

트랜스포머 기반 한국어 언어 이해

지능정보연구본부 / 언어지능연구실

류지희 (chrisjihee@etri.re.kr)

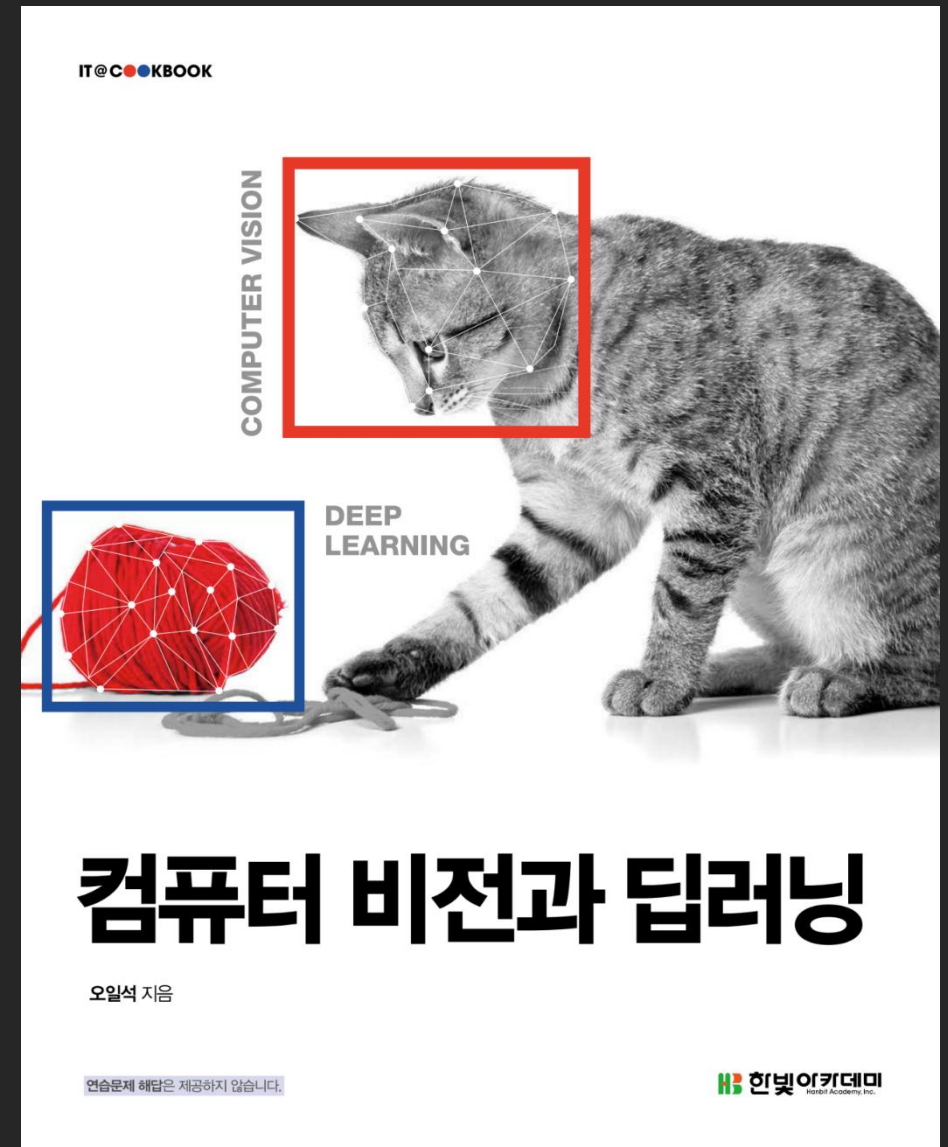
2024. 03. 19.

BERT 언어모델 등장 과정

참고문헌

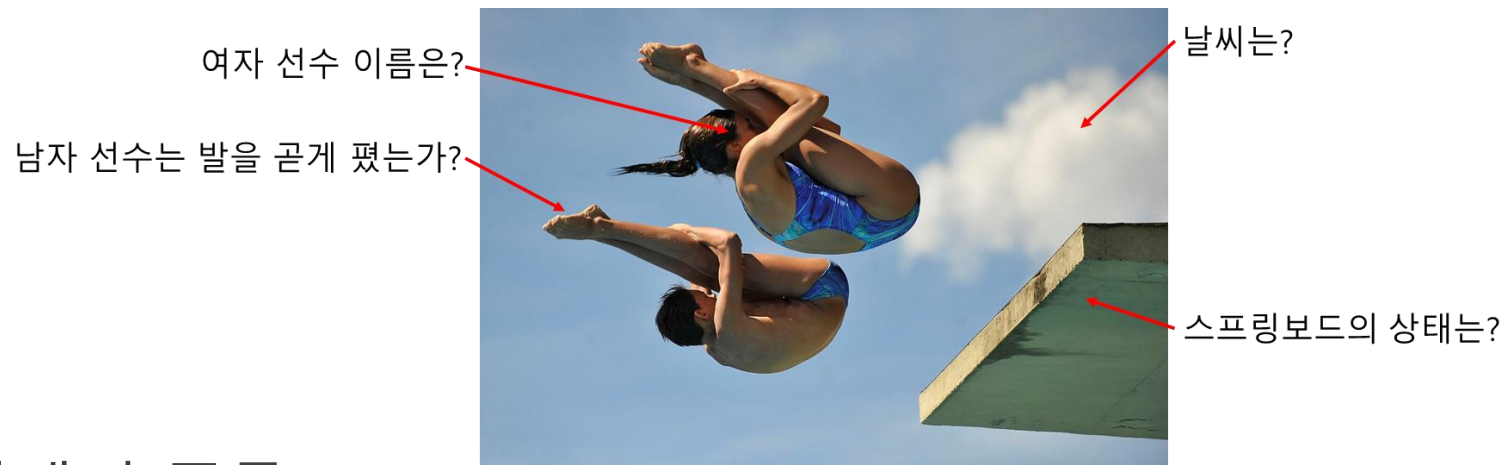
- [Vaswani2017] A. Vaswani, et al., "Attention is all you need," NIPS.
- [Han2022] K. Han, et al., "A survey on vision transformer," IEEE Tr. PAMI.
- [Khan2022] S. Khan, et al., "Transformers in vision: a survey," ACM Computing Surveys.
- [오일석2021] 오일석, 이진선, 파이썬으로 만드는 인공지능, 한빛아카데미.
- [오일석2022] 오일석, 컴퓨터 비전과 딥러닝, 11장(비전 트랜스포머)

(교수님 강의 자료를 동의 하에 발췌하였음)



인간의 주목

- 인간은 인식할 때 의도_{intention}에 따라 특정한 곳에 주목_{attention}
 - 질문에 따라 의도가 형성되고 의도에 따라 주목할 곳을 정함
 - 인간의 주목 맵 측정 [Das2016], 뇌가 어떻게 주목을 달성하는지 밝힘 [Corbetta2002]



- 기계의 주목
 - 예) VQA(visual question answering) 시스템: 주목을 잘 처리해야 높은 성능 달성
 - 컴퓨터비전은 분류, 검출, 분할, 추적 등의 문제를 푸는데 주목 적용

딥러닝의 주목

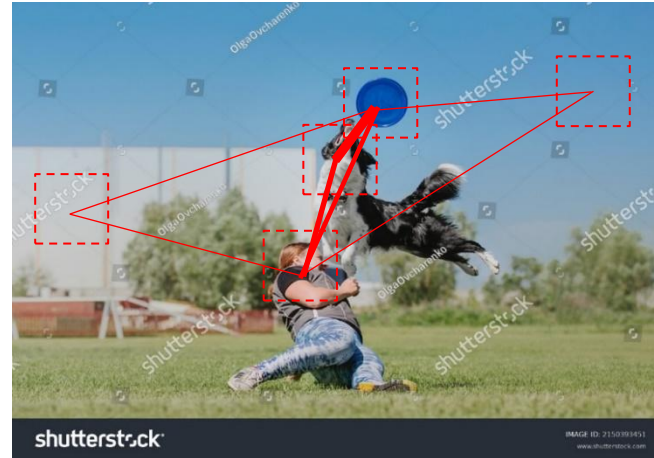
- 주목

- 영상의 중요한 부분에 더 큰 가중치를 주어 성능 개선
- 그림에서 색의 농도로 주목하는 정도를 표시

주목



자기 주목



- 자기 주목_{self attention}

- 영상을 구성하는 요소 상호간에 주목을 결정
- 그림에서는 선의 굵기로 주목하는 정도를 표시

자기 주목

■ 초기 자기 주목

- Non-local 신경망은 자기 주목을 컴퓨터비전에 최초 도입한 논문 [Wang2018]
 - 아래 식으로 자기 주목 계산: 출력 특징 맵 \mathbf{y} 는 입력 특징 맵 \mathbf{x} 와 같은 크기
 - \mathbf{y} 의 i 위치의 값 y_i 는 \mathbf{x} 의 모든 위치의 값인 $g(\mathbf{x}_j)$ 의 가중치 합
 - 가중치 $f(\mathbf{x}_i, \mathbf{x}_j)$ 는 \mathbf{x}_i 가 \mathbf{x}_j 에 얼마나 주목해야하는지 나타내는 값
 - $f(\mathbf{x}_i, \mathbf{x}_j)$ 는 주로 내적과 같이 \mathbf{x}_i 와 \mathbf{x}_j 의 유사도를 측정해주는 함수 사용
 - 보통 컨볼루션층을 여러 번 거쳐 7*7처럼 작아진 특징 맵에 적용

