SUNG KYUN KWAN UNIVERSITY

Large Language Model(LLM)

Transformer & BERT

Artificial Intelligence Project

SWE3032-41
Prof. Hogun Park
TA Jiwon Jeong

성균관대학교

Table of Contents

Part 1

Language Model

Introduction

- Language Model is a model that represents the probability of a sentence
    - Predict the probability of occurrence of the sentence itself
    - A model to predict the next word given the previous words
- 버스 정류장에서 방금 버스를 OOO.
    - 사랑해
    - 고양이
    - 굿바이
    - 큰일남
    - 놓쳤다

Objective :

- $D = \{x^i\}_{i=1}^N$
- $\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmax}} \sum_{i=1}^N \log P(x_{1:n}^i\,;\,\theta)\,, where\ x_{1:n} = \{x_1, \ldots, x_n\}$.

Chain Rule :

- We can convert joint probability to conditional probability.

$$\begin{aligned} P(A,B,C,D) &= P(D|A,B,C)P(A,B,C) \\ &= P(D|A,B,C)P(C|A,B)P(A,B) \\ &= P(D|A,B,C)P(C|A,B)P(B|A)P(A) \end{aligned}$$

By Chain Rule,

- We can re-write the equation,

$$
\begin{aligned}
P(x_{1:n}) &= P(x_1, \cdots, x_n) \\
&= P(x_n | x_1, \cdots, x_{n-1}) \cdots P(x_2 | x_1) P(x_1) \\
&= \prod_{i=1}^{n} P(x_i | x_{<i})
\end{aligned}
$$

$$
\log P(x_{1:n}) = \sum_{i=1}^{N} \log P(x_i | x_{<i})
$$

By Chain Rule,

- We can re-write the Objective,

$$\mathcal{D} = \{x^i\}_{i=1}^{N}$$

$$\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmax}} \sum_{i=1}^{N} \log P(x_{1:n}^i; \theta)$$

$$= \underset{\theta \in \Theta}{\operatorname{argmax}} \sum_{i=1}^{N} \sum_{j=1}^{n} \log P(x_j^i | x_{<j}^i; \theta)$$

$$\text{where } x_{1:n} = \{x_1, \cdots, x_n\}.$$

Using Language Model

- Pick better(fluent) sentence
- Predict next word given previous words.

$$\hat{x}_t = \underset{\mathbf{x}_t \in \mathcal{X}}{\operatorname{argmax}} \log P(\mathbf{x}_t | x_{<t}; \theta)$$

Part 2

Transformer

Attention is all you need

Attention is **all you need**

A Vaswani, N Shazeer, N Parmar… - Advances in neural …, 2017 - proceedings.neurips.cc

… to attend to **all** positions in the decoder up to and including that position. **We need** to prevent … **We** implement this inside of scaled dot-product **attention** by masking out (setting to −∞) …

