

Sequence-to-Sequence model

박진우 이영신 유건희
조상현 유재형 하은겸

발표 : 하은겸, 조상현

- 
- 
1. Sequence-to-Sequence
 2. Transformer model

What is the best way for translation? (English to Korean)

I love you = Nan nul saranghey
난 널 사랑해

WORD TO WORD TRANSLATION?

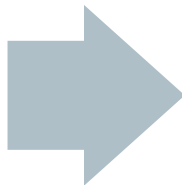
INPUT

I

Love

You

I love you



PREDICTION

nan(난)

saranghey(사랑해)

nul(널)

nan saranghey nul (난 사랑해 널)

nan nul saranghey (난 널 사랑해)

word to word translation의 문제점

1. 영어와 한국어의 문법 순서가 다르다.
2. 단어별로 번역시 결과의 단어수가 input의 단어 수와 동일할 확률이 크다.

How are you? = **Jal jiney?** (잘 지내)
3 words 2 words

Sequence to Sequence Learning

- Sequence to sequence model

- ▲ 기계 번역, 문서 요약, 그리고 이미지 캡셔닝 등의 문제에 큰 성공을 거둔 딥러닝 모델
- ▲ 최근 10년 동안의 자연어 처리 연구 중에 가장 영향력이 컸던 3가지 모델 중 한가지로 모델로 선정 됨.
- ▲ 구글 번역기도 2016년 말부터 이 모델을 실제 서비스에 이용하고 있음.

Sequence to Sequence Learning

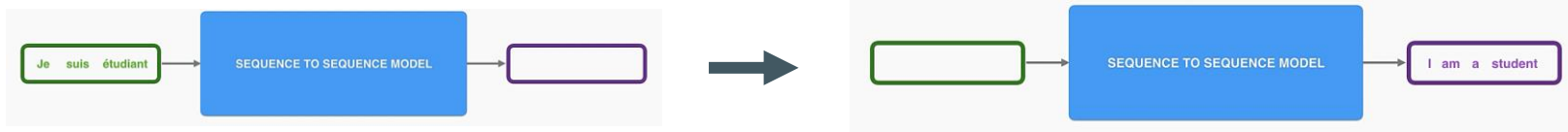
- Sequence to sequence model

- ▲ Seq2seq 모델은 글자, 단어, 이미지의 feature 등의 아이템 시퀀스를 입력받아 또 다른 아이템 시퀀스를 출력함.

- Trained model



- 신경망 기계 번역

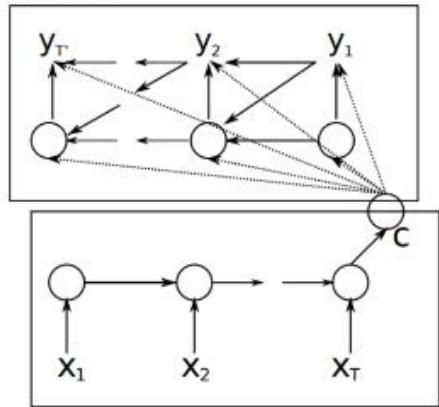


- Input : 일련의 단어로 이루어진 sequence

- Output : 비슷한 형태의 그러나 다른 언어로의 단어 sequence

Sequence to Sequence Learning

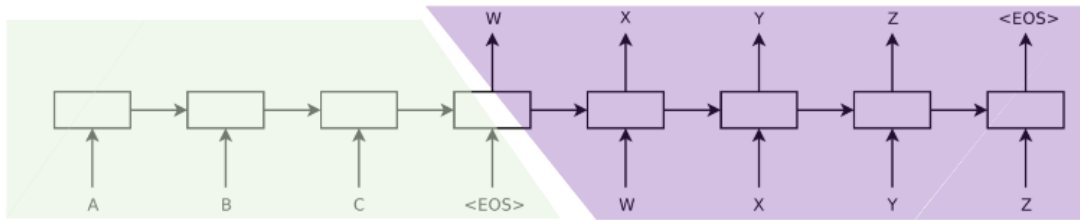
Decoder



Encoder

Cho et al., 2014

encoder



Sutskever et al., 2014

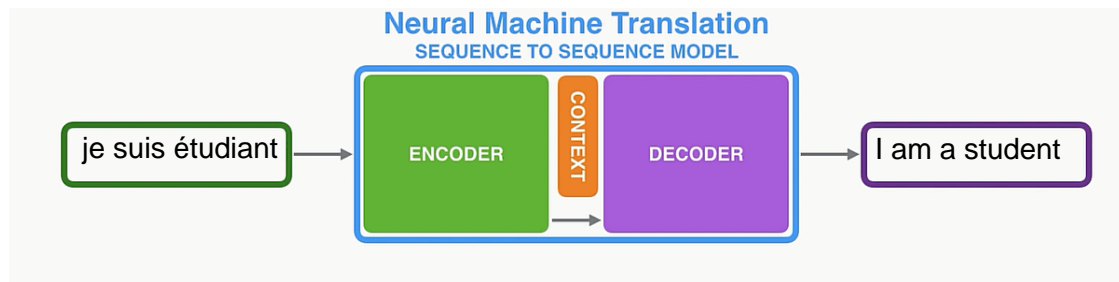
Seq2seq 모델은 하나의 encoder와 하나의 decoder의 구조로 이루어져 있음.

Sequence to Sequence Learning

- Encoder - Decoder

- ✓ Encoder는 입력의 각 아이템을 처리하여 정보를 추출한 후 그것을 하나의 벡터(context)로 만들어 냄.
- ✓ 입력의 모든 단어에 대한 처리가 끝난 후 encoder는 context를 decoder에게 보내고 출력할 아이템이 하나씩 선택되기 시작함.

- 신경망 기계 번역



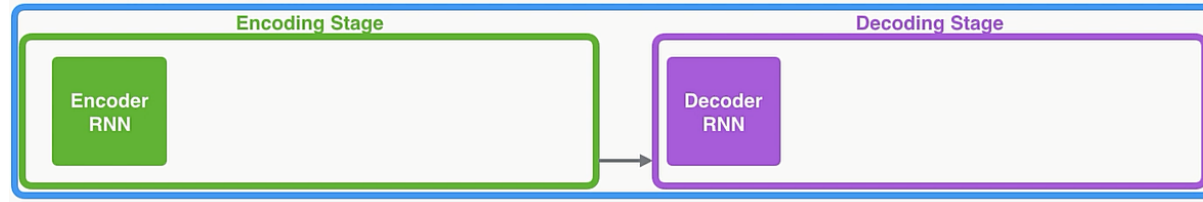
- context가 하나의 벡터 형태로 전달
- Encoder, decoder 둘다 recurrent neural networks(RNN)를 이용하는 경우가 많음.

