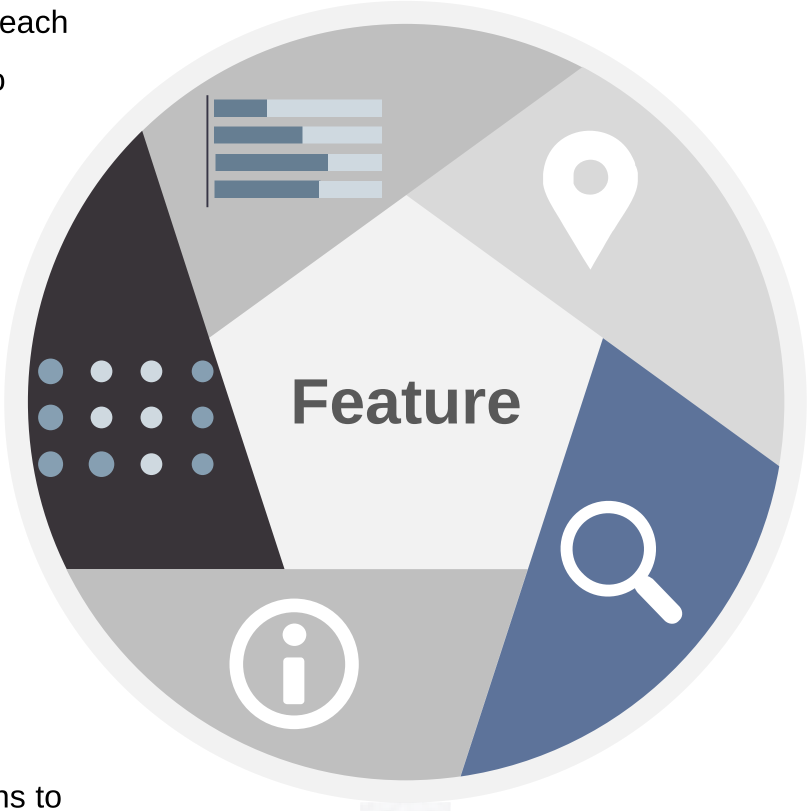
- In order to parallelize operations, we need to fill each input sequence in the batch with zero padding to ensure that its length is consistent.

Special Token

- [CLS] : Label token
- [PAD] : zero padding mask
- [UNK] : unknown word
- [SEP] : Sentence separate

Mask

- To distinguish the range of self-attention, 1 means to pay close attention, 0 means no.



Positional Embedding

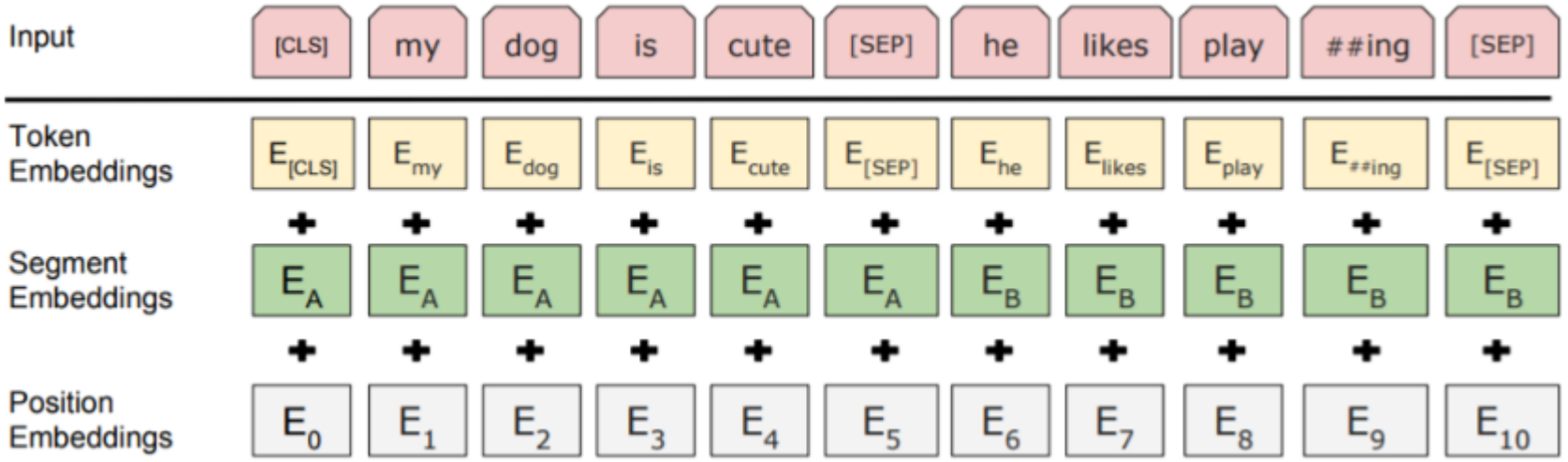
- Positional information.