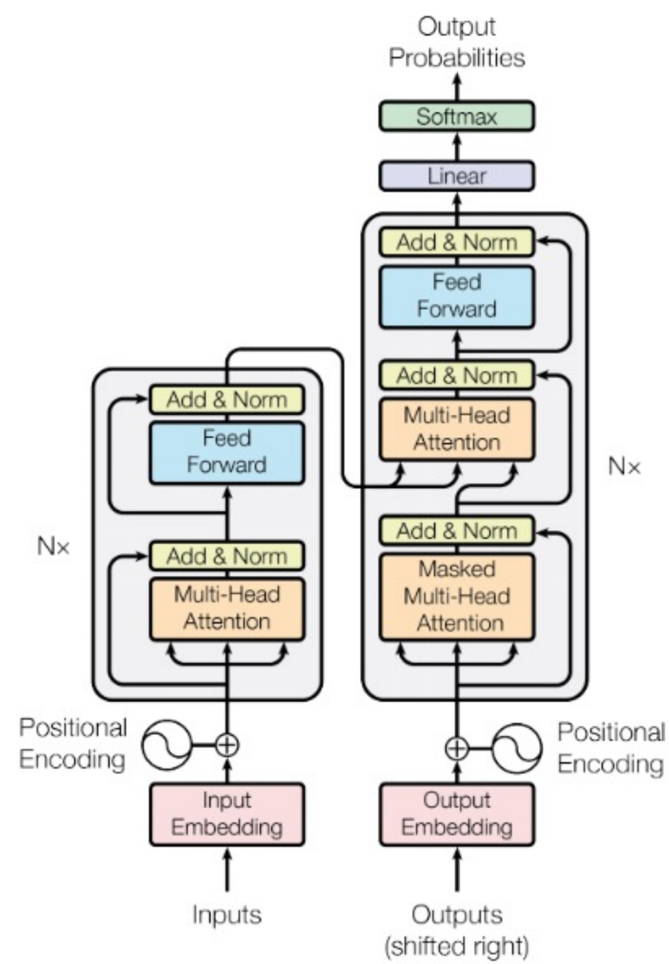
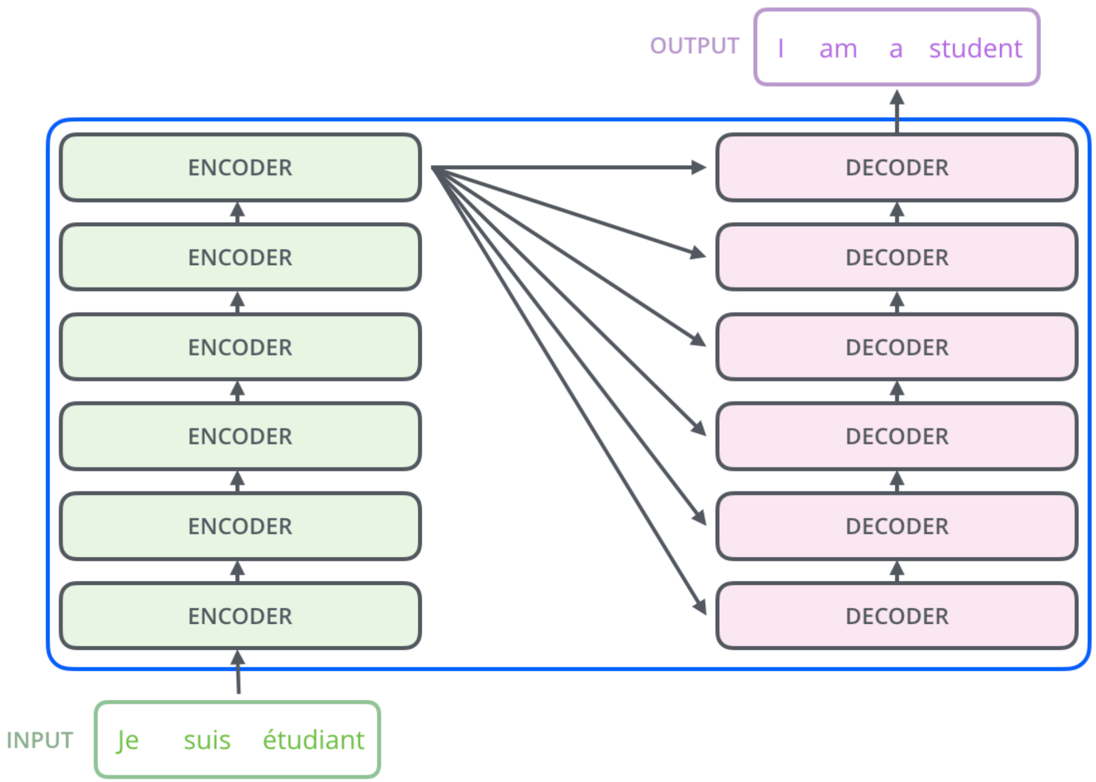


Figure 1: The Transformer - model architecture.

A High-Level Look

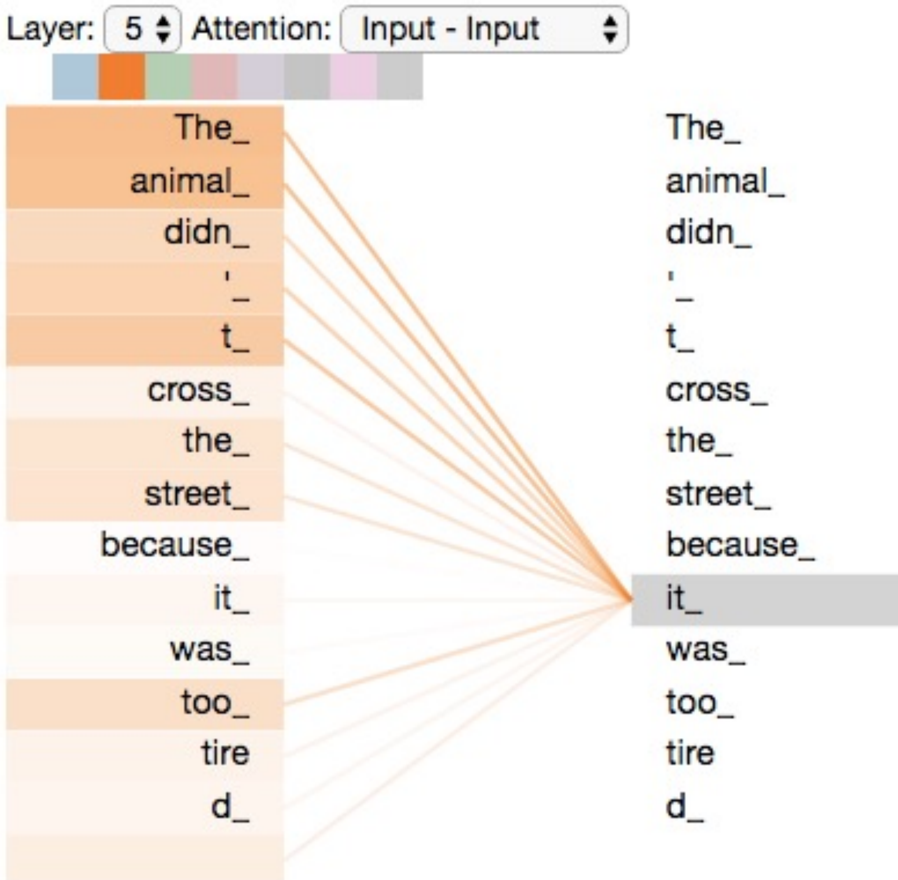
- The encoding component is a stack of encoders
- The decoding component is a stack of decoders of the same number