Sequence to Sequence Learning

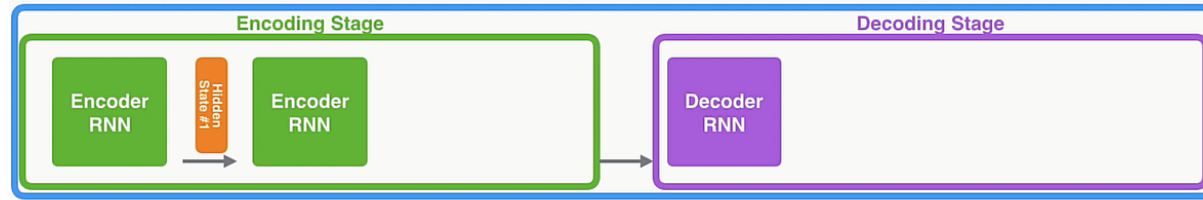
- ✓ encoder 와 decoder 는 모두 RNN이며, RNN은 한번 아이템을 처리할 때마다 새로 들어온 아이템을 이용해 그의 hidden state를 업데이트 함. 이 hidden state는 그에 따라 encoder가 보는 입력 시퀀스에 대한 정보를 담게 됨.
- ✓ 마지막 단어의 hidden state는 우리가 decoder에게 넘겨주는 context임.
- ✓ 가장 마지막 단어가 반환되면 중단이 됨.

Neural Machine Translation

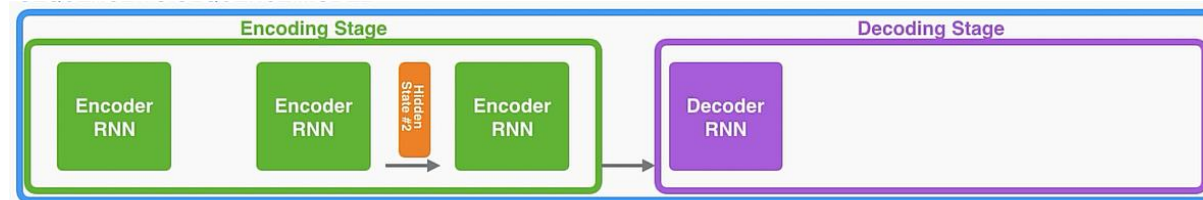
SEQUENCE TO SEQUENCE MODEL



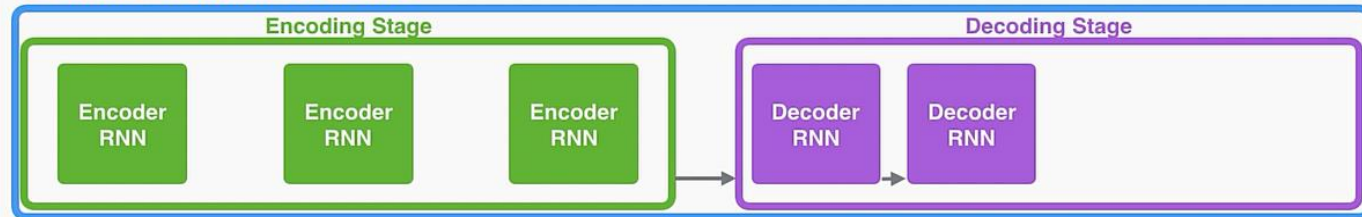
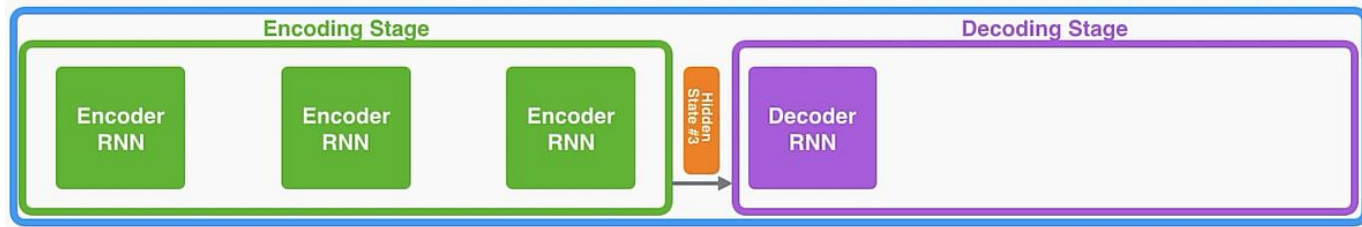
je



suis



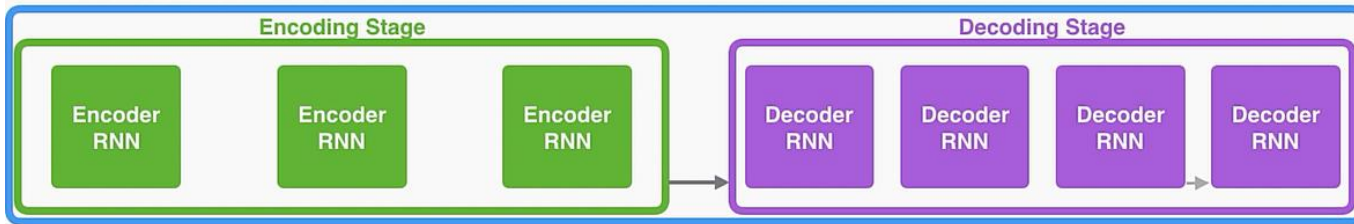
étudiant



I am a student

Neural Machine Translation

SEQUENCE TO SEQUENCE MODEL



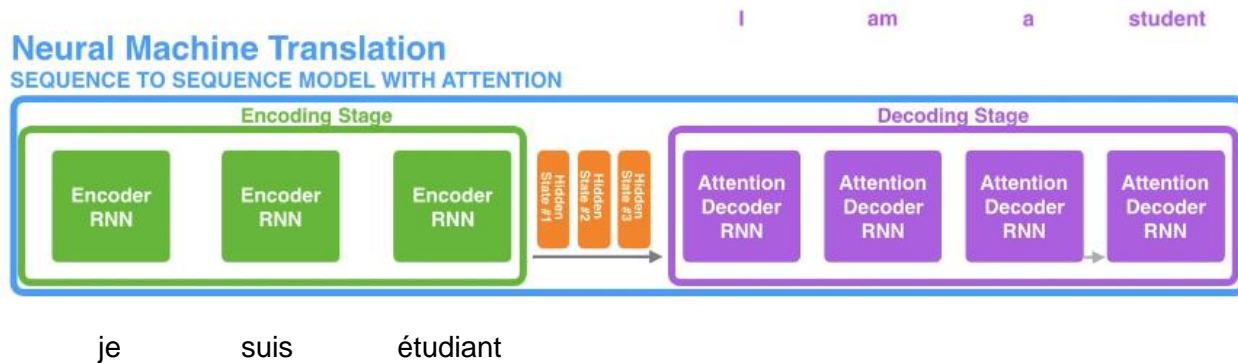
Attention in Seq2Seq Learning

- ✓ 이전의 Seq2seq 모델은 하나의 고정된 벡터로 전체의 맥락을 나타내는 방법은 특히 긴 문장들을 처리하는 것이 어려움.
- ✓ 이에 대한 해결책 : 'Attention'
- ✓ Attention 매커니즘은 seq2seq모델이 디코딩 과정에서 현재 스텝에서 가장 관련된 입력 파트에 집중할 수 있도록 해 줌.
- ✓ 기계 번역의 품질을 매우 향상 시킴.

