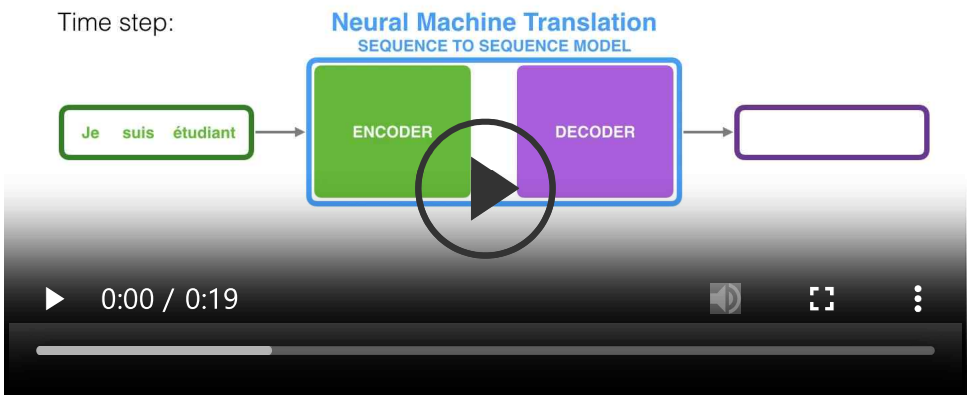
ATTENTION MECHANISM

Attention Is All You Need

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks in an

https://arxiv.org/abs/1706.03762

1. 어텐션(Attention) 매커니즘의 등장



https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

기존 seq2seq 모델 : 인코더에서 입력 시퀀스 → 컨텍스트 벡터(고정크기 벡터) 로 압축

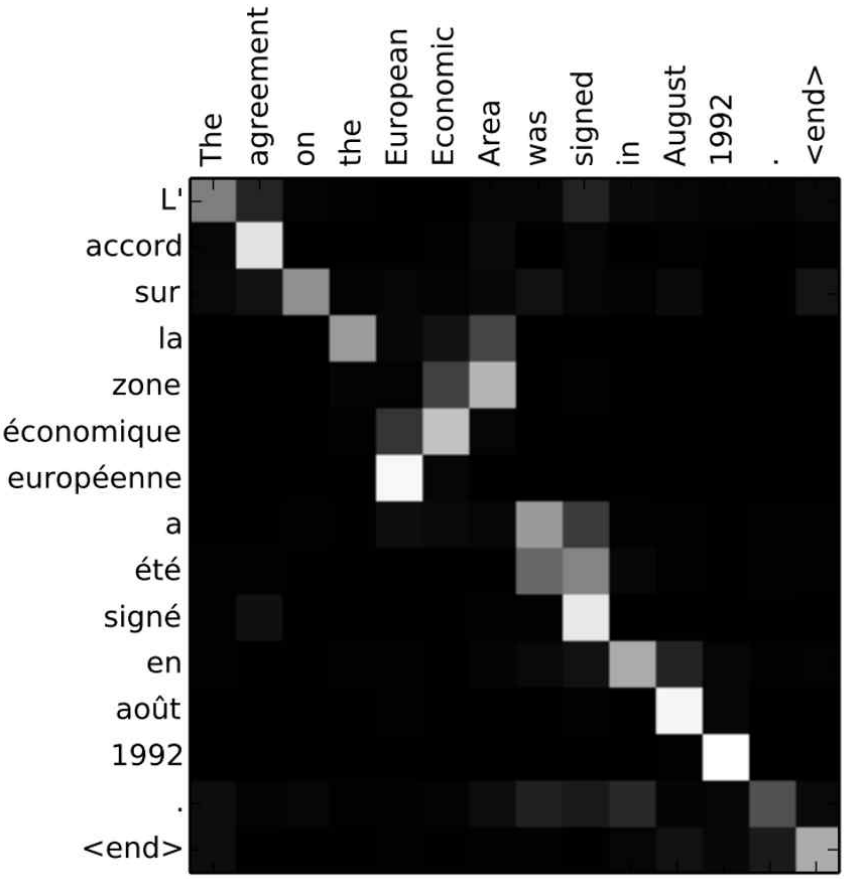
⇒ 문제점 1. 하나의 고정된 크기의 벡터에 모든 정보 압축→ 정보 손실 발생