Self-Attention at a High Level

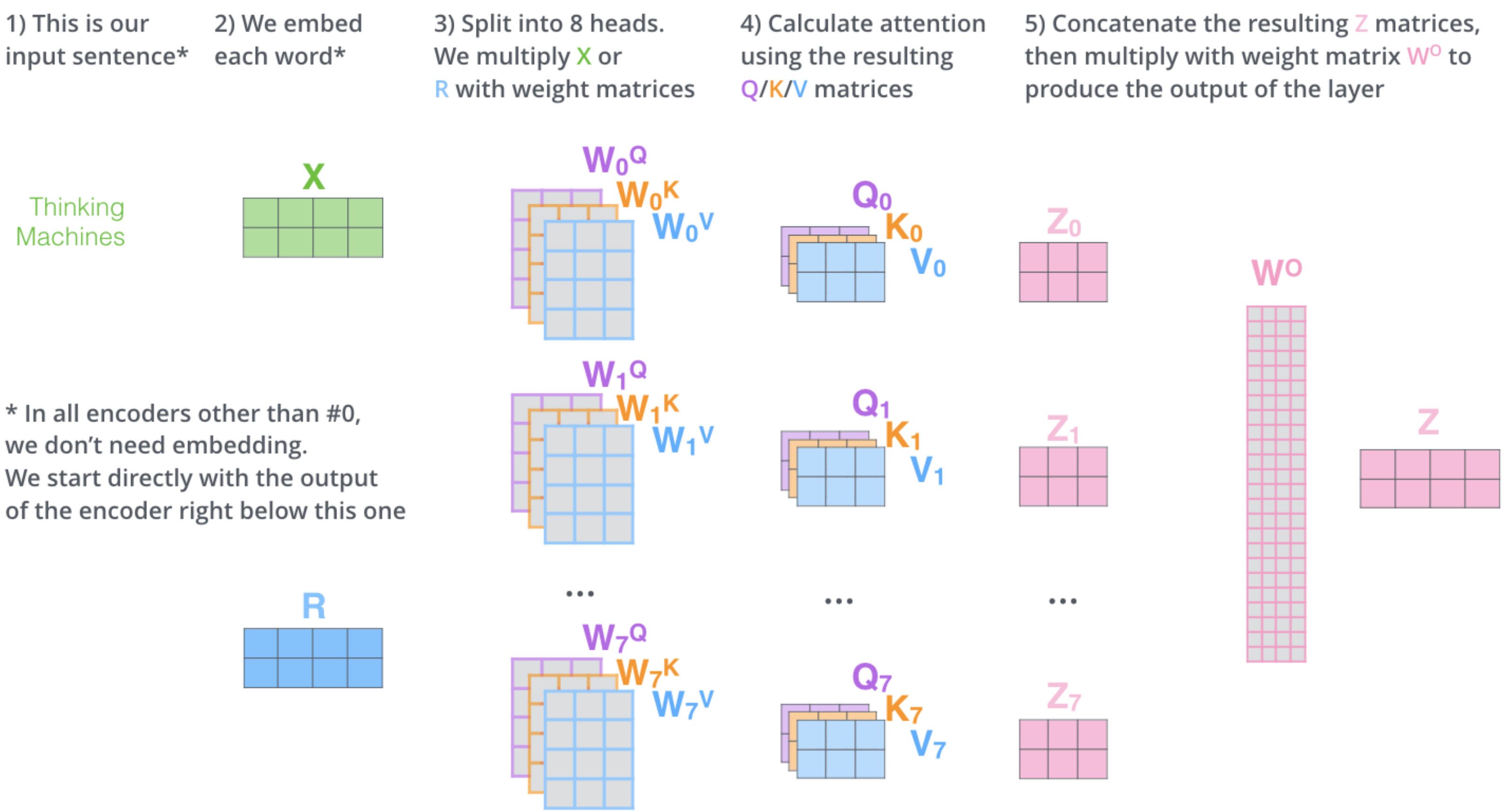
- Input sentence to translate :
  - "The animal didn't cross the street because it was too tired"
- What does "`it`" in this sentence refer to? `Street` or `animal`?
  - Simple question to a human but not as simple to an algorithm
- Self attention allows it to look at other positions in the input sequence for clues that can help lead to a better encoding for this word.
- Self-attention is the method the Transformer uses to bake the "**understanding**" of other relevant words into the one we're currently processing.