Attention in Seq2Seq Learning

- attention 모델과 기존의 seq2seq 모델의 차이점(2가지)

1. encoder 가 decoder에게 넘겨주는 데이터의 양이 attention 모델에서 훨씬 더 많음.
 - 기존 seq2seq 모델에서는 그저 마지막 아이템의 hidden state 벡터를 넘겼던 반면 attention 모델에서는 모든 스텝의 hidden states를 decoder에게 넘겨줌.



Attention in Seq2Seq Learning

Example)

Time step: 7

Neural Machine Translation SEQUENCE TO SEQUENCE MODEL WITH ATTENTION



- Step 7에서 attention 매커니즘은 영어 번역을 생성하려 할 때 decode가 "étudiant" ("학생"을 의미하는 불어)에 집중하게 함.
- 이렇게 스텝마다 관련된 부분에 더 집중할 수 있게 해주는 attention model 은 attention이 없는 모델보다 훨씬 더 좋은 결과를 생성함.

Attention in Seq2Seq Learning

- attention 모델과 기존의 seq2seq 모델의 차이점(2가지)
 2. attention 모델의 decoder가 출력을 생성할 때에는 하나의 추가 과정이 필요

Step 1) encoder 에서 받은 전체 hidden states를 봄. -- 각 스텝에서의 encoder hidden states는 이전의 맥락에 대한 정보도 포함하고 있지만 그 스텝에서의 입력 단어와 가장 관련이 있음.

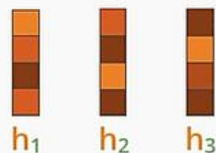
Step 2) 각 스텝의 hidden state마다 점수를 매김.

Step 3) 매겨진 점수들에 softmax를 취하고 이것을 각 타임 스텝의 hidden states에 곱해서 더함. 이를 통해 높은 점수를 가진 hidden states는 더 큰 부분을 차지하게 되고 낮은 점수를 가진 hidden states는 작은 부분을 가져가게 됨.

‘이러한 점수를 매기는 과정은 decoder가 단어를 생성하는 매 스텝마다 반복’

Attention in Seq2Seq Learning

1. Prepare inputs



Encoder
hidden
states



Decoder hidden
state at time step 4

2. Score each hidden state

13	9	9
----	---	---

scores

Attention weights for
decoder time step #4

3. Softmax the scores

0.96	0.02	0.02
------	------	------

softmax scores

4. Multiply each vector by
its softmaxed score



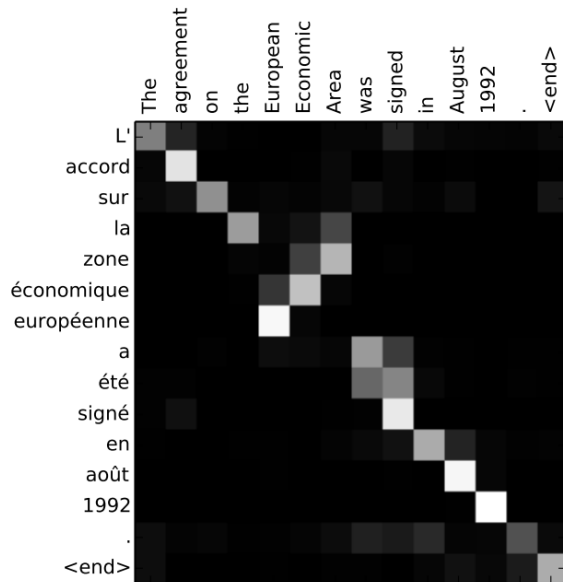
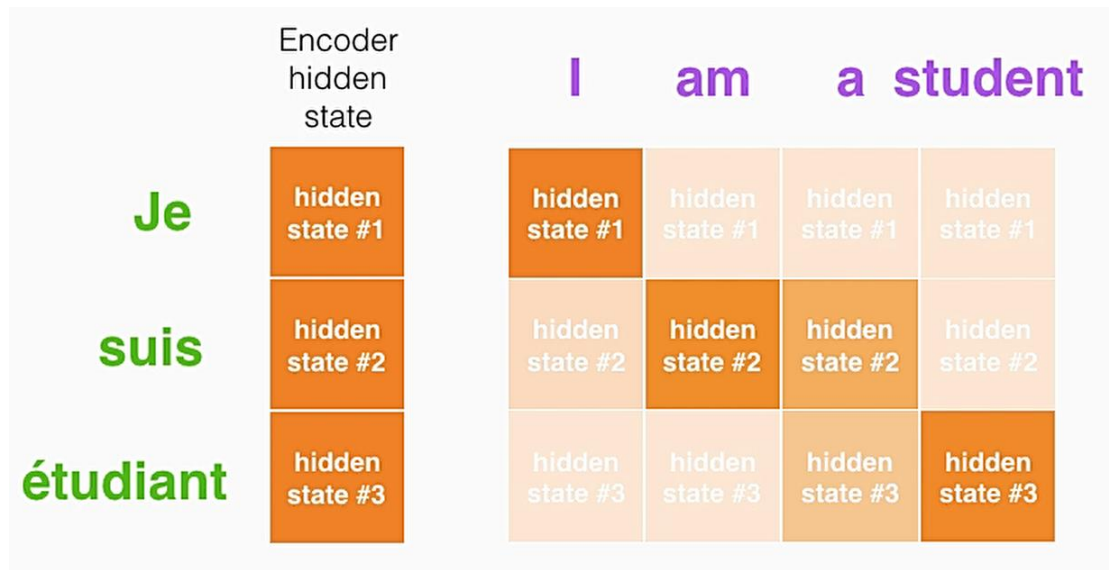
=

5. Sum up the weighted
vectors



Context vector for
decoder time step #4

Attention in Seq2Seq Learning



attention 을 이용하면 각 decoding 스텝에서 입력 문장에서 어떤 부분을 집중하고 있는지에 대해 볼 수 있음.


