Attention Mechanism

Attention Mechanism

: machine translation용 인코더-디코더 모델의 성능 향상을 위해 Bahdanau et al. (2014)에서 처음으로 소개된 개념

- 길이가 고정된 인코딩 벡터를 압축하여 사용하면, 디코더에게 주어지는 정보는 제한적일 수 밖에 없는 문제를 해결하고자 함. 특히 입력 문장이 길면 번역의 품질이 떨어짐
- 인코더가 압축시킨 인풋을 그대로 받는 대신, 디코더가 출력 단어를 예측하는 매 시점마다 인풋을 다시 직접 참고하는데 특히 중요한 부분에 더 집중하는 방식으로 진행됨
- 추후 transformer 모델의 핵심 아이디어 기반이 됨
- Bahdanau et al (2014) 논문에서는 **alignment score, weights, context vector**의 계산법을 상세히 다루고 있음

Alignment Model/Alignment scores

먼저 인코더의 은닉 상태 h_i와 이전 시점의 디코더 아웃풋 s_{t-1}으로 스코어 $e_{t,i}$이 계산된다. 이는 입력 시퀀스가 현재 시점 t의 출력값과 얼마나 잘 align 되었는지 나타낸다. 이러한 스코어를 계산하는 alignment model은 feedforward neural network가 계산할 수 있는 함수 a이다.

$$e_{t,i} = a(\mathbf{s}_{t-1}, \mathbf{h}_i)$$

Weights

이전 시점들에서 계산된 alignment scores에 소프트맥스 함수를 적용한 값들을 weight라고 한다.

$$\alpha_{t,i} = \mathrm{softmax}(e_{t,i})$$

Context Vector

매 시점마다 디코더에 주어지는 벡터는 weight $\alpha_{t,i}$에 인코더의 은닉상태 h_i를 가중합 한 것과 같다. 즉, 인코더가 만들어낸 인풋 시퀀스의 representation에 대해 weight를 적용한 것

$$\mathbf{c}_t = \sum_{i=1}^{T} \alpha_{t,i} \mathbf{h}_i$$

Queries, Keys, Values

위 개념에 기반해 일반화된 어텐션 메커니즘을 이해해보자.

- query = t 시점의 디코더 셀에서의 은닉 상태
- keys = 모든 시점의 인코더 셀의 은닉 상태들
- values = 모든 시점의 인코더 셀의 은닉 상태들

데이터베이스에는 keys (k)와 values (v)의 튜플 쌍으로 이루어져있다.

예를 들어 데이터베이스 $\mathcal{D}$에 {(“Zhang”, “Aston”), (“Lipton”, “Zachary”), (“Li”, “Mu”), (“Smola”, “Alex”), (“Hu”, “Rachel”), (“Werness”, “Brent”)}가 존재한다고 가정하면, “Li”에 대한 query를 적용하면 “Mu”를 리턴하고, 해당 튜플이 없는 경우에는 비슷한 튜플인 (”Lipton”, “Zachary”)를 받을 수 있다.

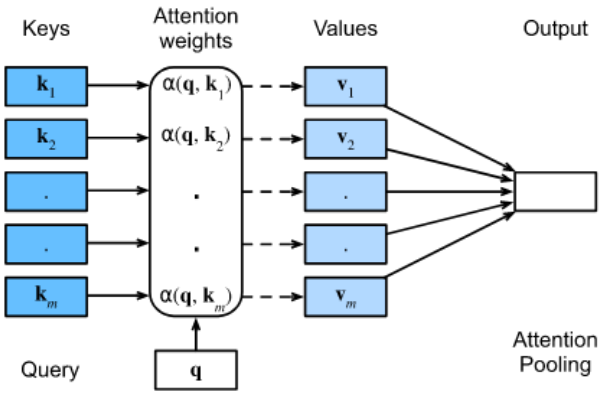
어텐션을 간단하게 함수로 나타내면: **Attention(Q, K, V) = Attention Value**

<수식>

m개의 튜플을 가진 데이터베이스를 $\mathcal{D} \stackrel{\text{def}}{=} \{(\mathbf{k}_1, \mathbf{v}_1), \ldots (\mathbf{k}_m, \mathbf{v}_m)\}$로 정의하고, query를 $\mathbf{q}$라고 하면 $\mathcal{D}$에 대한 attention을 다음과 같이 표현할 수 있다.