2. RNN(LSTM)의 고질적인 문제인 기울기 소실(vanishing gradient) 문제 존재

입력문장이 길면 번역 품질이 떨어짐

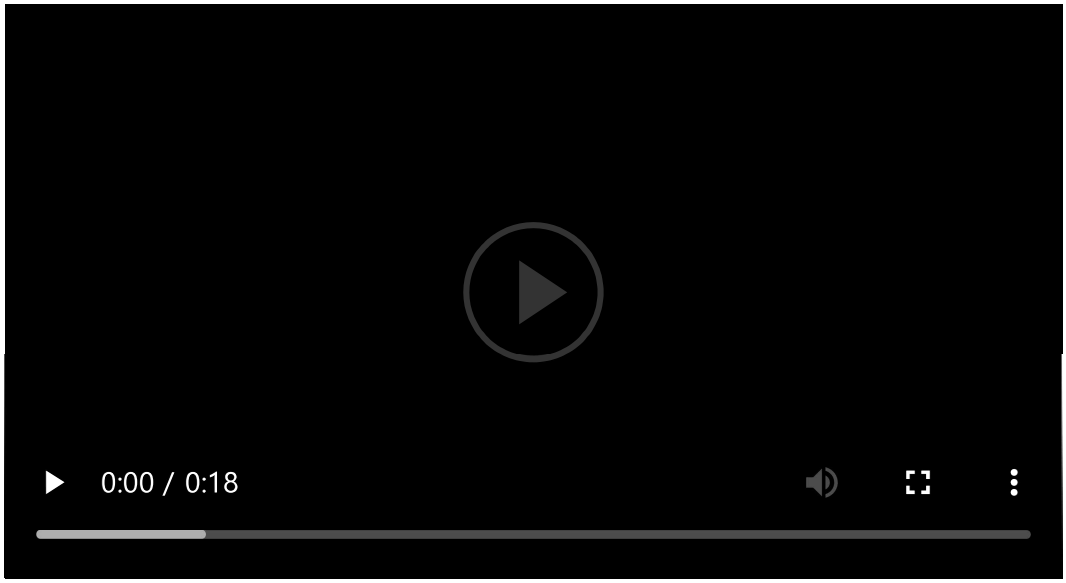
⇒ 해당 시점에서 중요한 단어에만 집중해서 Decoder에 전달하는 게 좋지 않을까?

2. 어텐션(Attention) 매커니즘의 작동 방식



franch → english 번역하는 attention을 사용한 sequence modeling에서의 correlation matrix

attention의 기본 아이디어 : 디코더에서 출력 단어를 예측하는 매 step마다, 인코더의 입력 시퀀스를 다시 참고(동일 비중이 아닌 **예측 단어와 관련 있는 입력 단어**부분을 집중해서 봄)



https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

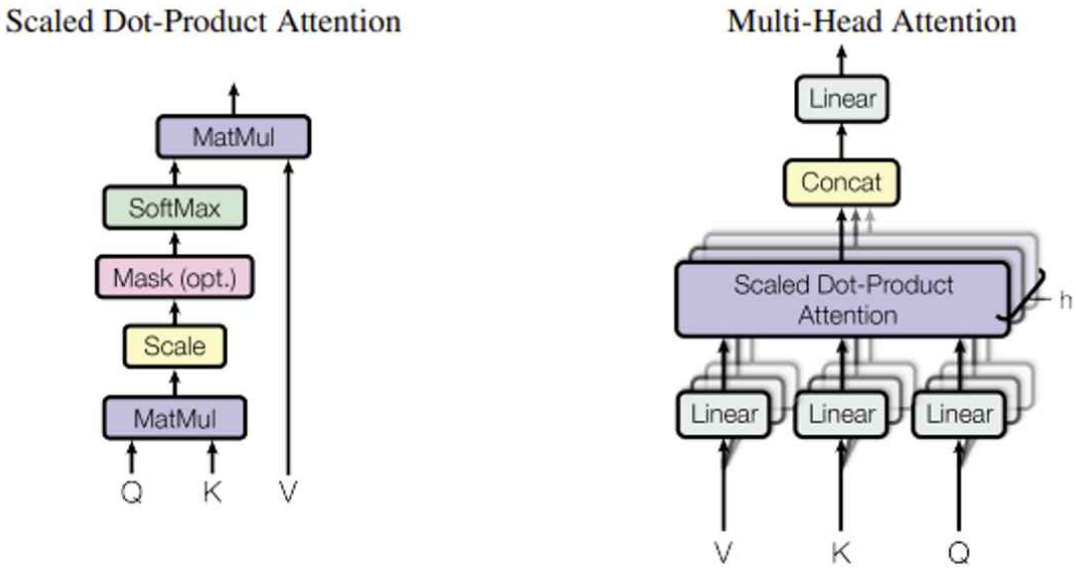
ATTENTION Decoder에 전달할 Context Vector 만들기