References

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

<https://arxiv.org/pdf/1409.0473.pdf>



Transformer model

- Attention is all you need
- 

Why Transformer?

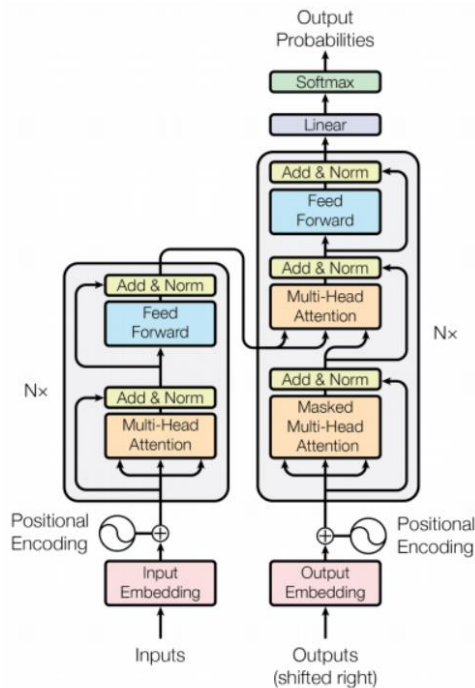
RNN에 기반한 seq2seq 모델의 문제

: 하나의 고정된 크기의 벡터에 모든 정보를 압축해 사용하기 때문에, 정보의 손실이 발생

: 기울기 소실 (Vanishing Gradient) 문제가 발생해 입력 문장이 길어지면 품질이 떨어지는 현상이 발생한다.

: 토큰 하나씩 처리해야 하기 때문에, 속도가 느리다.

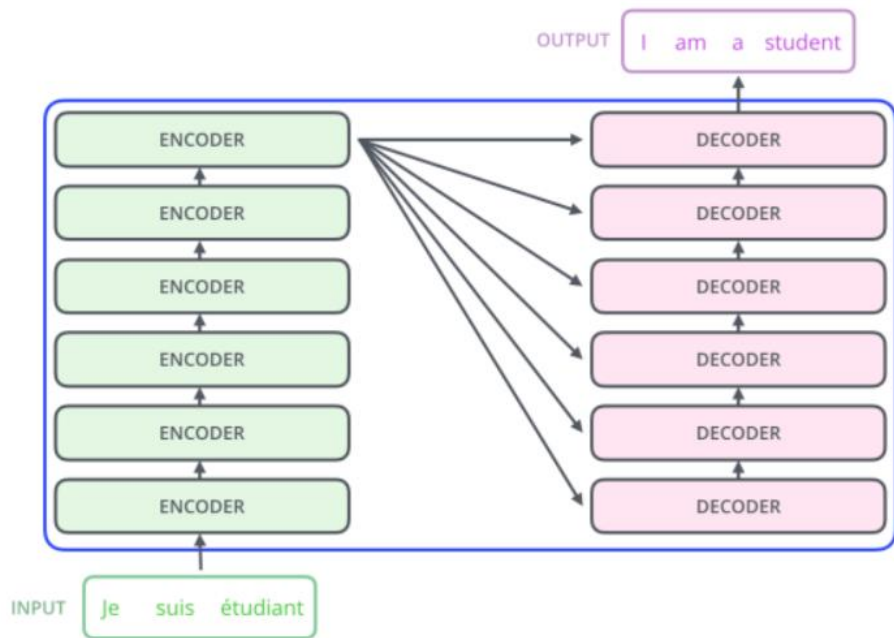
Transformer



- RNN을 대신 Attention으로 구현한 모델
 - Scaled Dot-Product Attention
 - Multi-Head Attention
- ➔ 어느 정도 이상의 문맥을 이어갈 수 있다!!
- 병렬처리로 학습 속도가 빠르다.
- N개의 인코더, 디코더 구조

Transformer 구조

Jay Alammar(Transformer)



- 디코더에서 출력 단어를 예측하는 매 시점마다, 인코더에서의 전체 입력 문장을 다시 한번 참조
- 연관성이 있는 입력 단어 부분에 더 집중

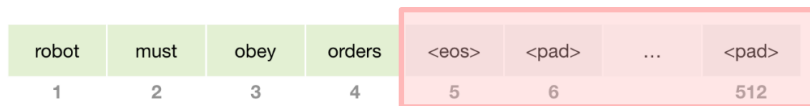
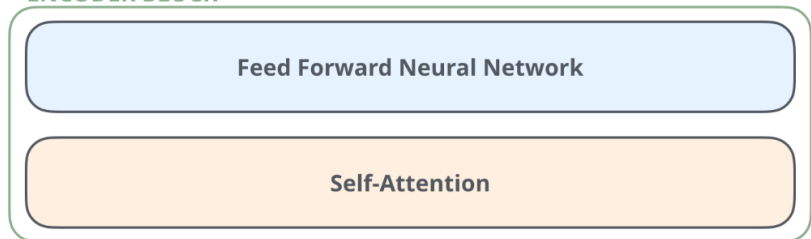
Encoding block vs Decoding block

Unmasked vs. Masked



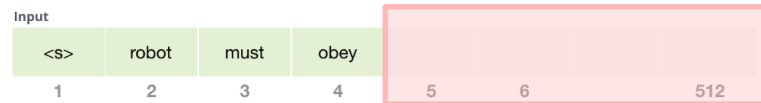
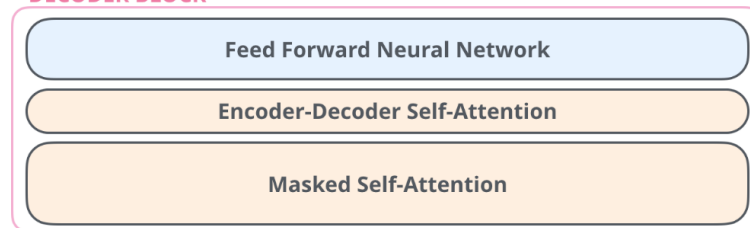
THE TRANSFORMER