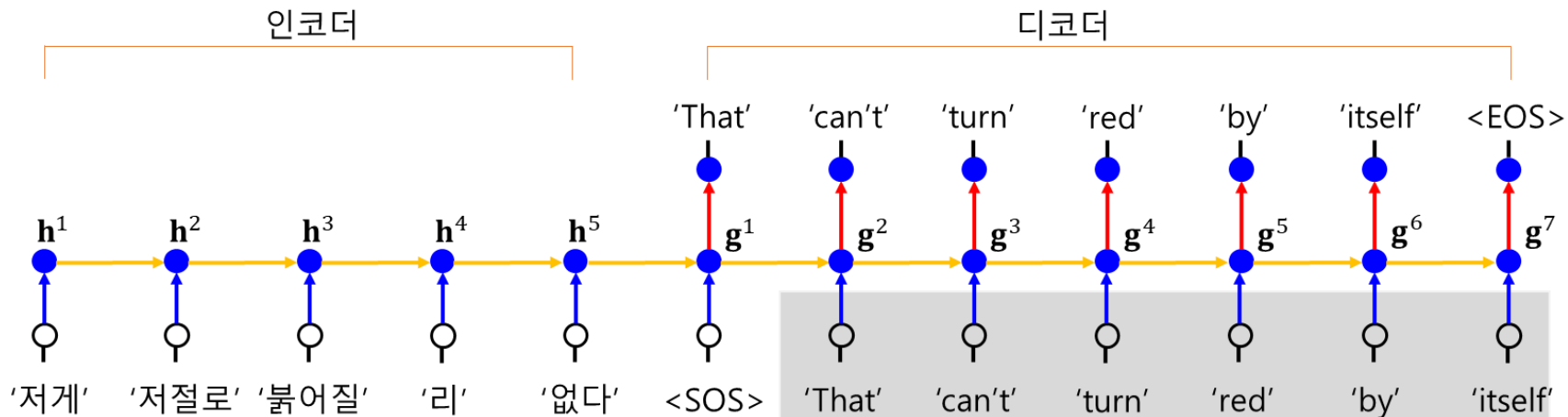
$$y_i = \frac{1}{C(\mathbf{x})} \sum_{\text{모든 } j} f(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_j)$$

■ 2017년 트랜스포머 등장

- 자연어처리 분야의 혁신
- 2020~2022년에는 컴퓨터비전에 적용한 성공 사례 봇물

Seq2seq 모델의 등장

- 서츠케버의 seq2seq 모델
 - 2014년에 발표된 자연어처리 분야의 획기적 아이디어
 - 가변 길이의 문장을 가변 길이의 문장으로 변환, 예) 언어 번역
 - 자기 회귀 방식으로 동작
 - 큰 공헌 했지만 한계를 보임: 인코더의 마지막 은닉 상태만 디코더로 전달함

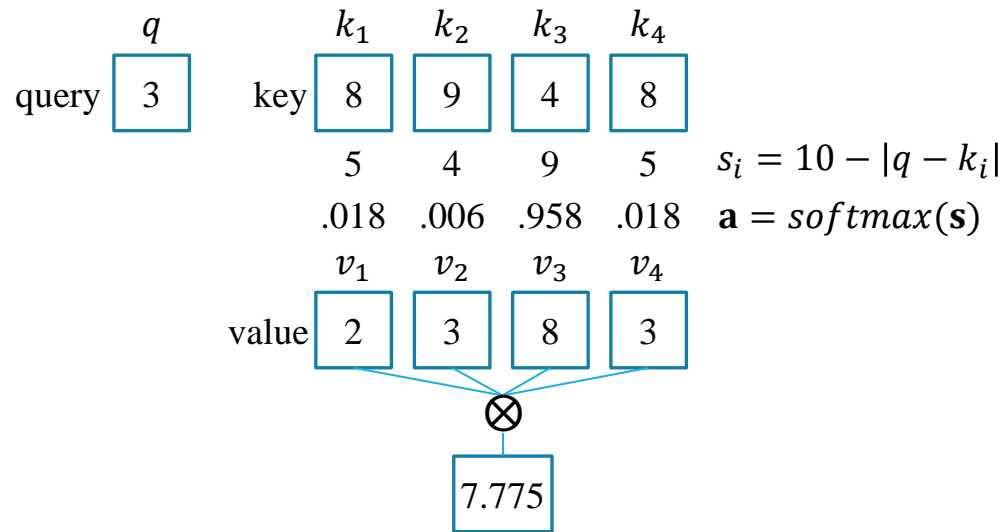


Query-Key-Value로 계산하는 주목

- 주목 계산 방법이 여러인데, 최근에는 query-key-value 방식을 주로 사용
 - Query가 key와 유사한 정도를 측정하고 value를 가중 합함
 - 트랜스포머가 이 방법을 사용하므로 확실히 이해해야 함

Query-Key-Value로 계산하는 주목

- [예제 1] 1~10 자연수



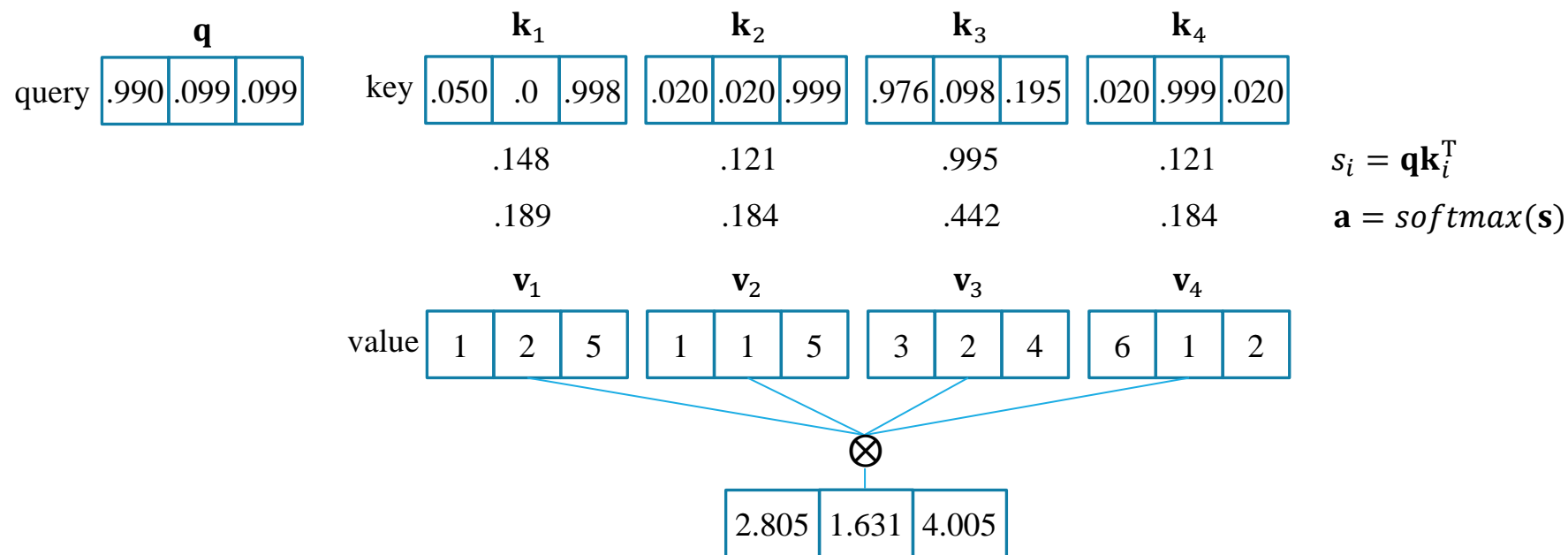
$$c = 0.018 * 2 + 0.006 * 3 + 0.958 * 8 + 0.018 * 3 = 7.775$$

- Query q 는 세번째 key k_3 과 가장 유사하여 세번째 value v_3 에 가장 많이 주목함

$$c = \text{softmax}(\text{similarity}(q, k_1), \text{similarity}(q, k_2), \dots, \text{similarity}(q, k_T)) \begin{pmatrix} v_1 \\ \vdots \\ v_T \end{pmatrix}$$

Query-Key-Value로 계산하는 주목

- [예제 2] 벡터로 확장



Query-Key-Value로 계산하는 주목

- [예제 3] 행렬로 확장

$$\mathbf{q} = (0.990 \quad 0.099 \quad 0.099), \mathbf{K} = \begin{pmatrix} 0.050 & 0.000 & 0.998 \\ 0.020 & 0.020 & 0.999 \\ 0.976 & 0.098 & 0.195 \\ 0.020 & 0.999 & 0.020 \end{pmatrix}, \mathbf{V} = \begin{pmatrix} 1 & 2 & 5 \\ 1 & 1 & 5 \\ 3 & 2 & 4 \\ 6 & 1 & 2 \end{pmatrix}$$

주목 벡터

$$\begin{aligned} \mathbf{a} &= \text{softmax}(\mathbf{qK}^T) = \text{softmax} \left((0.990 \quad 0.099 \quad 0.099) \begin{pmatrix} 0.050 & 0.020 & 0.976 & 0.020 \\ 0.000 & 0.020 & 0.098 & 0.999 \\ 0.998 & 0.999 & 0.195 & 0.020 \end{pmatrix} \right) \\ &= (0.189 \quad 0.184 \quad 0.442 \quad 0.184) \end{aligned}$$