1. 인코더의 hidden state(h1,h2,h3,h4)들을 step별로 구함
2. 각각 step의 hidden state에 이전 step 디코더의 hidden state를 각각 dot-product하거나 다른 스코어 함수를 사용해 점수 부여 (=Attention Score)
3. 점수를 softmax(점수 합이 1) → 단어가 모든 단어들과 어느정도 correlation이 있는지 확인
4. Softmax된 점수에 해당하는 각각의 hidden state를 곱해줌
5. 점수에 곱해진 Vector 들을 더함 ⇒ Context Vector 생성

⇒ 디코더에서 출력 단어를 예측하는 매 시점(time step)마다, 인코더의 전체 입력 문장을 참고(해당 시점에서 예측해야 할 단어와 연관 있는 단어에 집중해서)

2. 멀티 헤드 어텐션(Multi-Head Attention)

하나의 attention function을 사용하는 것보다 여러 개의 attention function을 만드는 것이 더 효율적 (CNN에서 여러 개의 필터를 사용하는 것과 유사)

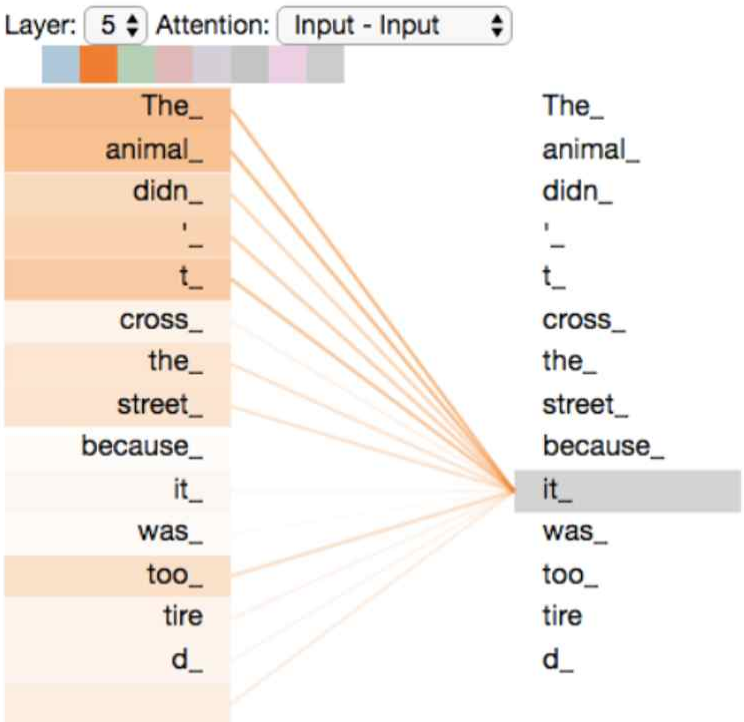


3. 셀프 어텐션(Self - Attention)

특정 문장에서 자기 자신의 문장 스스로에게 attention을 수행해 학습

> *ex. A dog ate the food because **it** was hungry.*

→ 이 문장에서 id은 dog인가? food인가? ⇒ self attention을 이용해 알 수 있음



https://velog.io/@tobigs-nlp/Attention-is-All-You-Need-Transformer - 문장의 self attention 결과

self attention에서 각 단어들의 표현은 문장 안에 있는 다른 단어의 표현과 연결이 되어 단어가 문장 내에서 갖는 의미를 이해함

⇒ position 상 멀리 떨어져 있는 단어(long range dependencies)라도 잘 학습하기 위해서 사용

4. 트랜스포머의 구조(The Transformer - Model Architecture)