ENCODER BLOCK

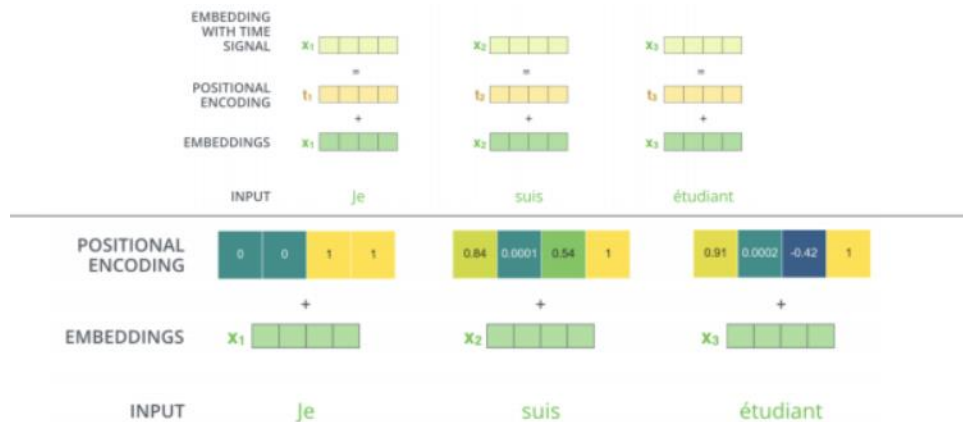
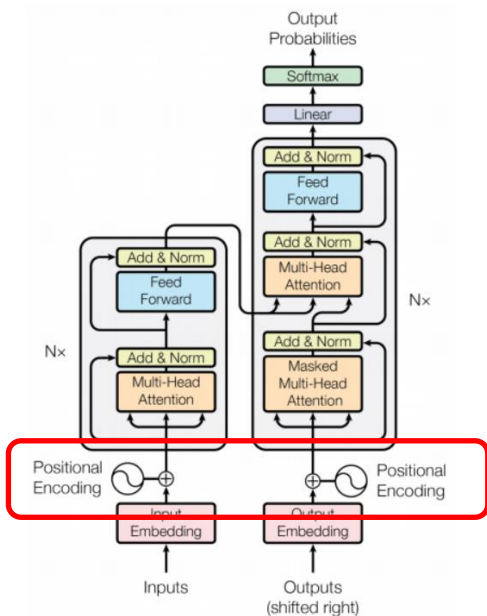


THE TRANSFORMER

DECODER BLOCK

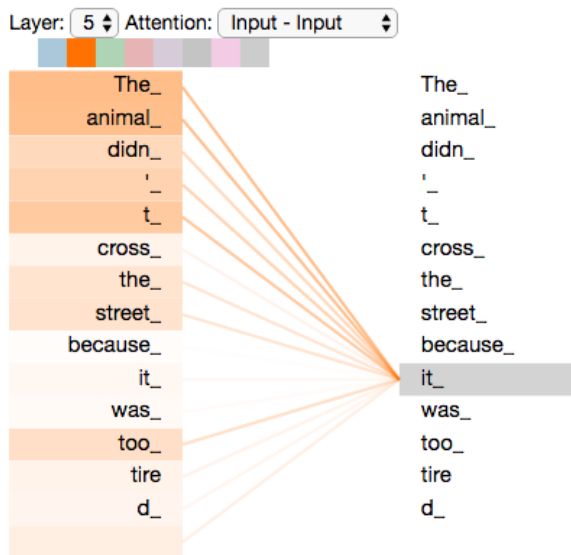


Positional encoding



- 입력 문장의 순서를 고려해주기 위한 방법
- 각 Embedding에 더해주는 Vector

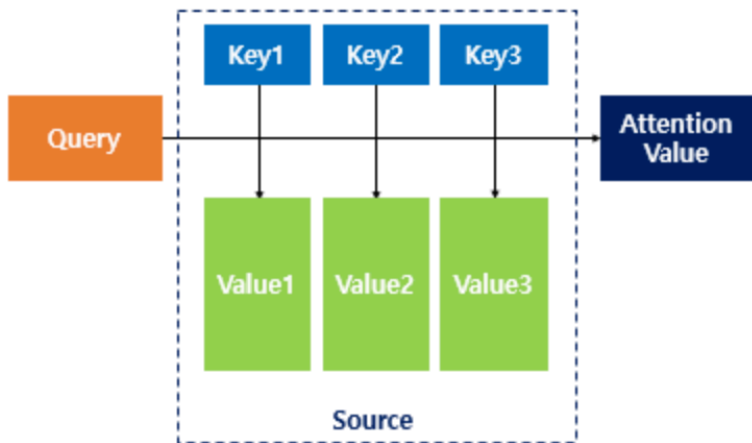
Attention: Self-Attention



- Self-Attention을 통해 다른 위치의 토큰들을 살펴볼 수 있다.
- 각 단어의 Encoding을 개선할 수 있는 단서를 찾을 수 있다.
- 어텐션 스코어: 다른 단어들과의 관계값
- 어텐션 맵: 어텐션 스코어 값을 하나의 테이블로 표현

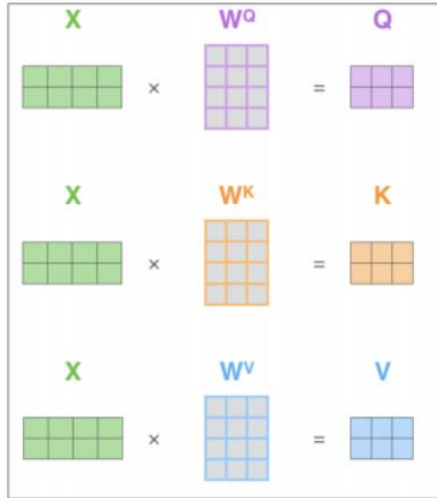
https://colab.research.google.com/github/tensorflow/tensor2tensor/blob/master/tensor2tensor/notebooks/hello_t2t.ipynb#scrollTo=QJKU36QAfqOC

Attention Mechanism



- Query: 다른 모든 단어에 대해 점수를 줄 때 사용되는 **현재 단어의 표현**
 - Key: Segment의 모든 단어에 대한 **Label**
 - Value: **실제 단어 표현**
- Query와 Key를 통해서 적절한 value를 찾아서 연산하겠다!
- $\text{Attention}(Q, K, V) = \text{Attention Value}$