문맥 벡터

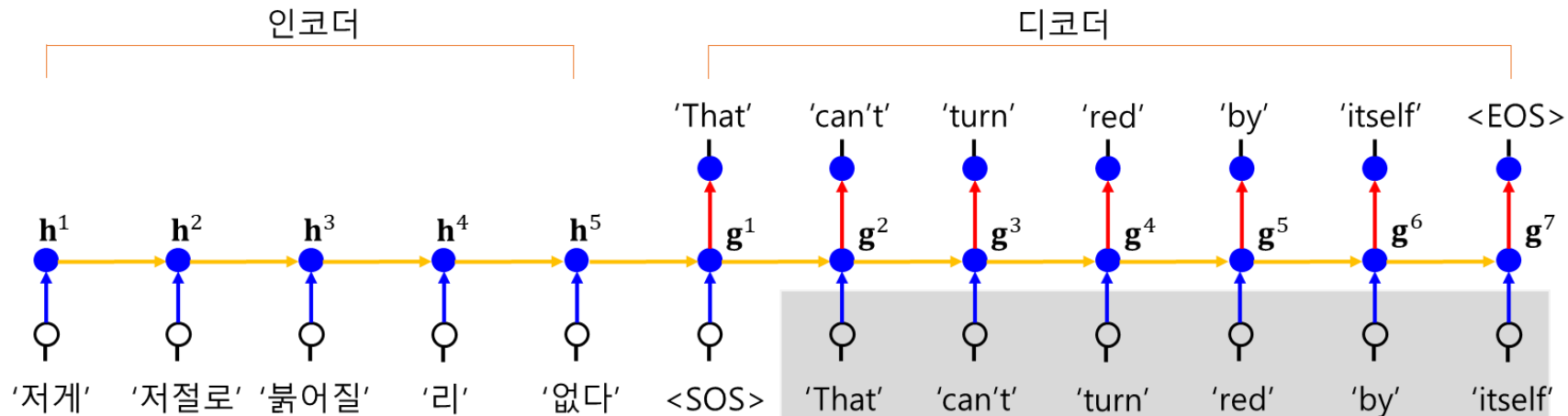
$$\begin{aligned} \mathbf{c} &= \text{softmax}(\mathbf{qK}^T)\mathbf{V} = \mathbf{aV} \\ &= (0.189 \quad 0.184 \quad 0.442 \quad 0.184) \begin{pmatrix} 1 & 2 & 5 \\ 1 & 1 & 5 \\ 3 & 2 & 4 \\ 6 & 1 & 2 \end{pmatrix} \\ &= (2.805 \quad 1.631 \quad 4.005) \end{aligned}$$

$\mathbf{c} = \text{softmax}(\mathbf{qK}^T)\mathbf{V}$

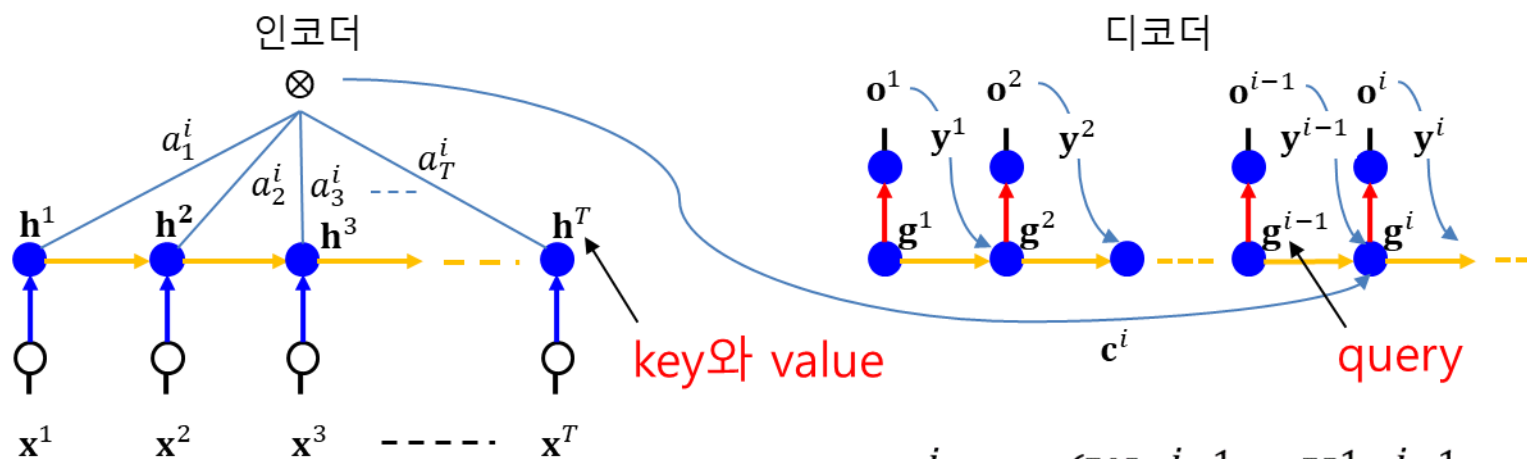
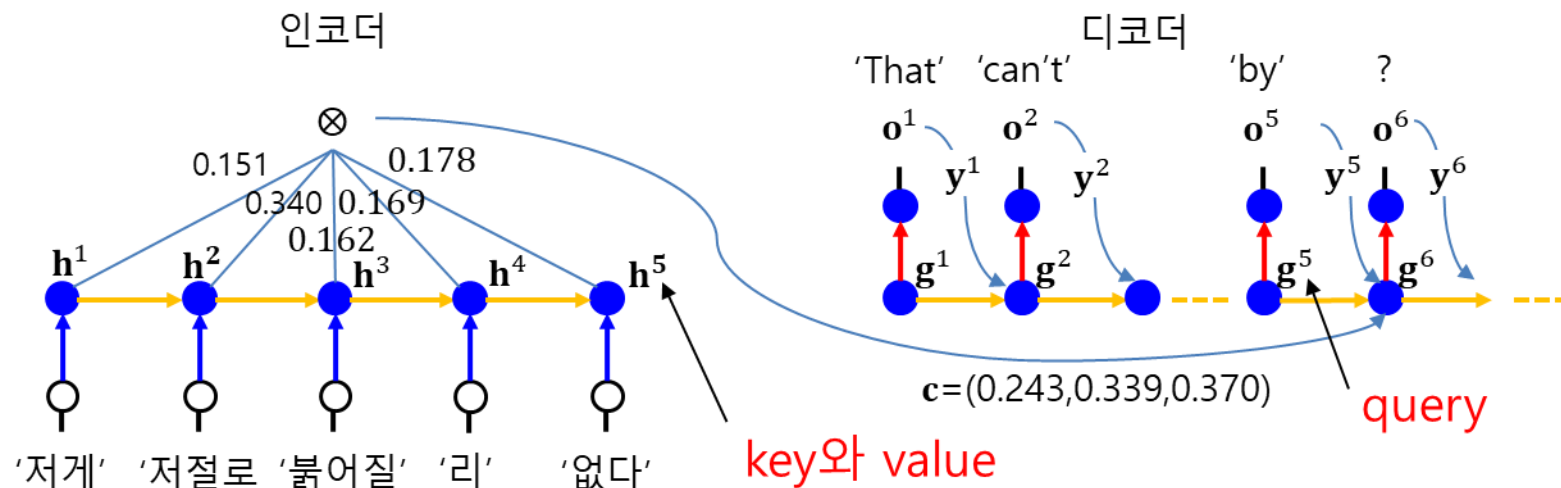
남은 일은 무엇을 query로 하고, 무엇을 key로 하고,
무엇을 value로 할 지 결정하는 것뿐

주목을 반영한 seq2seq 모델

- 바다나우의 아이디어
 - 디코더가 인코더의 모든 상태 $\mathbf{h}^1, \mathbf{h}^2, \dots$ 에 접근 허용하고 주먹을 적용함
 - 예) 순간 6에서 인코더의 순간 2의 '저절로'에 주목해야 'itself'를 제대로 생성할 수 있을것임. 디코더의 순간 6에서 주먹 벡터 $\mathbf{a}=(0.01, 0.9, 0.02, 0.03, 0.04)$ 이면 바람직
 - 무엇을 query, key, value로 해야 이런 의도가 반영될까?