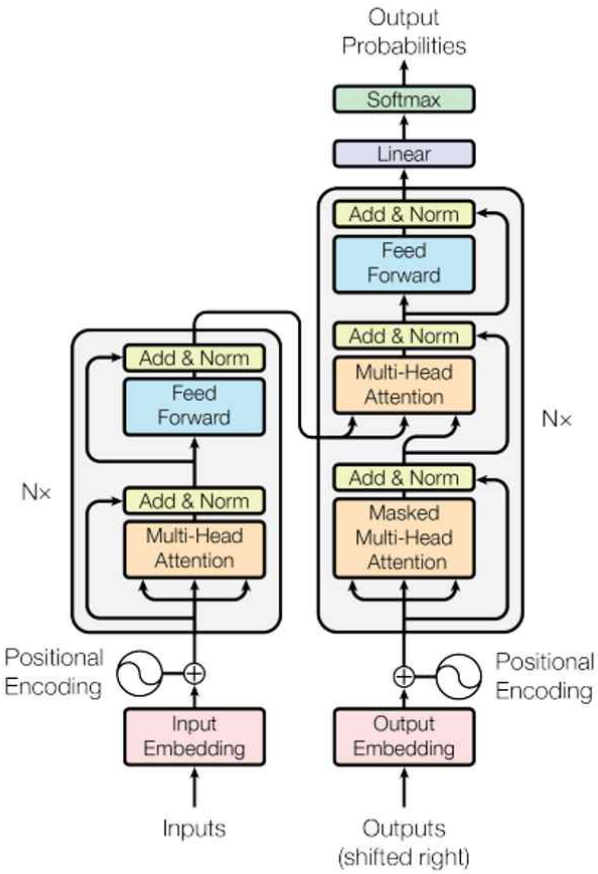
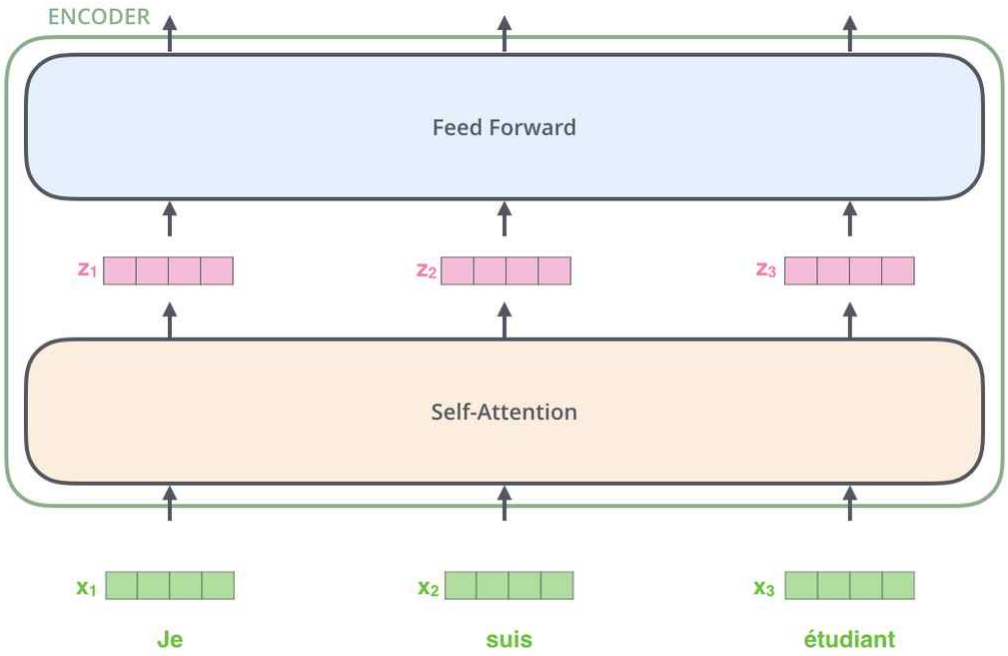


Figure 1: The Transformer - model architecture.



https://jalammar.github.io/illustrated-transformer/

self attention과 position-wise feed forward network 구조를 합한 것과 같음

인코딩 된 단어 임베딩으로 들어옴

→ 1. self attention에 의해 attention이 가해짐

→ 2. 정규화 수행

→ 3. feed forward 과정

→ 4. 정규화 수헹

반복

디코더 : 인코더의 결과값을 받아 단어 예측 시작