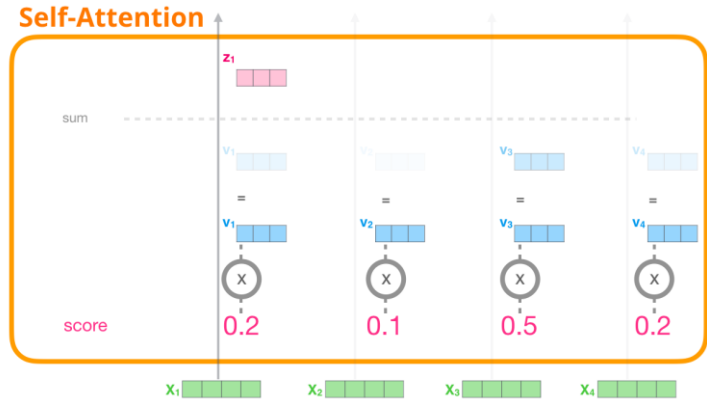
Attention: Self-Attention



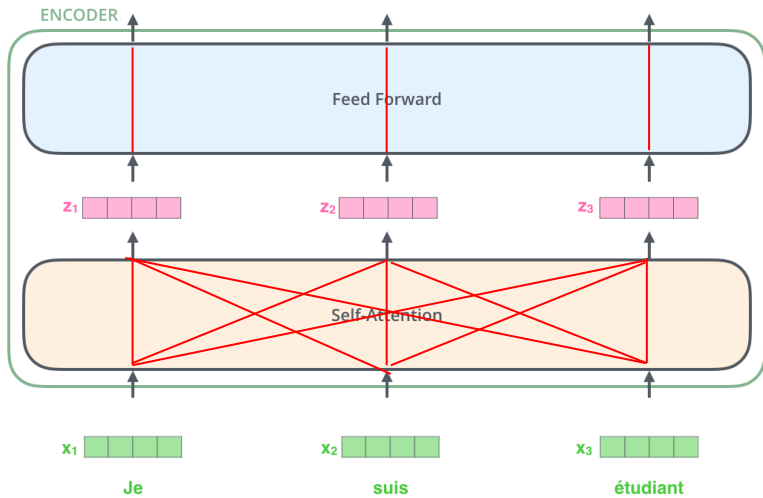
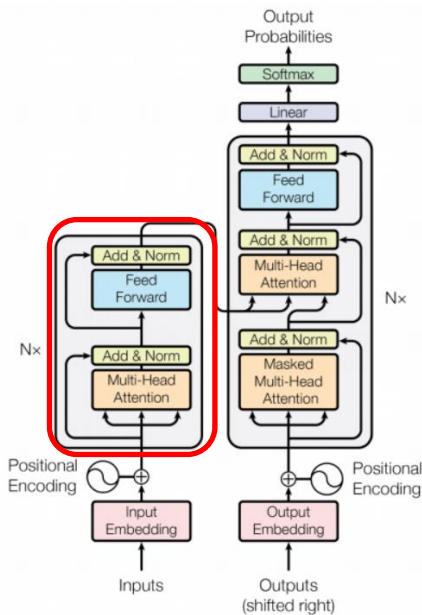
$$\text{softmax}\left(\frac{Q \times K^T}{\sqrt{d_k}}\right) = Z$$

Diagram illustrating the calculation of the attention weights (Z) using the query matrix (Q) and the key matrix (K^T).

3) Multiply the **value vectors** by the **scores**, then sum up

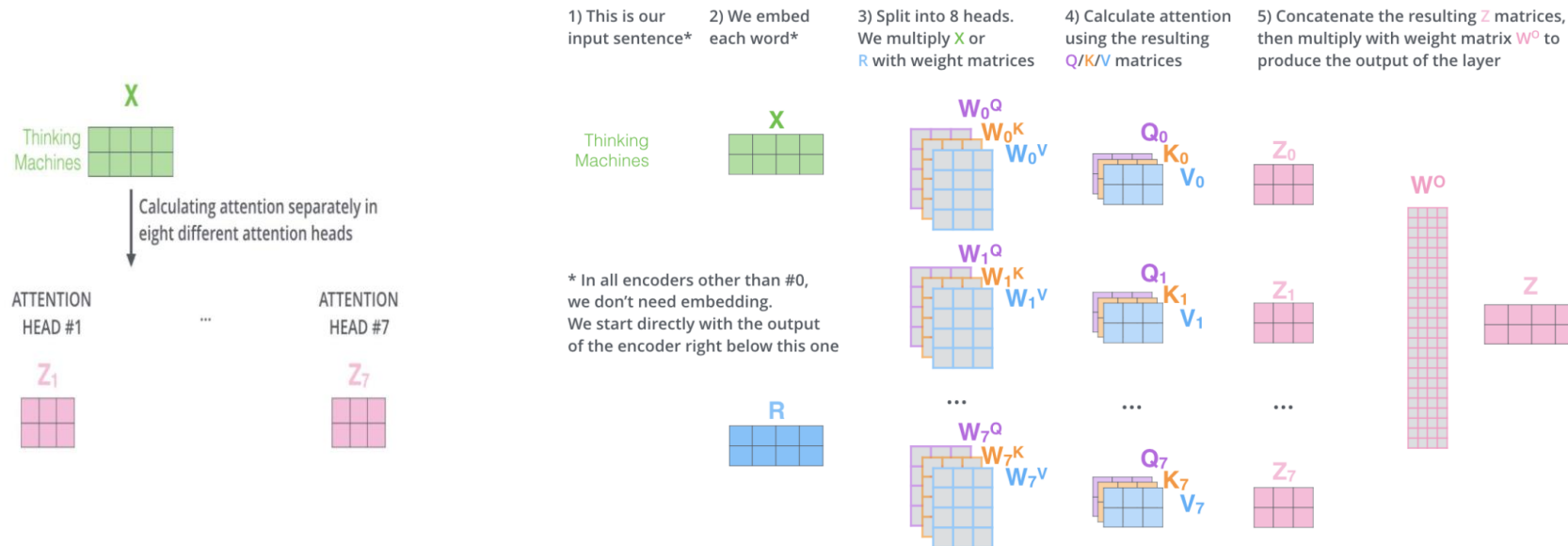


Attention: Self-Attention / Feed Forward-Attention

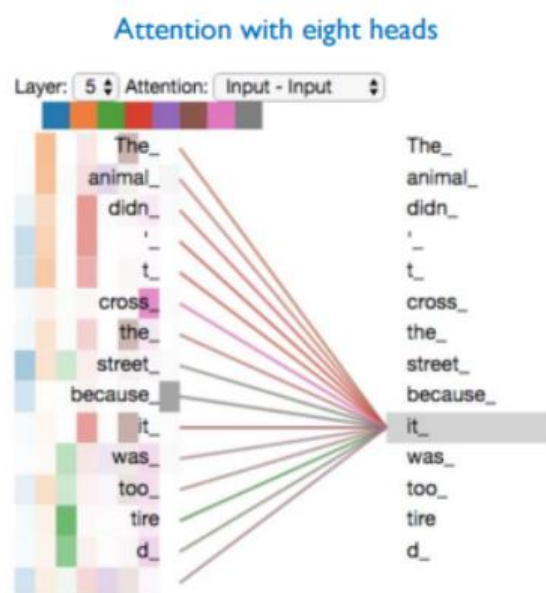
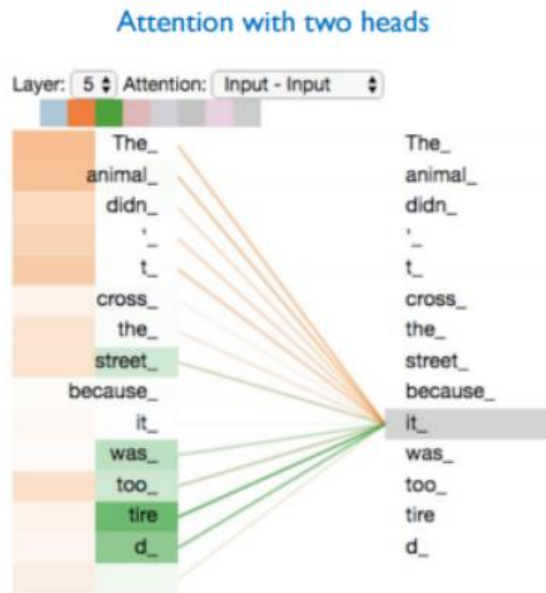


- Self-Attention Layer는 Dependency가 있고,
- Feed Forward Layer는 Dependency가 없다.