Segment Embedding

- Recognize the separation of sentences, 0 for first sentence, 1 for second one.
- <u>All the data were embedded as 0 in our work.</u>

```
Token ID :
[101, 1030, 2402, 3207, 4478, 1035, 1045, 2228, 2065, 1057, 5831, 1998, 11122, 2072, 7510, 6928, 2017, 10657, 4530, 13451, 24185, 4430, 2036, 102, 0, 0, 0, 0, 0
Mask :
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
Segment embedding :
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
Label ID :
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
```

Training !!



Batch Size

Batch is set as 8 or 16 based on the epoch number and max training time limit.

16

Loss Function

We use sigmoid to get the probabilities instead of softmax.

$$\frac{1}{1+e^{-x}}$$

Epoch Number

Epoch number is set as 8 in our work.

8

Learning Rate

Learning rate is set as 3e - 5.

3e-5

/04

Results

Results

Results

Team Name	P (all) ▲	P1 (GIF w/ text) ▲	P2 (GIF only) ▲
Team Alpha	0.515 (1)	0.499 (1)	0.525 (1)
Team_India	0.479 (2)	0.437 (4)	0.506 (3)
Team_Papa	0.477 (3)	0.406 (6)	0.523 (2)
Team Yellow	0.473 (4)	0.456 (2)	0.485 (7)
NCTU_Team_Golf	0.470 (5)	0.441 (3)	0.488 (6)