$$\text{Attention}(\mathbf{q}, \mathcal{D}) \stackrel{\text{def}}{=} \sum_{i=1}^{m} \alpha(\mathbf{q}, \mathbf{k}_i)\mathbf{v}_i,$$

$\alpha(\mathbf{q}, \mathbf{k}_i)$는 scalar attention weights를 나타내는데 이는 query와 key를 유사도를 구하여 일종의 **스코어**를 계산한 후 **소프트맥스** 함수를 적용한 것이고, α가 클수록 알고리즘이 더 “pay attention”을 한다 (소프트맥스를 적용하는 이유는 모든 weight들의 합이 1이 되어야 하기 때문). 이후 value에 대해 선형 조합을 구한 것이 아웃풋 (attention value)이 된다.

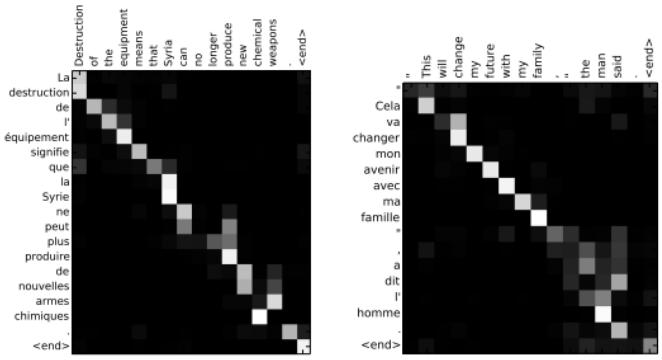


이러한 아웃풋을 만들어내는 과정을 attention pooling이라고 함

이러한 attention pooling은 미분가능하고 gradient가 소실되지 않아 모든 딥러닝 프레임워크에 적용이 가능하다는 장점이 있다.

요약: query에 대해 모든 key와의 유사도 구하기 → 해당 유사도를 키와 매핑된 각각의 value에 반영 → 반영된 값을 모두 더해 attention value 리턴

<시각화>



Bahdanau et al (2014)에서 제시한 RNN search-50로 영어 (y축)에서 불어 (x축)으로 번역한 결과의 샘플. 각 인풋-타겟 단어에 적용된 weight가 1에 가까울수록 흰색으로 표시된다.