주목을 반영한 seq2seq 모델



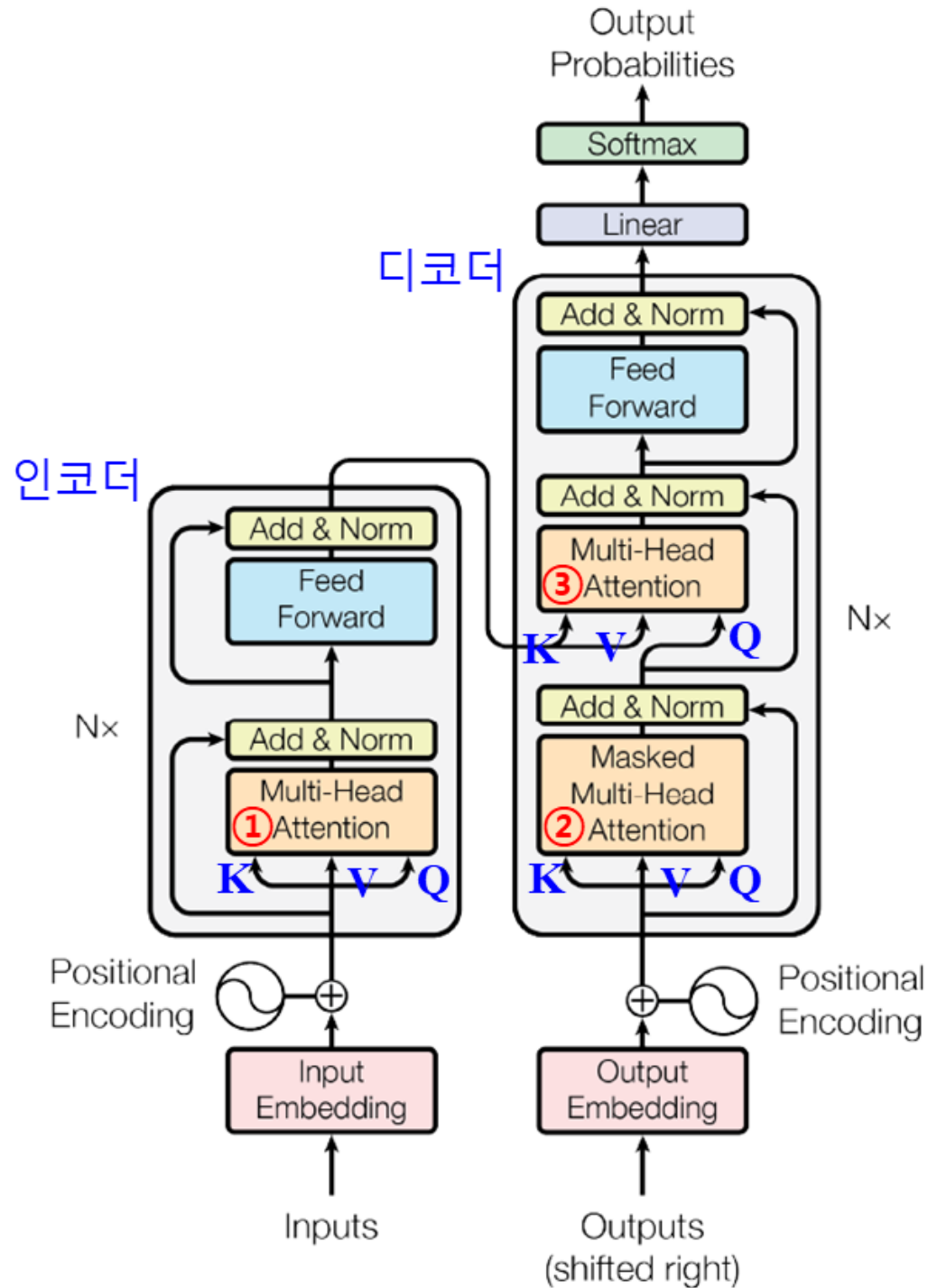
$$\left. \begin{aligned} \mathbf{g}^i &= \tau_1(\mathbf{W}\mathbf{g}^{i-1} + \mathbf{U}^1\mathbf{y}^{i-1} + \mathbf{Z}\mathbf{c}^i) \\ \mathbf{o}^i &= \tau_2(\mathbf{U}^2\mathbf{g}^i) \end{aligned} \right\}$$



트랜스포머의 등장

- “Attention is all you need” 논문
 - 구글 연구진이 2017년 발표
 - 자연어 처리를 위한 혁신적인 모델
“트랜스포머” 제안
- 컨볼루션 신경망과 순환 신경망의
구성요소를 모두 없애고 주목만으로
신경망을 구현

트랜스포머의 기본 구조



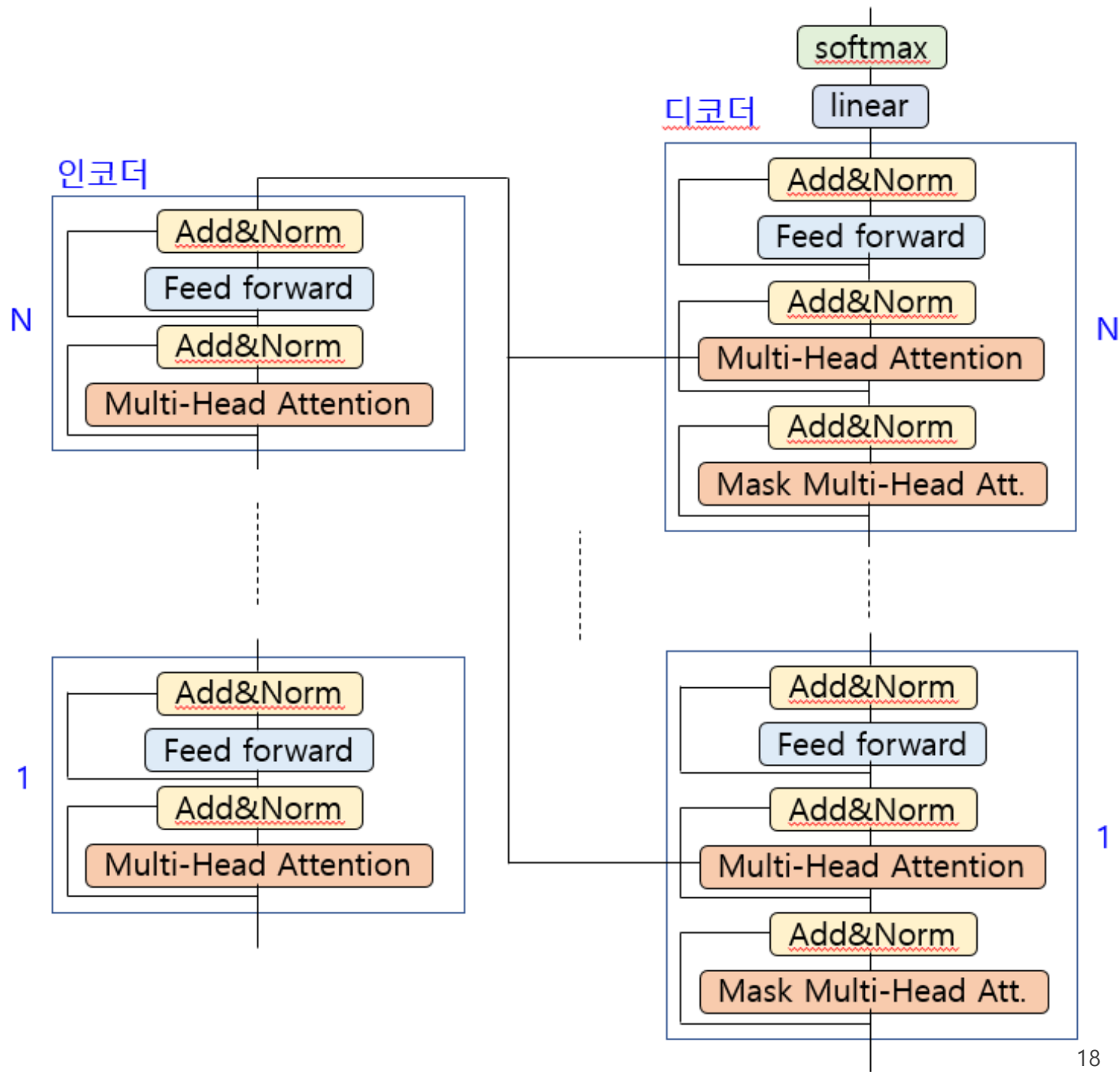
트랜스포머의 기본 구조

- 인코더와 디코더로 구성
 - 한영번역 트랜스포머라면, 인코더에 한국어 입력, 디코더에서 영어 출력
 - 문장의 모든 단어를 한꺼번에 입력 → 자기 주목이 가능해짐. 병렬 처리 유리
 - 단어는 임베딩을 통해 d_{model} 차원의 임베딩 벡터로 표현. 논문은 $d_{model}=512$
 - 위치 인코딩으로 위치 정보 추가
 - MHA층이 자기 주목을 담당. 헤드를 여러 개 사용하여 성능 향상
 - Feed Forward층
 - Add&Norm 적용
- N개의 인코더 및 디코더층을 반복해서 쌓음 (N=6)

트랜스포머의 기본 구조

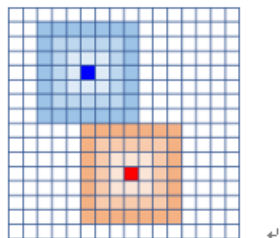
- 세 개의 MHA층이 주목을 담당
 - 인코더의 ①과 디코더의 ② 표시한 MHA층은 자기 주목을 담당
 - 문장을 구성하는 단어끼리 주목을 처리
 - 디코더의 ③ 표시한 MHA는 자기 주목이 아님
 - 인코더와 디코더 사이의 주목을 담당
 - 인코더의 상태 벡터가 key와 value, 디코더의 상태 벡터는 query로 작용함
- MHA로 처리하는 자기 주목은 트랜스포머의 핵심 아이디어!