Self-Attention Example



PART 2 Transformer

Multi-Head Attention



Masked Multi-Head Attention

- Do not need to be done sequentially, but can be done at one batch

Example:	Features: position 1	2	3	4	Labels
1	robot	must	obey	orders	must
2	robot	must	obey	orders	obey
3	robot	must	obey	orders	orders
4	robot	must	obey	orders	<eos>

Masked Multi-Head Attention

Queries

robot	must	obey	orders

X

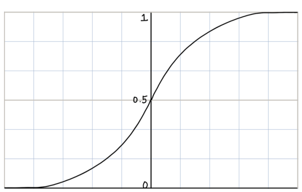
Keys

robot	must	obey	orders
robot	must	obey	orders
robot	must	obey	orders
robot	must	obey	orders

=

Scores
(before softmax)

0.11	0.00	0.81	0.79
0.19	0.50	0.30	0.48
0.53	0.98	0.95	0.14
0.81	0.86	0.38	0.90



Scores
(before softmax)

0.11	0.00	0.81	0.79
0.19	0.50	0.30	0.48
0.53	0.98	0.95	0.14
0.81	0.86	0.38	0.90

Apply Attention Mask →

Masked Scores
(before softmax)

0.11	−inf	−inf	−inf
0.19	0.50	−inf	−inf
0.53	0.98	0.95	−inf
0.81	0.86	0.38	0.90

Masked Scores
(before softmax)

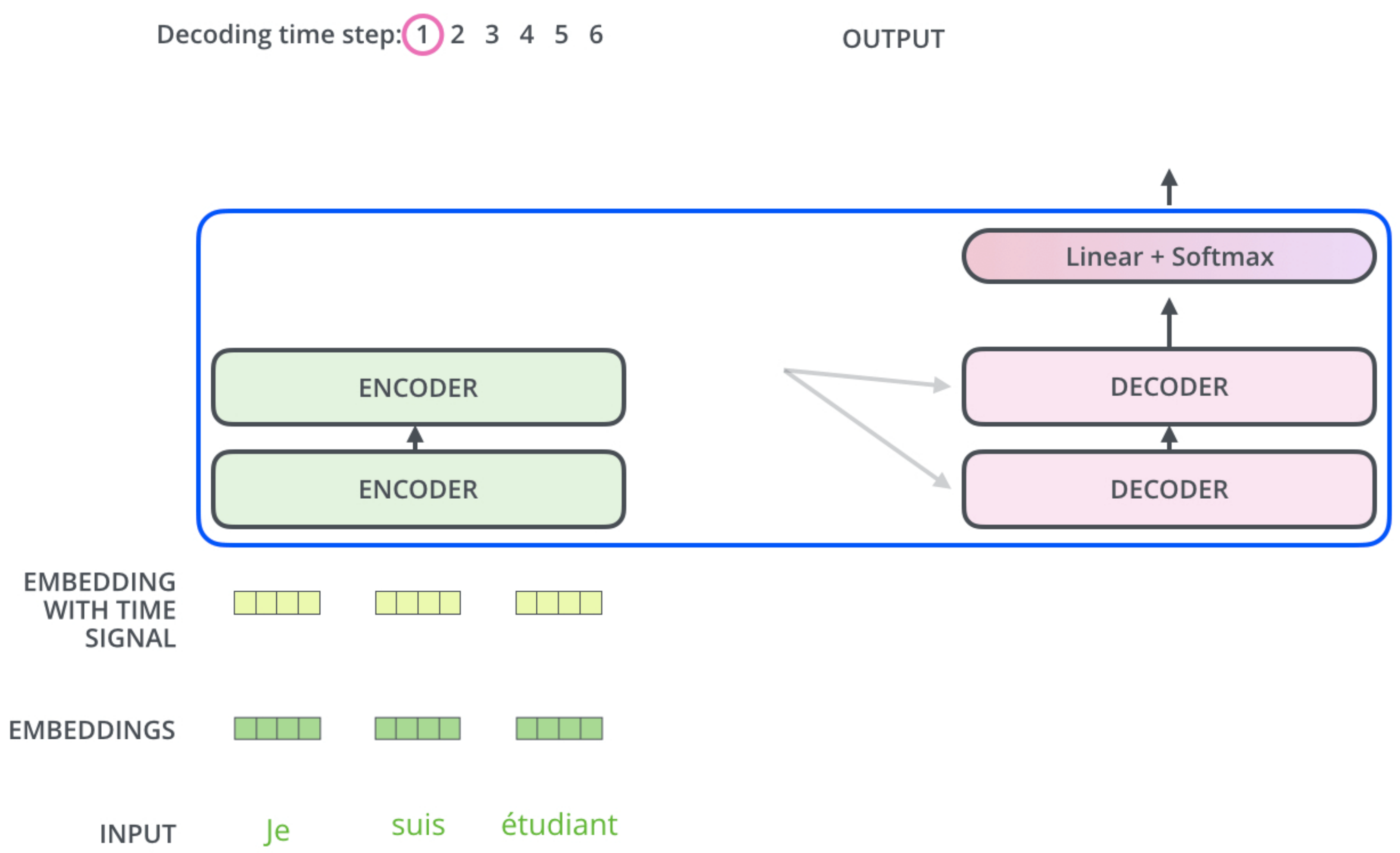
0.11	−inf	−inf	−inf
0.19	0.50	−inf	−inf
0.53	0.98	0.95	−inf
0.81	0.86	0.38	0.90