Types of Attention Mechanisms

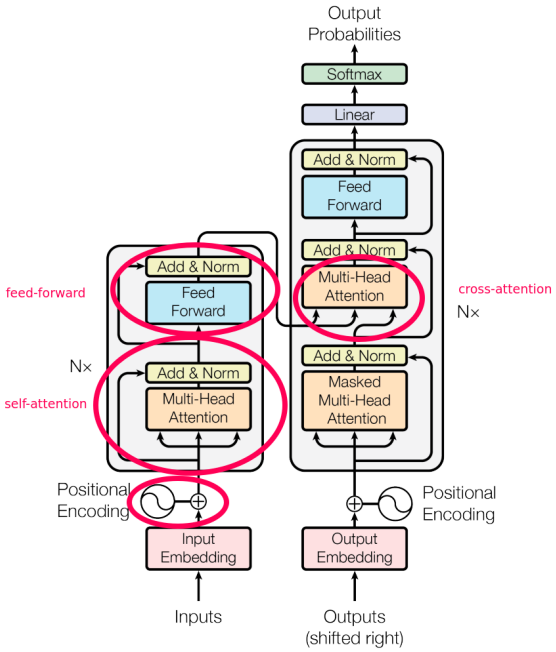
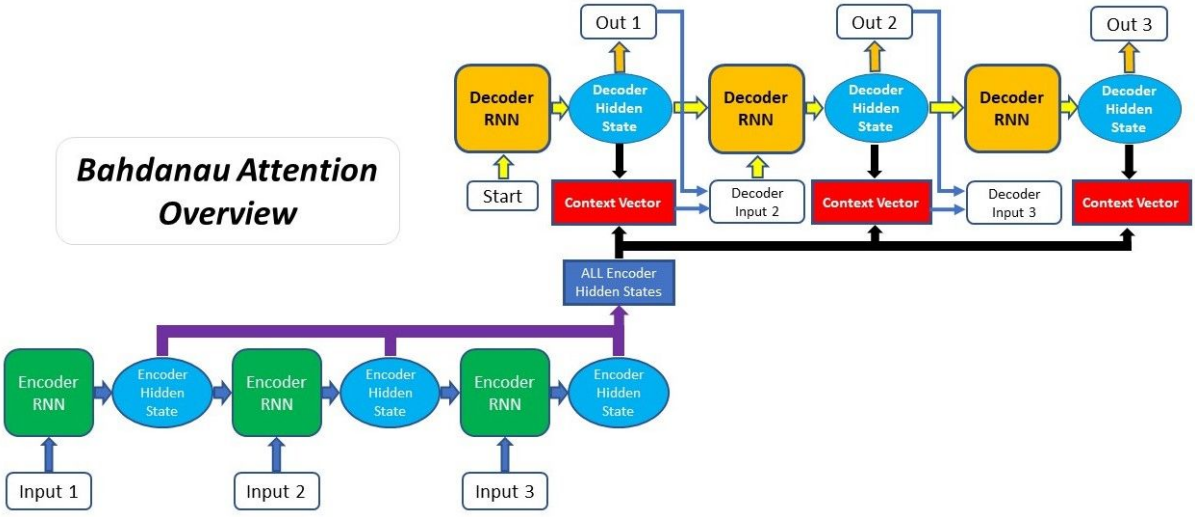
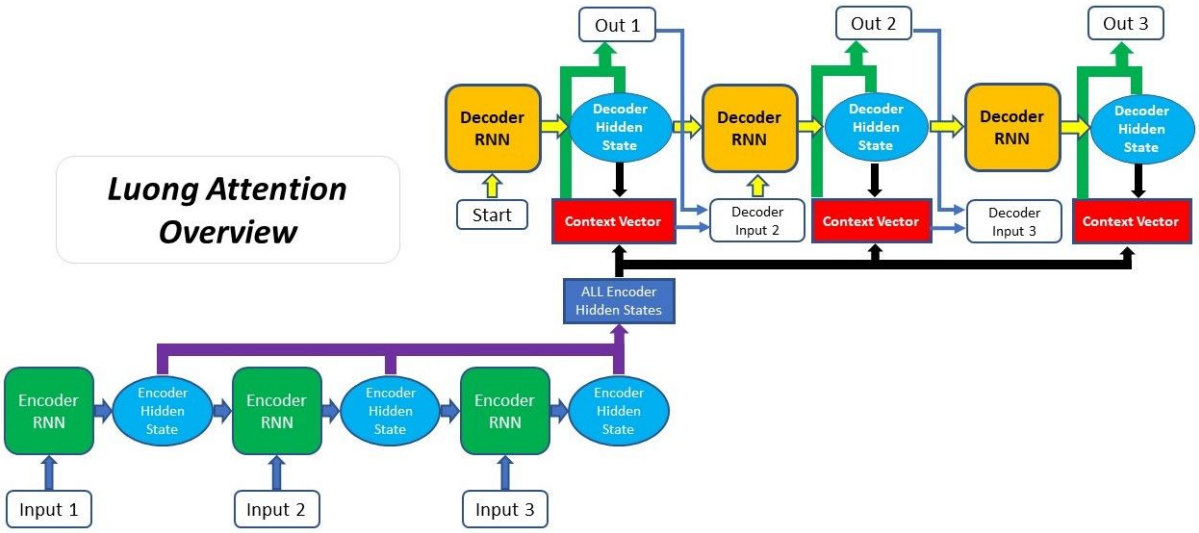


Figure 1: The Transformer - model architecture.

- 광범위한 의미로는 위에 정리한 general attention과 self-attention (transformer의 구성요소) 두가지로 나눌 수 있음





- seq2seq 모델의 어텐션 기법 종류
- https://blog.floydhub.com/attention-mechanism/에 각 어텐션 기법에 대한 자세한 과정과 코드가 나와있음

참고

https://d2l.ai/chapter_attention-mechanisms-and-transformers/index.html

https://machinelearningmastery.com/the-attention-mechanism-from-scratch/

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014).

https://vaclavkosar.com/ml/transformers-self-attention-mechanism-simplified