트랜스포머의 전체 동작

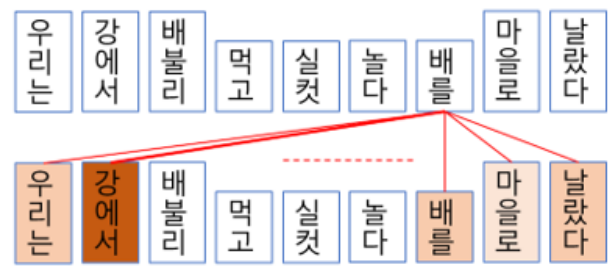


트랜스포머의 특성

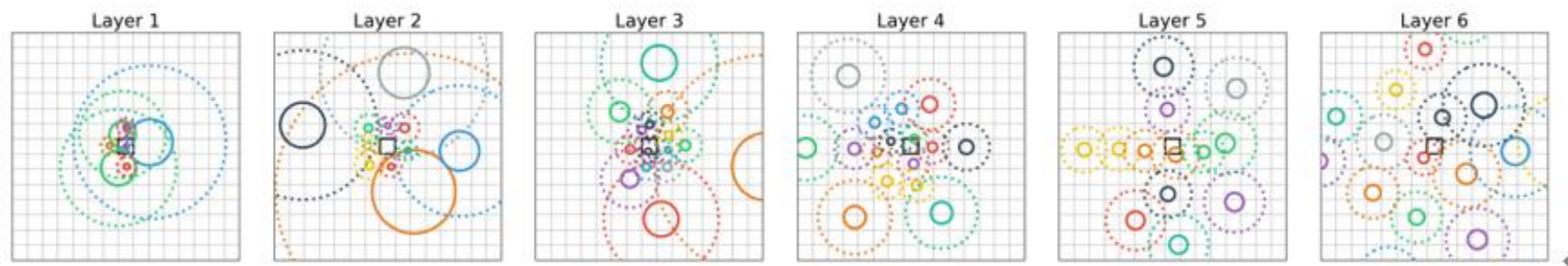
- 장거리 의존 long-range dependency



[그림 11-28] 컨볼루션 연산을 통한 장거리 의존



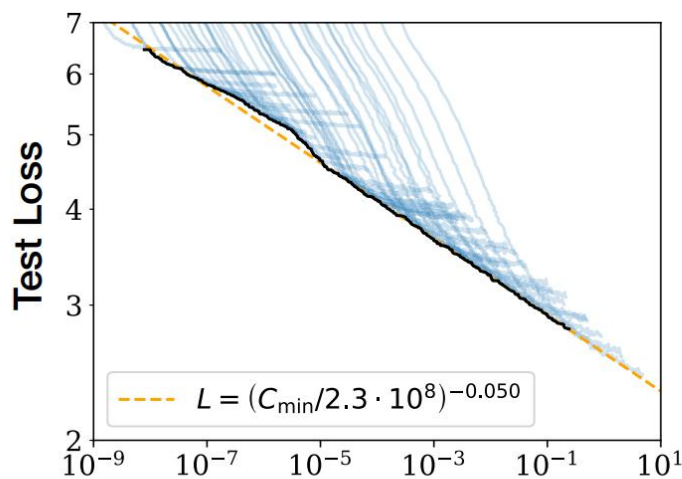
[그림 11-29] 트랜스포머는 자기 주목을 통해 장거리 의존을 표현



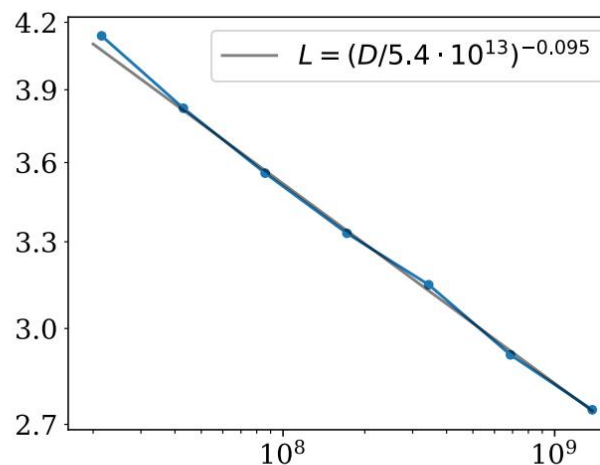
[그림 11-30] 지역 정보와 전역 정보를 모두 능숙하게 처리하는 트랜스포머 [Cordonnier2020]

트랜스포머의 특성

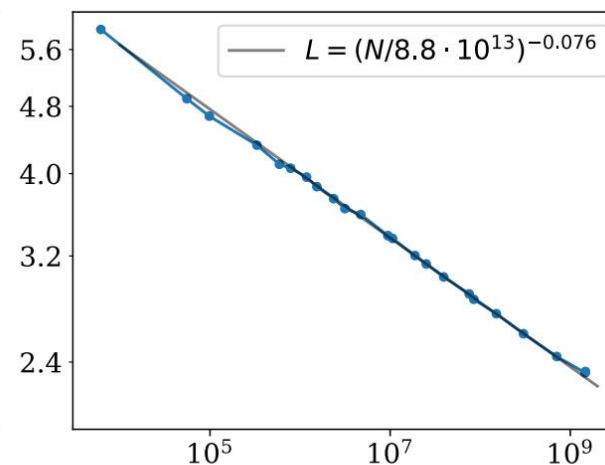
- 확장성 scalability
 - 초기 트랜스포머는 40M개를 조금 넘는 수준, 이후 초거대 모델 등장
 - GPT-3: 175B개
 - Switch 트랜스포머: 1.6T개
 - 모델 크기, 데이터셋 크기, 학습 계산 자원 사이의 관계 분석 연구 발표
 - 큰 모델을 사용하는 것이 유리하다는 근거 제시