Softmax
(along rows) →

Scores

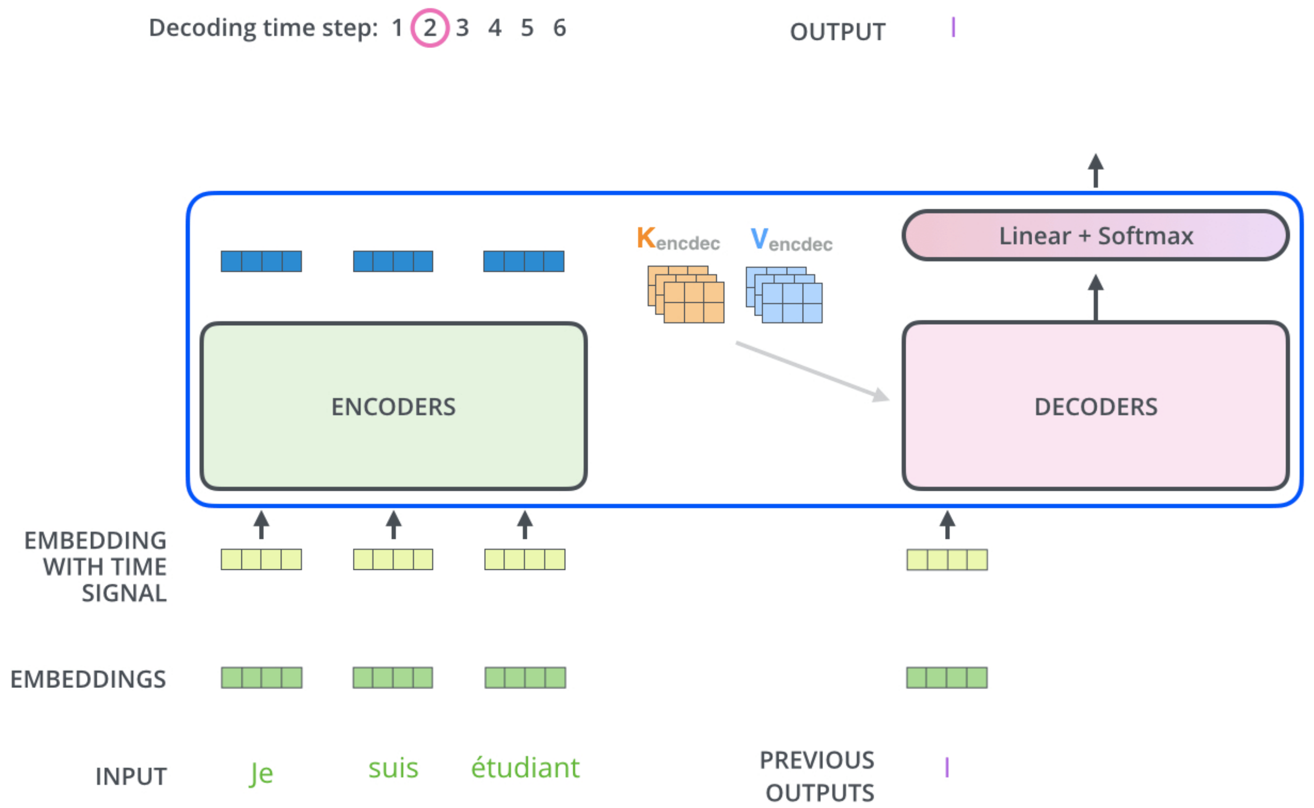
1	0	0	0
0.48	0.52	0	0
0.31	0.35	0.34	0
0.25	0.26	0.23	0.26