Attention: Multi-head Attention

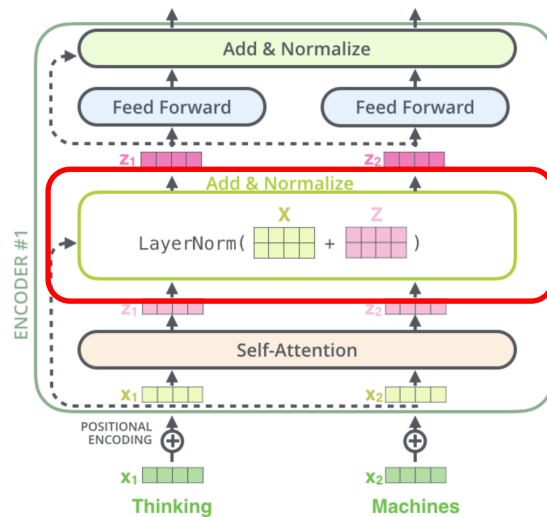
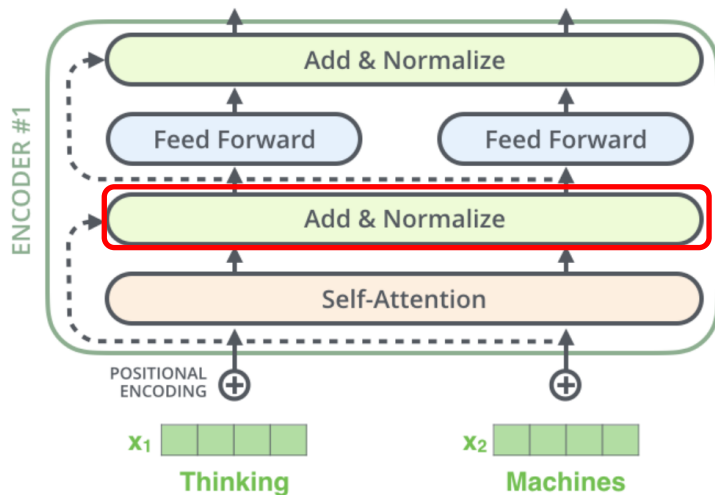


Attention: Multi-head Attention



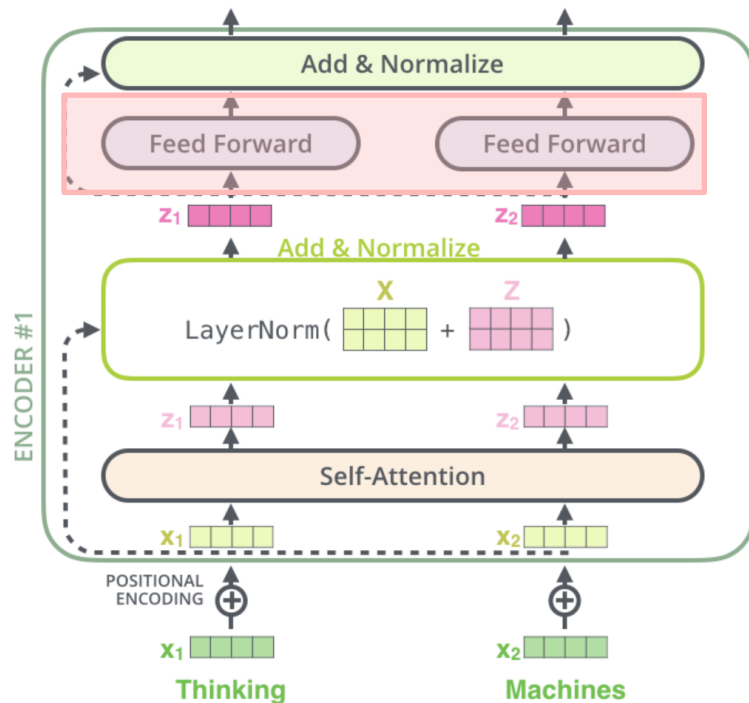
Residual

- 각 sub Layer마다 residual connection이 진행된다.
 $f(x) \rightarrow f(x) + x$: 미분했을 때 최소 1 이상으로 만들어주기 위해

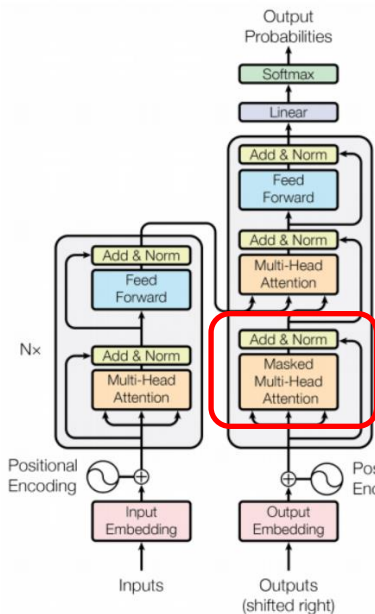


Position-wise-Feed-Forward Networks

- Fully connected feed-forward network
- 각 위치에서 독립적으로 이루어진다
- $FFN(x) = \max(0, xW1 + b1) W2 + b2$
- 각 층마다 독립적인 parameter를 가진다.
(같은 층에서는 같은 parameter 공유)