PF-days, non-embedding



tokens

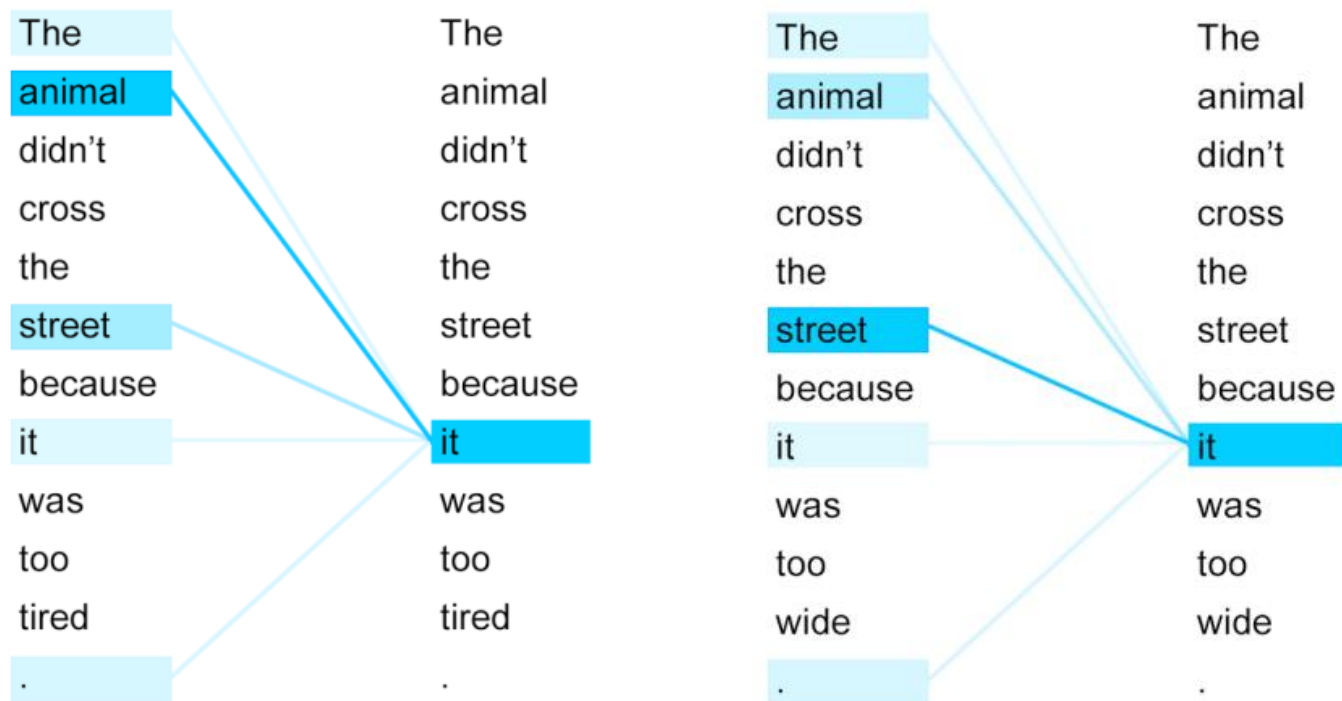


non-embedding

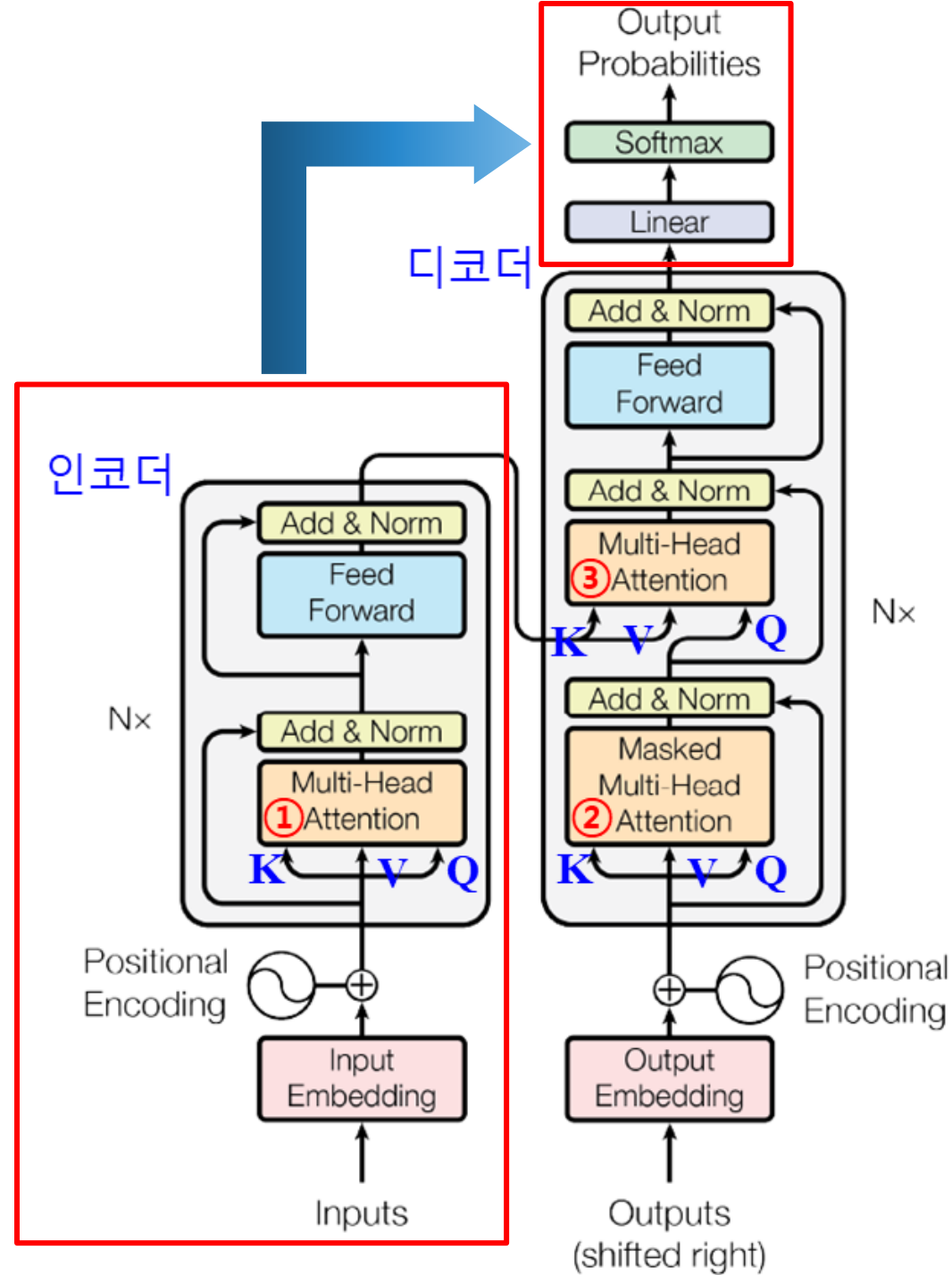
자기 주목 결과 시각화 예시



Encoder Self-Attention

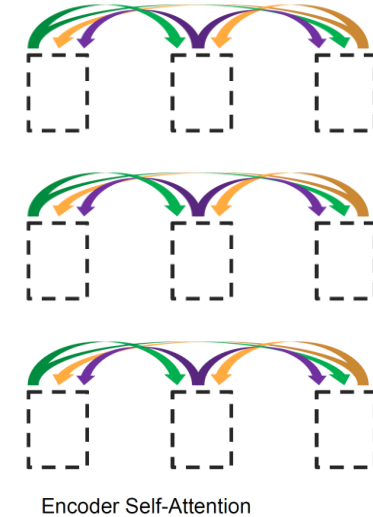
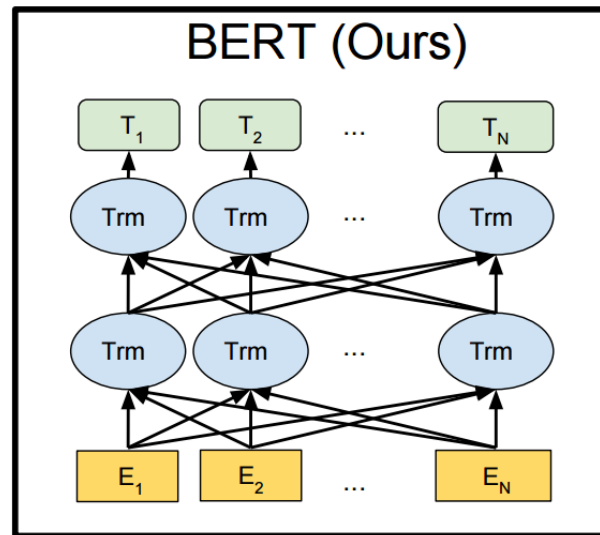
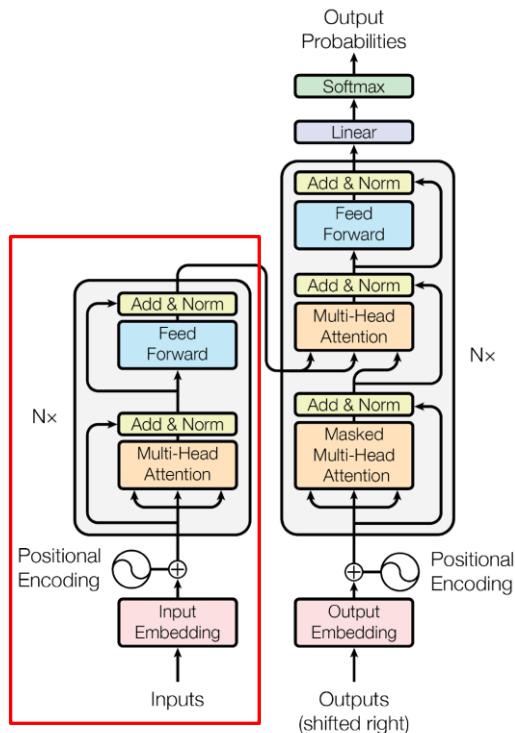


BERT의 등장

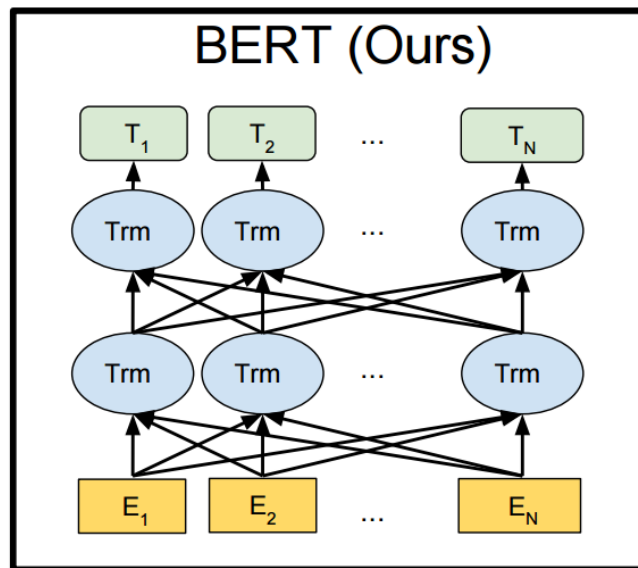


BERT 모델

- BERT: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers
 - Pre-training of Deep Bidirectional Transformers for Language Understanding (Google AI Language, 2018)



BERT 입력



Input	[CLS]	my	dog	is	cute	[SEP]	he	likes	play	##ing	[SEP]
Token Embeddings	$E_{[CLS]}$	E_{my}	E_{dog}	E_{is}	E_{cute}	$E_{[SEP]}$	E_{he}	E_{likes}	E_{play}	$E_{##ing}$	$E_{[SEP]}$
	+	+	+	+	+	+	+	+	+	+	+
Segment Embeddings	E_A	E_A	E_A	E_A	E_A	E_A	E_B	E_B	E_B	E_B	E_B
	+	+	+	+	+	+	+	+	+	+	+
Position Embeddings	E_0	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}

(학습) 1단계 : Pre-training

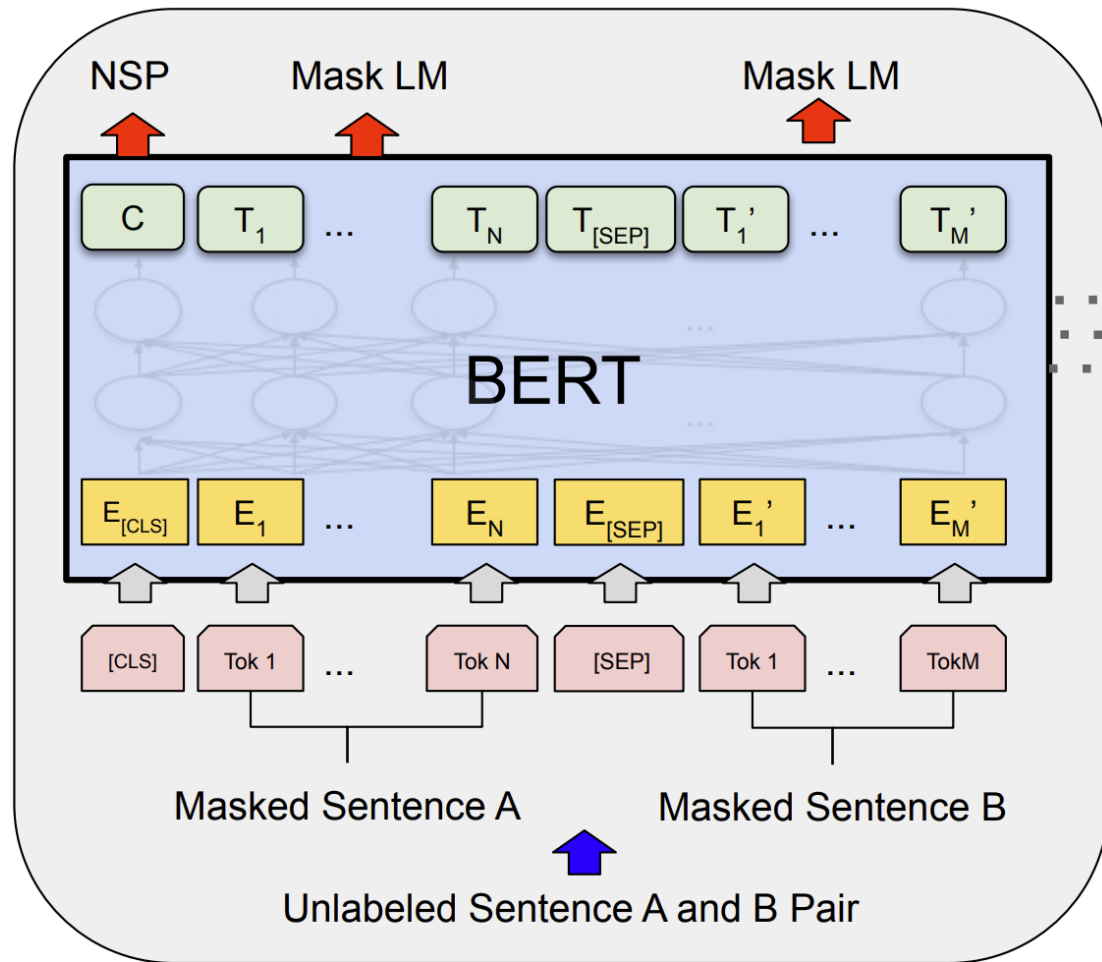
- Task#1: Masked LM

Input: the man went to the [MASK1] . he bought a [MASK2] of milk.
Labels: [MASK1] = store; [MASK2] = gallon

