Wrap-up

Ki Hyun Kim

nlp.with.deep.learning@gmail.com

Transformer

- Attention 연산을 통해 정보의 encoding / decoding을 해결
  - RNN과 달리 위치(순서) 정보를 따로 넣어줘야 함

- 3가지 attention으로 구성
  - Self-attention @ encoder
  - Self-attention with mask @ decoder
  - Attention from encoder @ decoder

- Residual connection으로 깊게 쌓을 수 있음
  - 추후 BERT와 같은 Big LM이 가능하게 됨

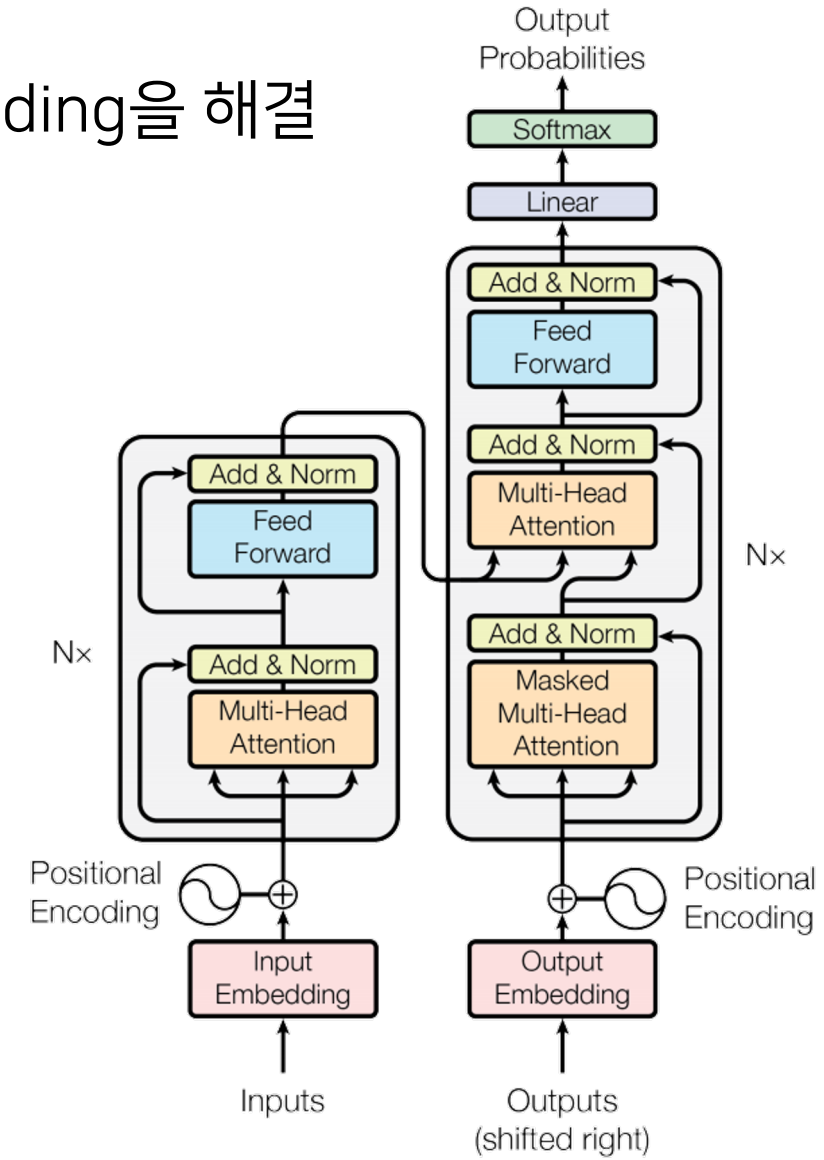
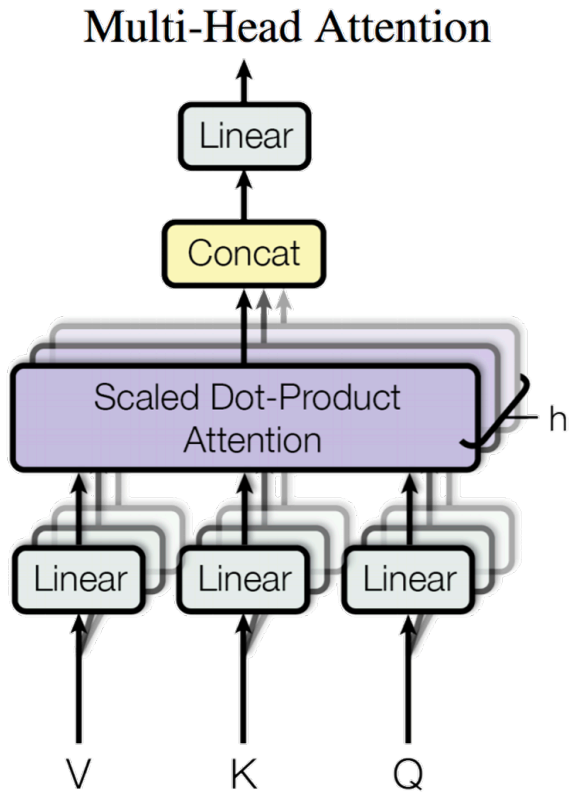