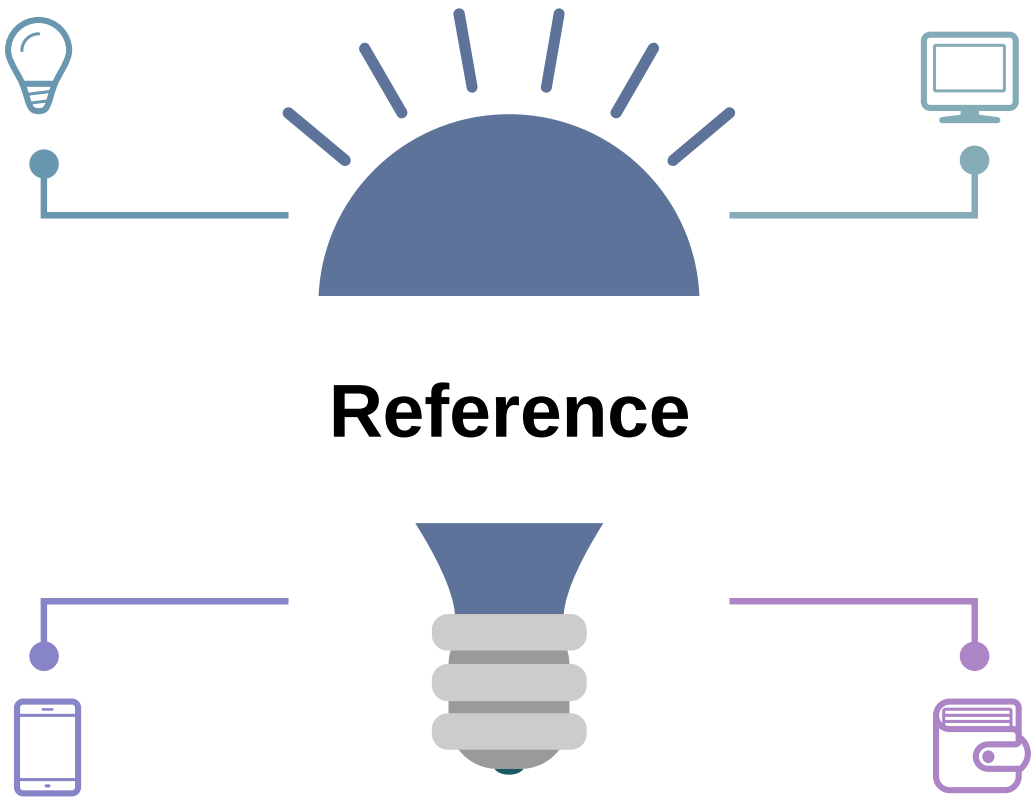
Training results

- Example number : 28800
- Batch size : 16
- Epoch : 8
- Global step : 14400
- Training time : about 4 hrs
- Final loss : 0.13366494

Predict results

```
----------------------------------------------------------------------------------------------------------------------------------------------------------------
[0.0037810206413269043, 0.004612833261489868, 0.002726644277572632, 0.001063019037246704, 0.001396268606185913, 0.0011425316333770752, 0.001496642827987
[('0.723295', 'yes'), ('0.431268', 'agree'), ('0.038485', 'good_luck'), ('0.035561', 'win'), ('0.029237', 'you_got_this'), ('0.023304', 'no')]
['yes', 'agree', 'good_luck', 'win', 'you_got_this', 'no']
----------------------------------------------------------------------------------------------------------------------------------------------------------------
[0.3780688941478729, 0.038095325231552124, 0.023927271366119385, 0.008916646242141724, 0.01858338713645935, 0.010083168745040894, 0.024433910846710205,
[('0.488286', 'applause'), ('0.396191', 'yes'), ('0.236198', 'happy_dance'), ('0.203073', 'slow_clap'), ('0.173208', 'dance'), ('0.113904', 'win')]
['applause', 'yes', 'happy_dance', 'slow_clap', 'dance', 'win']
----------------------------------------------------------------------------------------------------------------------------------------------------------------
[0.776212751865387, 0.015515685081481934, 0.006968498229980469, 0.0010664761066436768, 0.007569402456283569, 0.007955104112625122, 0.01178237795829773,
[('0.279888', 'yes'), ('0.177015', 'agree'), ('0.079209', 'no'), ('0.040416', 'applause'), ('0.033239', 'ok'), ('0.032701', 'eye_roll')]
['yes', 'agree', 'no', 'applause', 'ok', 'eye_roll']
----------------------------------------------------------------------------------------------------------------------------------------------------------------
[0.07347774505615234, 0.12711471319198608, 0.030222177505493164, 0.0800132155418396, 0.010818302631378174, 0.001774519681930542, 0.0014872252941131592,
[('0.320326', 'idk'), ('0.274703', 'oops'), ('0.264315', 'shrug'), ('0.154736', 'sorry'), ('0.130699', 'scared'), ('0.081891', 'deal_with_it')]
['idk', 'oops', 'shrug', 'sorry', 'scared', 'deal_with_it']
----------------------------------------------------------------------------------------------------------------------------------------------------------------
```

Reference Link



1. https://arxiv.org/pdf/1810.04805.pdf
2. https://github.com/google-research/bert/blob/master/README.md
3. https://leemeng.tw/attack_on_bert_transfer_learning_in_nlp.html
4. https://towardsdatascience.com/building-a-multi-label-text-classifier-using-bert-and-tensorflow-f188e0ecdc5d
5. https://gombru.github.io/2018/05/23/cross_entropy_loss/
6. https://zhuanlan.zhihu.com/p/46833276
7. https://github.com/javaidnabi31/Multi-Label-Text-classification-Using-BERT/blob/master/multi-label-classification-bert.ipynb

Thanks For Listening

Speaker : Noah Wang 0760054