Bert_Review (1)

August 26, 2020

0.1 Bert

0.1.1 01 what is Bert?

1. Bert Architecture
2. State of Art

0.1.2 02 Bert Architecture

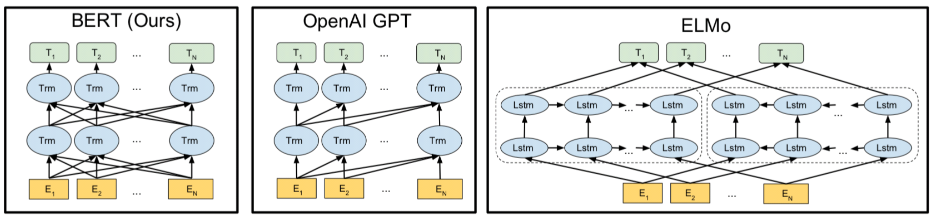
1. Pre-training task
2. Input Embedding
3. Encoding
4. Fine-tuning

1. RNN ( ) : hidden layer /
2. LSTM : RNN hidden state cell state
3. Transformer (Encoder-Decoder): RNN base Encoding ( ) Decoding ( )
4. Attention: Transformer
5. Bert : Transformer Encoder Self Attenion (Bi-directional) ,

0.2 BERT (Bi-directional Encoder Representation from Transformers)

Purpose: /

- ELMO
- ELMO : Uni-directional LSTM Model



alt text

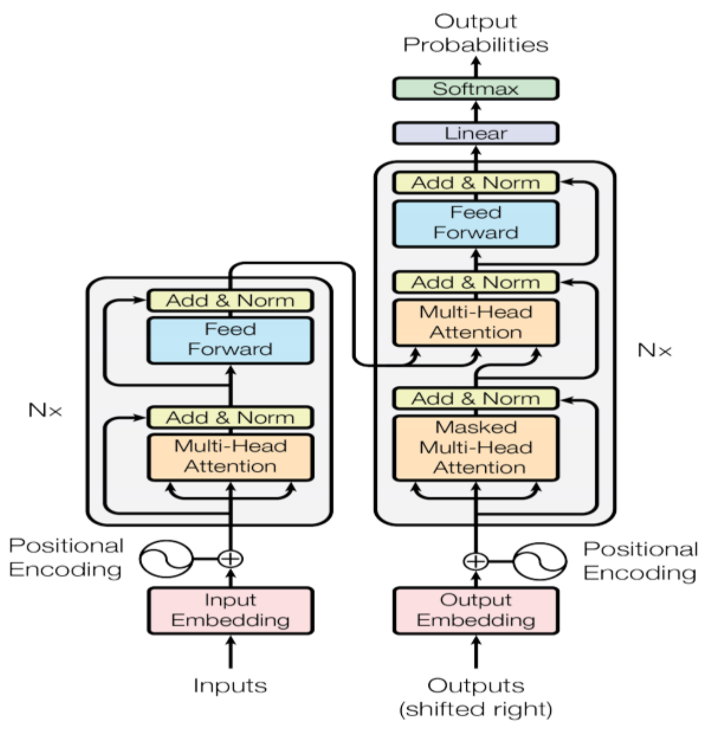


Figure 1: The Transformer - model architecture.

alt text

- GPT : left to right, and right to left transformer model
- NLP 11 state of art
- SQuAD
- Transfer Learning Fine -tuning .
- Transformer
  - Encoder
  - Speed, Accuracy , Long-term decedency