The Decoder side



PART 2 Transformer

The Decoder side

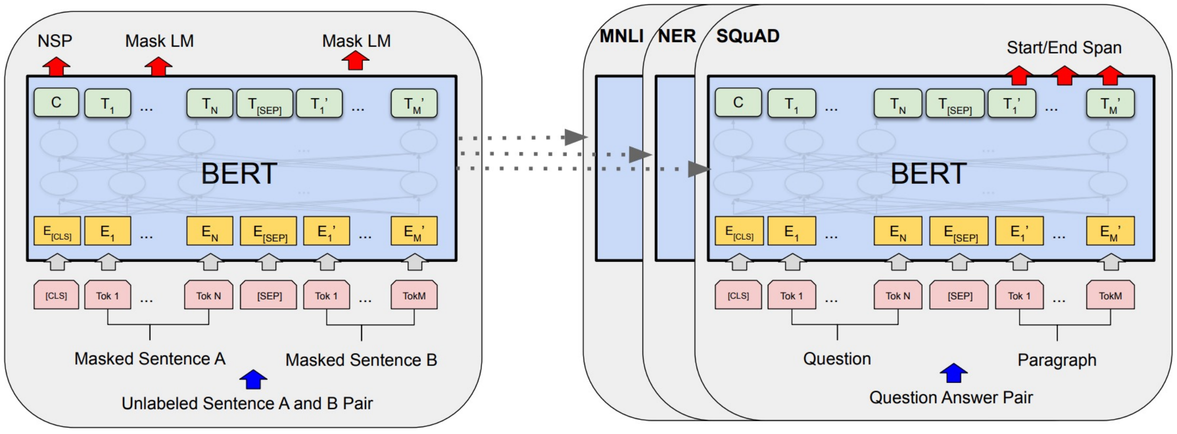


Part 3

BERT

BERT : Bidirectional Encoder Representations from Transformer

- Designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers
  - Masked language Model(MLM) : bidirectional pre-training for language representations
  - Next sentence prediction(NSP)



Pre-training Fine-Tuning

  - Pretrained BERT model can be fine-tuned with just one additional output layer to create SOTA models for a wide range of NLP task(QA, NER, Sentiment Analysis, etc.)