Attention: Multi-Head-Attention / 마스크 어텐션

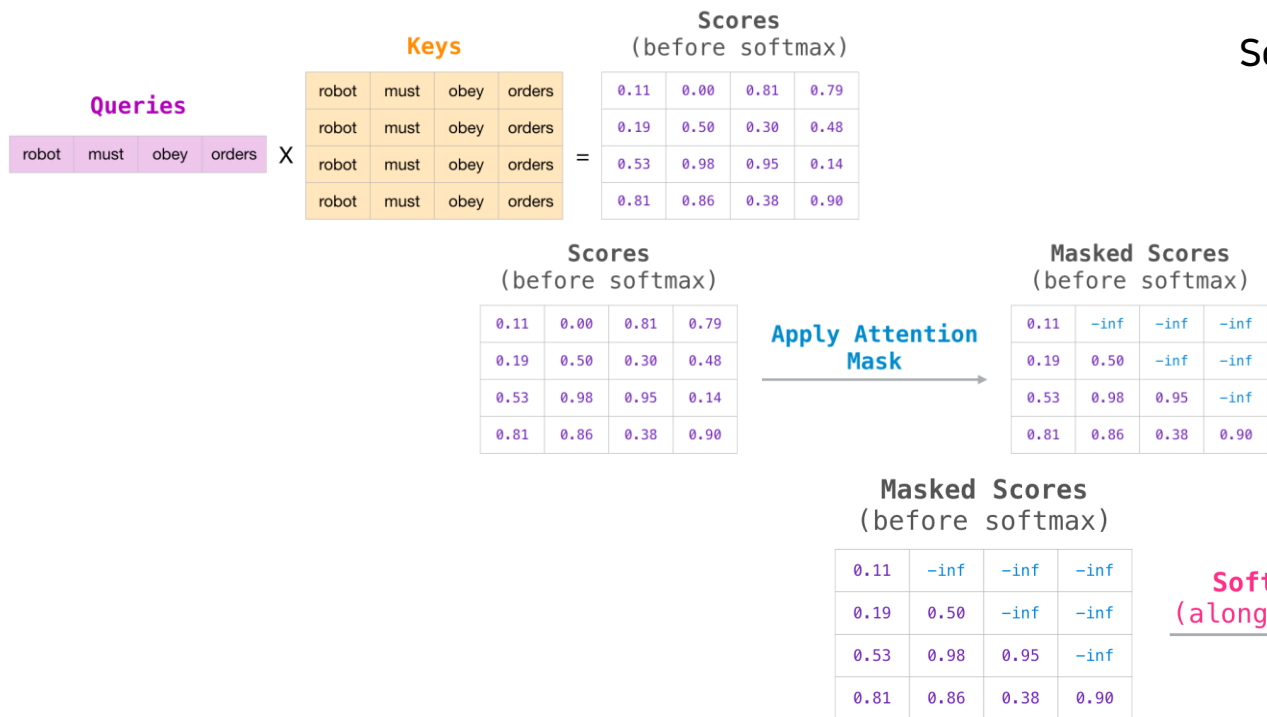


- Masked Multi-head Attention

: 자신보다 앞에 있는 토큰들의 어텐션 스코어만 볼 수 있다.

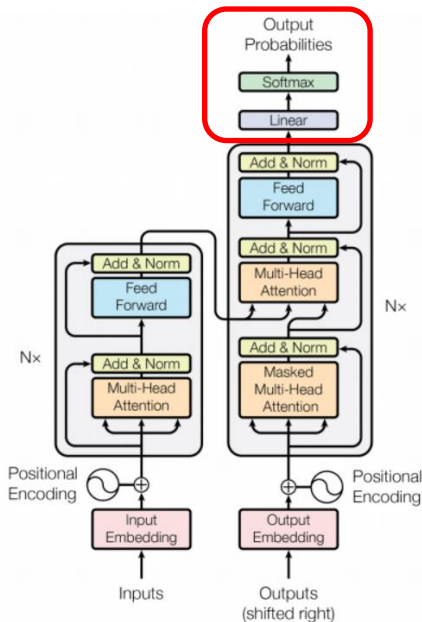
		Features				Labels	
		position: 1	2	3	4		
Example:	1	robot	must	obey	orders	must	
	2	robot	must	obey	orders	obey	
	3	robot	must	obey	orders	orders	
	4	robot	must	obey	orders	<eos>	

Attention: Multi-Head-Attention / 마스크 어텐션

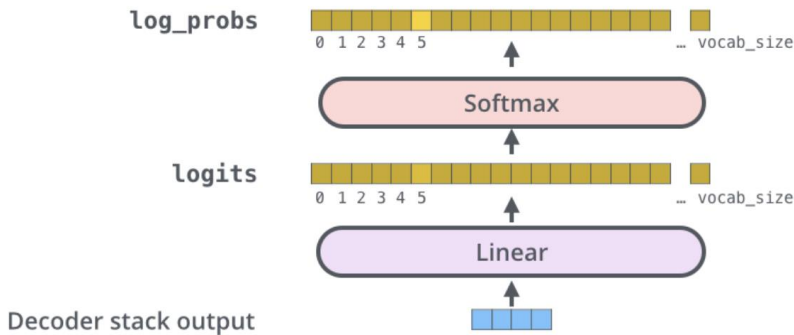


Softmax 함수: $f(Xi) = \frac{e^{Xi}}{\sum_{k=1}^K e^{Xk}}$

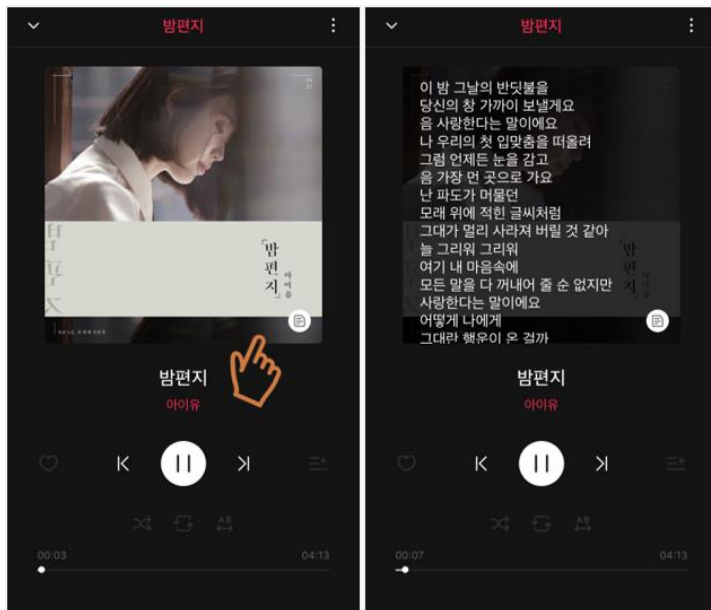
The Final Linear and Softmax Layer



- The Final Linear and Softmax Layer
- ✓ Linear layer: 결과를 logits vector를 반환해주는 neural network
- ✓ Softmax layer: 위 값들을 확률값으로 변환



Project 목표!!



GPT-2 model을 활용



문법에 맞고! 서사가 있는!

가사 만들기!!

Thank you /
Q/A