week 11 review - attention is all you need

1. Introduction
    - seq2seq모델의 문제점
        - 문맥벡터가 encoder의 모든 시퀀스 정보를 포함하고 있어서 decoding 할 때 개별 토근과의 관계 파악이 어렵다. - 병목현상이 발생한다
        - 시퀀스가 길어지면 gradient vanishing problem - 앞부분 토큰의 그라디언트가 희미해진다

    → Attention value를 이용하자! attention 매커니즘에 의존하는 transformer 모델을 제안

2. background
    - CNN : input, output에서 병렬적으로 hidden representation을 계산
    - CNN을 building block으로 사용
    - transformer에서는 계산수가 거리에 따라 증가하는 것이 아니라, 일정한 수의 계산이 요구된다.
    - self-attention(intra-attention) : 한 시퀀스의 representation을 계산하기 위해 서로 다른 위치에 있는 요소들을 관련시키는 매커니즘
    - transformer는 input과 output의 representation을 계산하기 위해 RNN, 합성곱을 사용하지 않고 오직 self-attention에만 의존한다.

3. Model architecture
    - encoder는 input representation x를 z로 매핑한다.
    - decoder는 z를 이용해 output sequence를 형성한다.
    - auto-regressive : 각 타임스텝에서 다음 심볼을 생성할 때 이전에 생성된 심볼을 추가 인풋으로 사용한다.

- self-attention과 point-wise fully connected layer들을 인코더와 디코더에 각각 쌓아올린다.

a. encoder
   - 6개의 동일한 레이어, 각 레이어는 두 개의 sublayer로 구성된다. (multi-hear self-attention 메커니즘, position-wise FC feed-forward 네트워크)
   - 각 sub-layer마다 residual connection, layer normalization
   - 모든 sub-layer들이 512로 동일한 차원을 갖는다.

b. decoder
   - 두 sub-layer 사이에 encoder의 output에 대해 multi-head attention을 수행하는 레이어를 추가
   - masking : decoder의 self-attention layer에서 예측을 진행할 때, 미래의 위치에 접근하는 것을 막고 이전의 위치에만 접근할 수 있도록 하였다.

c. attention

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

- query, key-value 쌍을 output에 매핑하는 것
- Q에 맞는 K를 이용하여 V에 가중치를 주는 기법

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_\text{h})W^O$$
$$\text{where head}_\text{i} = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

- multihead attention
  - Q, K, V를 다르게 projection후 concat하면 다른 representation subspace에서 얻은 정보에 attention이 가능하기 때문에 단일 어텐션보다 유리하다.

computing cost는 비슷하다.

4. Why Self-Attention
    - 각 레이어마다 필요로하는 총 computing cost가 줄어든다
    - 병렬화가 가능한 computation이 늘어난다.
    - 신경망 내에서 long-range dependencies를 잇는 path length가 줄어든다. path length 짧을수록 임의의 위치 간의 의존성을 학습하기 쉬워진다.
    - 모든 위치에 attention을 주어 maximum path length 를 1로 낮춰 의존성 문제를 해결했다. sequential operation들이 줄어들어 computational complexity가 감소했다.
    - attention을 사용하면 해석가능한 모델을 만들 수 있다. 문장 내에서의 관계를 확인할 수 있다.