- Task#2: Next Sentence Prediction

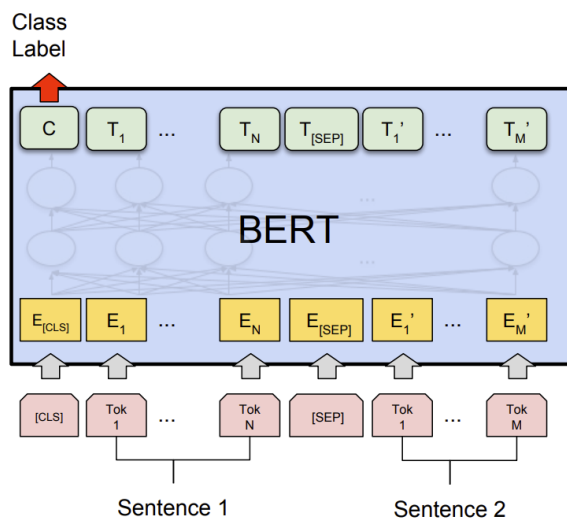
Sentence A: the man went to the store .
Sentence B: he bought a gallon of milk .
Label: IsNextSentence

Sentence A: the man went to the store .
Sentence B: penguins are flightless .
Label: NotNextSentence

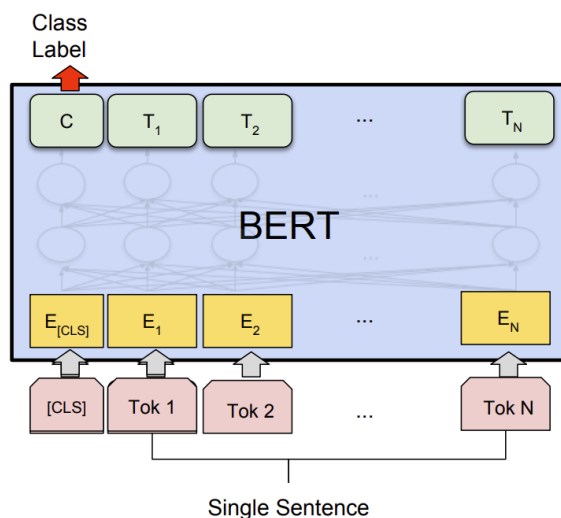


Pre-training

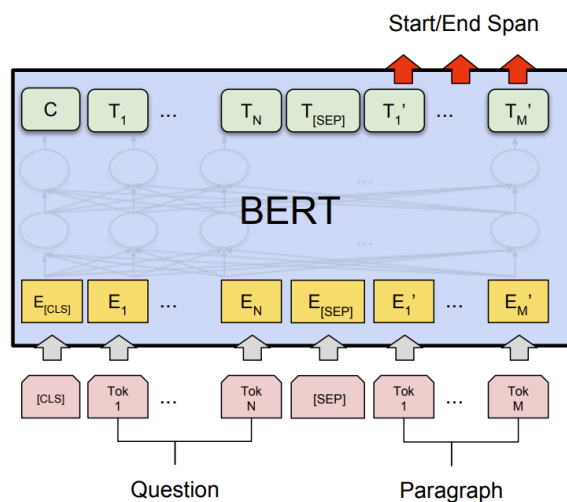
(학습) 1단계 :
Pre-training



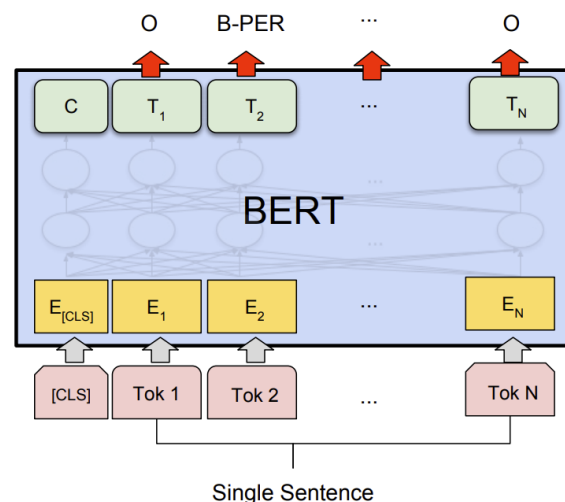
(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG



(b) Single Sentence Classification Tasks:
SST-2, CoLA



(c) Question Answering Tasks:
SQuAD v1.1



(d) Single Sentence Tagging Tasks:
CoNLL-2003 NER

(학습) 2단계 : Fine-tuning

BERT 언어모델 후속 연구

언어 처리 길이 확장 연구

■ Long Range Arena 벤치마크 결과

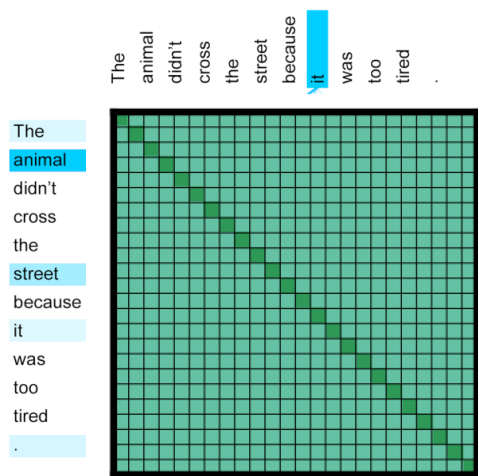
Model	ListOps	Text	Retrieval	Image	Pathfinder	Path-X	Avg
Transformer	36.37	64.27	57.46	42.44	71.40	FAIL	<u>54.39</u>
Local Attention	15.82	52.98	53.39	41.46	66.63	FAIL	46.06
Sparse Trans.	17.07	63.58	59.59	44.24	71.71	FAIL	51.24
Longformer	35.63	62.85	56.89	42.22	69.71	FAIL	53.46
Linformer	35.70	53.94	52.27	38.56	<u>76.34</u>	FAIL	51.36
Reformer	37.27	56.10	53.40	38.07	68.50	FAIL	50.67
Sinkhorn Trans.	33.67	61.20	53.83	41.23	67.45	FAIL	51.39
Synthesizer	<u>36.99</u>	61.68	54.67	41.61	69.45	FAIL	52.88
BigBird	36.05	64.02	59.29	40.83	74.87	FAIL	55.01
Linear Trans.	16.13	65.90	53.09	42.34	75.30	FAIL	50.55
Performer	18.01	<u>65.40</u>	53.82	<u>42.77</u>	77.05	FAIL	51.41
Task Avg (Std)	29 (9.7)	61 (4.6)	55 (2.6)	41 (1.8)	72 (3.7)	FAIL	52 (2.4)

Table 1: Experimental results on Long-Range Arena benchmark. Best model is in boldface and second best is underlined. All models do not learn anything on Path-X task, contrary to the Pathfinder task and this is denoted by FAIL. This shows that increasing the sequence length can cause seriously difficulties for model training. We leave Path-X on this benchmark for future challengers but do not include it on the Average score as it has no impact on relative performance.

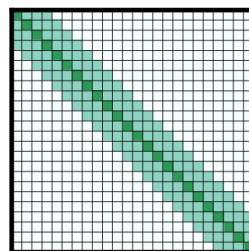
Longformer: The Long-Document Transformer

Longformer: The Long-Document Transformer

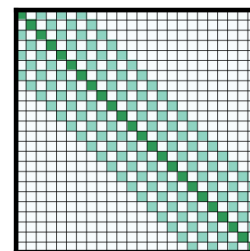
Iz Beltagy* **Matthew E. Peters*** **Arman Cohan***
Allen Institute for Artificial Intelligence, Seattle, WA, USA
{beltagy, matthewp, armanc}@allenai.org



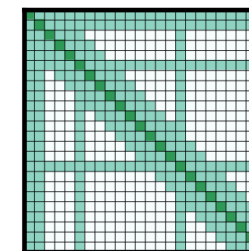
(a) Full n^2 attention



(b) Sliding window attention



(c) Dilated sliding window



(d) Global+sliding window

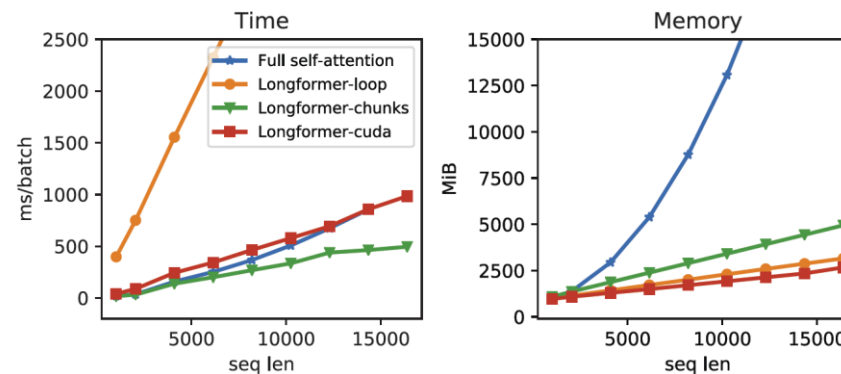


Figure 2: Comparing the full self-attention pattern and the configuration of attention patterns in our Longformer.