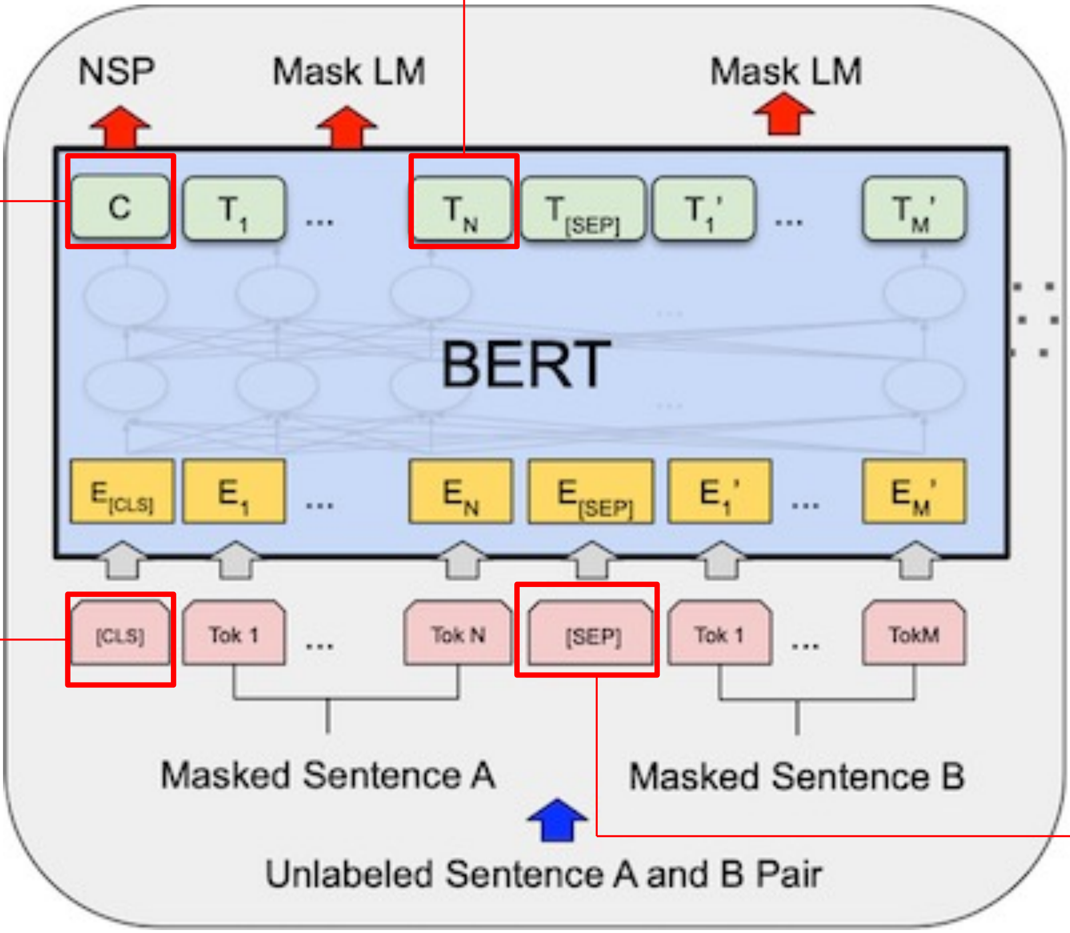
PART 3 BERT

BERT : Bidirectional Encoder Representations from Transformer

- BERT : Input/Output Representations

The first hidden vector of special [CLS] token

The first token of every sequence



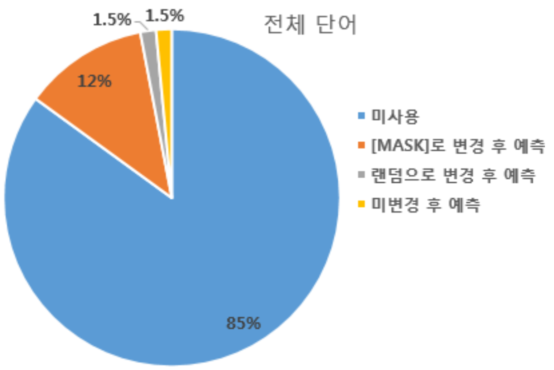
The final hidden vector of N^{th} token

A special token which is differentiate the sentences

PART 3 BERT

BERT : Bidirectional Encoder Representations from Transformer

- Pre-training BERT
  - ✓ Task 1 : Masked Language Model(MLM)
    - (Problem) A mismatch occurs between pre-training and fine-tuning, since the [MASK] token does not appear during fine-tuning
    - (Solution) If the i-th token is chosen, we replace the i-th token with
      1) The [MASK] token 80% of the time
         - The man went to the store -> The man went to the [MASK]
      2) A random token 10% of the time
         - The man went to the store -> The man went to the dog
      3) The unchanged i-th token 10% of the time
         - The man went to the store -> The man went to the store