Figure 1: The Transformer - model architecture.

Multi-head Attention

- Attention @ Sequence to Sequence
  - Dot-product 연산은 cosine similarity와 매우 유사함
  - 즉, attention은 query를 잘 만들어내서, key-value로부터 필요한 정보를 얻어내는 과정
  - Decoder의 각 time-step마다 encoder로부터 attention을 통해 정보를 얻어와 생성 토큰의 품질을 높임

- Multi-head Attention @ Transformer
  - 각 head 별로 attention을 수행하여 다양한 정보를 얻어올 수 있음
  - Self-attention을 통해 이전 layer의 정보를 encoding / decoding
  - Attention을 통해 encoder의 정보를 얻어옴



Optimization

- Layer Normalization과 Residual Connection으로 인한 최적화 난이도 증가

- Post-LN Transformer
  - Big Batch-size (over 4k)
  - Noam Decay (Learning rate Warm-up and Linear Decay)
  - Rectified Adam (RAdam)

- Pre-LN Transformer
  - Big Batch-size (over 4k)
  - Adam (+ Learning rate decay)

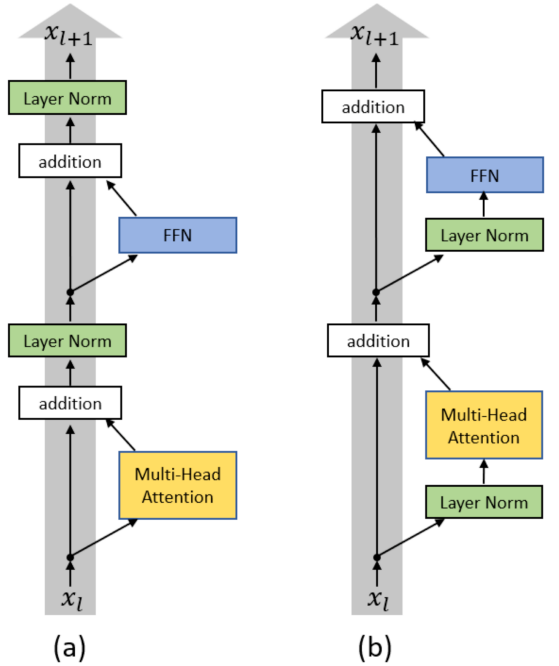


Figure 1: (a) Post-LN Transformer layer; (b) Pre-LN Transformer layer.

- 오픈 소스를 사용할 경우, 어떤 구조인지 확인 후 적절한 최적화 방식 선택 필요

Transformer is Everywhere

- Natural Language Processing
  - Natural Language Generations
  - Natural Language Understandings
- Graph Neural Networks
- Computer Vision

Finally,

- You can write your own NLG code with Transformer.