0.3 BERT ARCHITECTURE

'Attention is All' Transformer Encoding

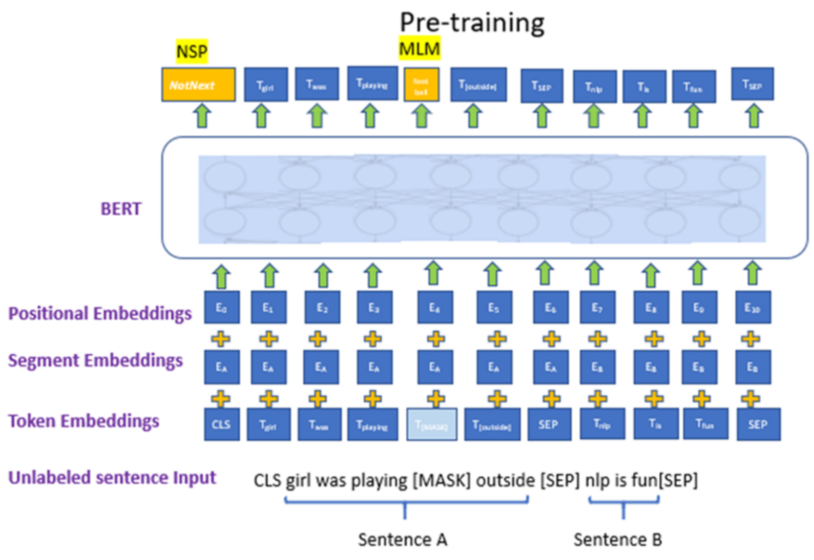
1. Pre-training task
2. Fine-Tuning

2 : * Bert-Base (L=12, H=768, A=12) * Bert-Large(L=24, H=1024, A=16) * Base

0.4 1. Pre - training

0.4.1 1. pre-training tasks

```
* Masked Language Modelling
    *  15%     <MASK>
```



alt text

```
    * 80% ['MASK'] Token
    * 10% Random , 10%
    *   , mask   ,
* Next Sentence Prediction
  * B   A
  * 50%
  * B  A   IsNext   ,  NotNext
    * ) Input = [CLS] the man went to [MASK] store [SEP] he bought a gallon [MASK] milk [S
    * ) Input = [CLS] the man [MASK] to the store [SEP] penguin [MASK] are flight ##less b
```

0.4.2 2. Embedding :

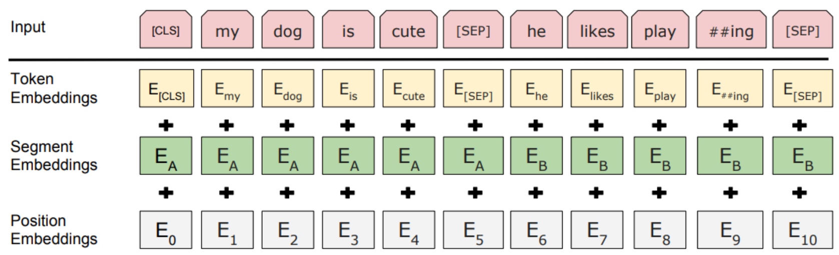

Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

Figure 2: BERT input representation. The input embeddings is the sum of the token embeddings, the segmentation embeddings and the position embeddings.

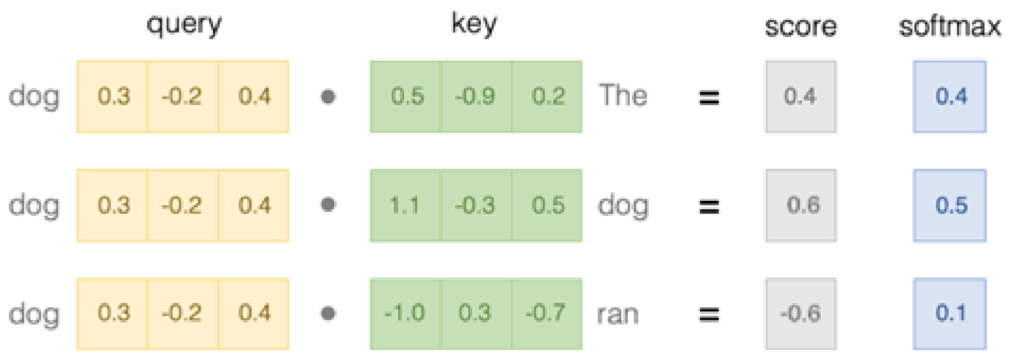
Token Vectorize

Input = Token Embedding + Segment Embedding + Position Embedding

- Position Encoding Postion embedding
- Token : Vector
- Segment : / (0, 1 etc)
- Position: /