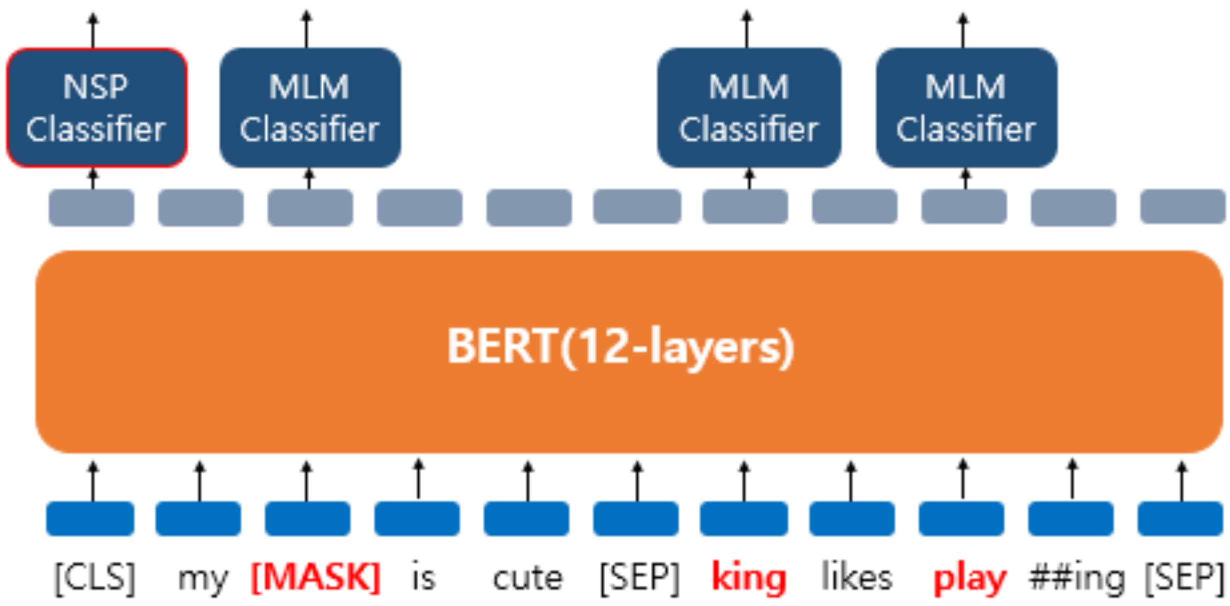


BERT : Bidirectional Encoder Representations from Transformer

- Pre-training BERT
  - ✓ Task 2 : Next Sentence Prediction(NSP)
    - Many important downstream task such as QA and NLI are based on understanding the relationship between two sentences, which is not directly captured by language modeling
    - A binarized next sentence prediction task that can be trivially generated from any monolingual corpus is trained
      - 50% of the time B is the actual next sentence that follows A (IsNext)
      - 50% of the time it is a random sentence from the corpus (NotNext)
      - C(CLS's hidden vector) is used for next sentence prediction
    - Despite its simplicity, pre-training towards this task is very beneficial both QA and NLI

BERT : Bidirectional Encoder Representations from Transformer

Pre-training BERT



PART 3 BERT

BERT : Bidirectional Encoder Representations from Transformer

Fine-training BERT

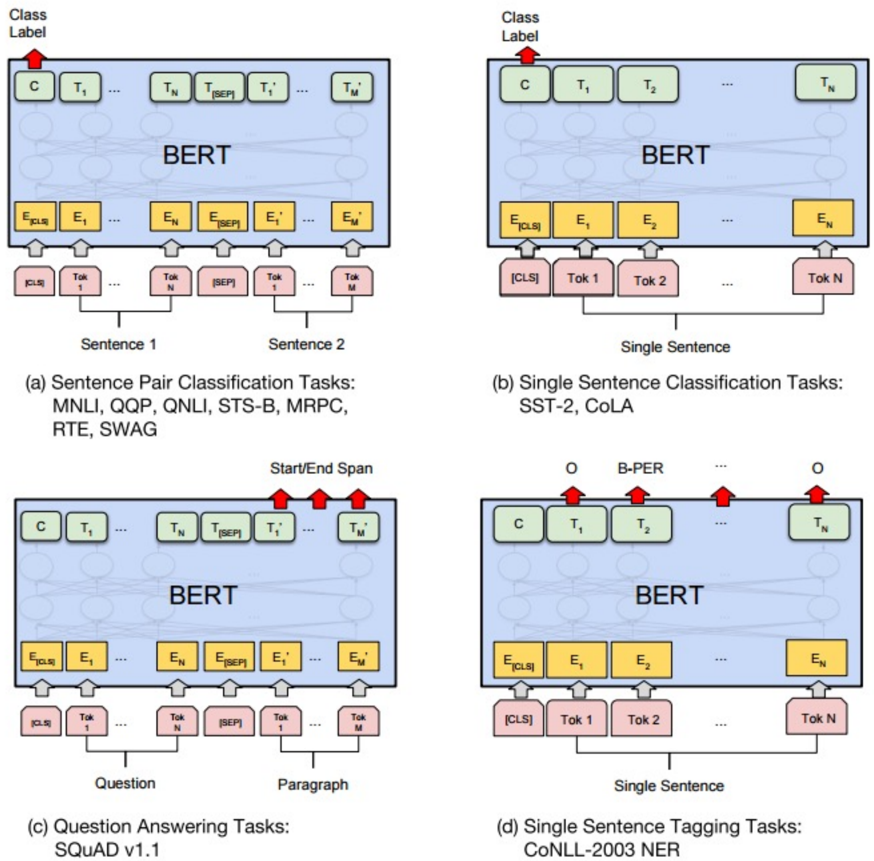


Figure 4: Illustrations of Fine-tuning BERT on Different Tasks.

References

- https://jalammar.github.io/illustrated-transformer/
- https://jalammar.github.io/illustrated-gpt2/
- https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270
- https://wikidocs.net/115055

Thank you..!!!!

Thank you for listening.
Tell us if you have any questions
jwjw9603@g.skku.edu