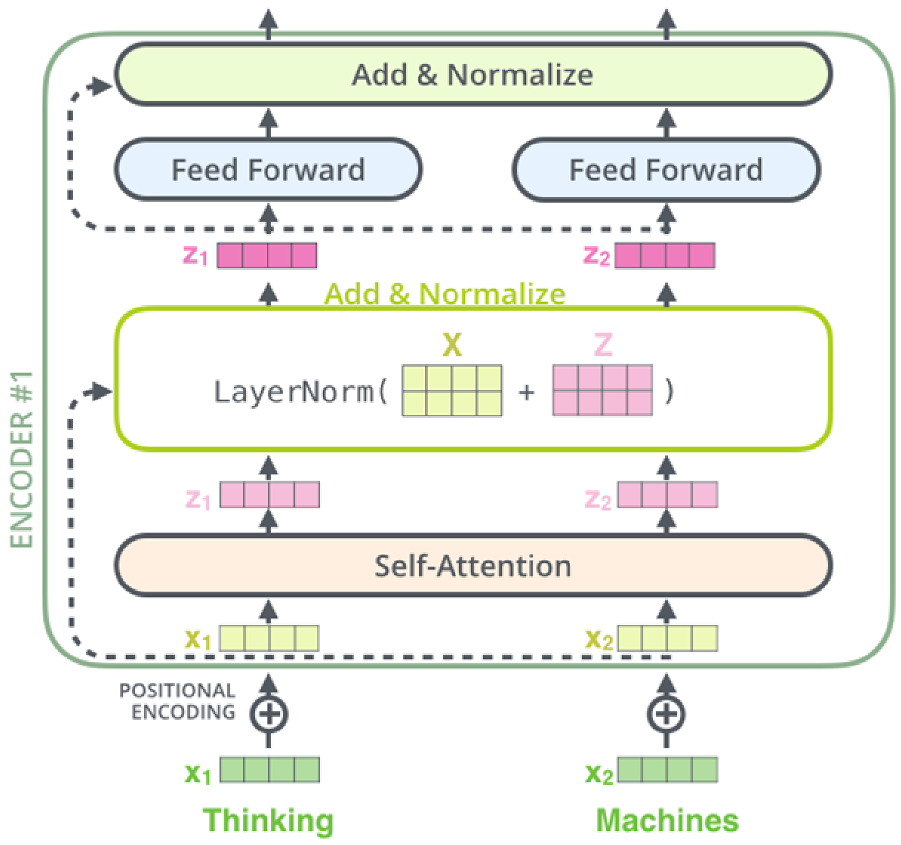
- : 3
  - (Base 768 Vector Large 1024)
  - + + 768 vector Embedding

0.4.3 3. Encoding Layers



1. Multi-Head Attention : / 1. 768 input vector 12 head 64 Vector Q,K,V . 2. Q K vector .) dog Q [0.3, -0.2 0.4] * 'The' K [0.5, -0.9, 0.2] = 0.4 3. soft max 0~1 scaling Score) [0.4 0,6 -0.6] –> [0.4 0.5 0.1] 4. softmax(score) V vector (64) 5. 12 Head 12 64 vector sum 768 vector

```
Encoding Layer  N Layer
```



2. Add & Normalize
   - ResNet : Embedding input vector Self-Attention Normalize

- Normalize: , Layer Stablize

0.5 2. Fine-Tuning

- Bert Tasks:
  - Text similarity : text corpus Binary Classification
  - Reply Matching : text corpus Binary Classification
  - Intent Classificaition : / classification
    - Sentiment Analysis
    - Question and Answer
    - Name Entity Recognition
- Feature-based vs. Fine-tuning
  - Feature- based approach : task network Feature , network (ELMO)
  - Fine-tuning approach : Pre-trained parameter downstream task
    - NER: Pretrained Bert CONLL (NNP Person, Organization Labelling system) Labelling Supervised Learning

```
In [ ]:
```