Big Bird: Transformers for Longer Sequences

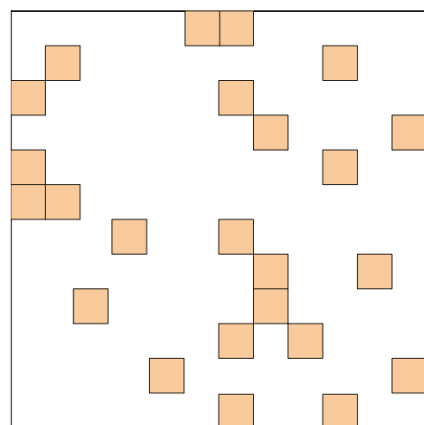
Big Bird: Transformers for Longer Sequences

Manzil Zaheer
Guru Guruganesh
*Google Research,
Mountain View, CA, USA*

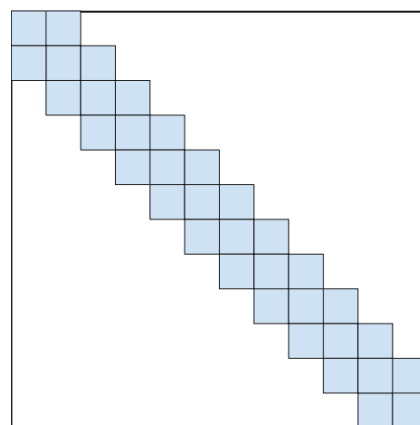
MANZILZ@GOOGLE.COM
GURUG@GOOGLE.COM

Avinava Dubey
Joshua Ainslie, Chris Alberti, Santiago Ontanon,
Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang,
Amr Ahmed
Google Research, USA

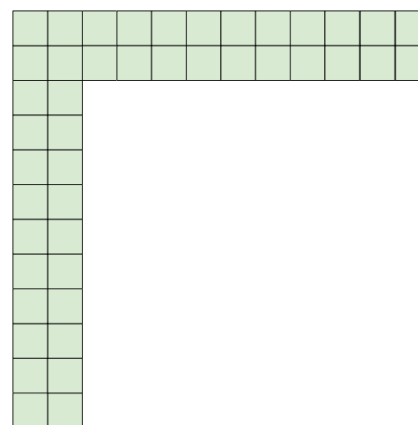
AVINAVADUBEY@GOOGLE.COM



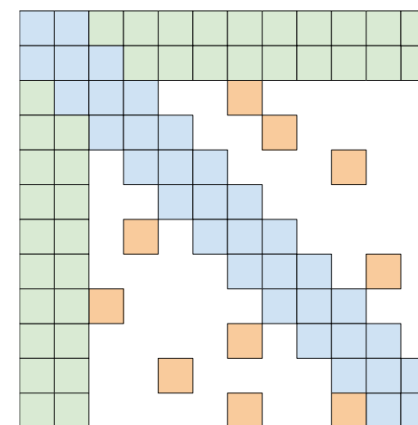
(a) Random attention



(b) Window attention



(c) Global Attention



(d) **BIGBIRD**

BERT 모델 경량화 연구

- DistilBert: a **distilled version of BERT** (Oct. 2019)
 - general purpose language representation model
 - BERT를 40% 정도 줄이고, 60%나 빠르게 연산, 97%의 성능 유지
- (참고) Knowledge Distillation [Bucila et al., 2006, Hinton et al., 2015]
 - larger model(teacher model)로부터 compact model(student model)을 만들어내는 방법
 - student를 학습하기 위해서 teacher의 output을 그대로 이용
 - 작은 모델을 바로 학습시키는 것보다 의미있는 이유는 near-zero인 확률들도 학습할 수 있기 때문

Encoder-Decoder Structure

- BERT
 - encoder structure
- BART, T5
 - encoder-decoder structure
 - achieved good results on both **generative** and **classification** tasks

BART

- BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension (Oct 2019, Facebook AI)

- <https://arxiv.org/abs/1910.13461>

- 다양한 denoising 방법 실험

- **Original text** : ABC. DE.

- **Token Masking**: 임의의 token을 [MASK]로 교체함

=> [MASK] token이 무엇인지 예측해야 함

- **Token Deletion**: 임의의 token을 삭제함

=> 삭제한 token의 위치를 찾아야 함

- **Text Infilling**: span length를 뽑아 하나의

[MASK] token으로 대체함

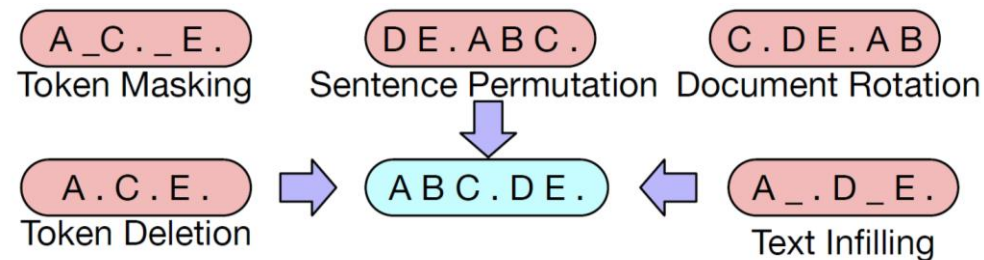
=> [MASK]로 대체된 token에

몇개의 token이 존재할지 예측해야 함

- **Sentence Permutation**: 문장의 순서를 랜덤으로 섞음

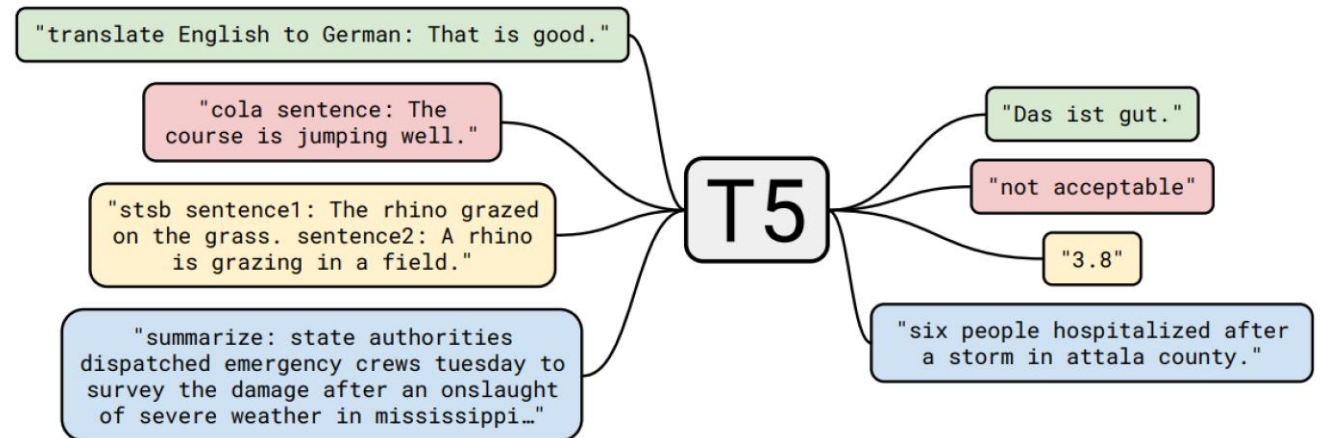
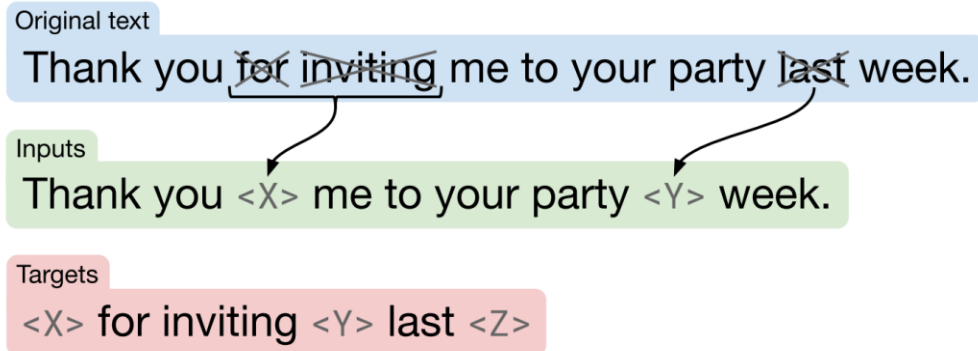
- **Document Rotation**: 하나의 token을 뽑은 후, 그 token을 시작점으로 회전함

=> 문서의 start point를 찾도록 학습시킴



Text-To-Text-Transfer-transformer (T5)

- Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer (Journal of Machine Learning Research 2020, Google)
- a unified framework that converts every text processing problem into "text-to-text" problem



Finetuning 구현 실습

세상의 속도를
따라잡고 싶다면

Do it!

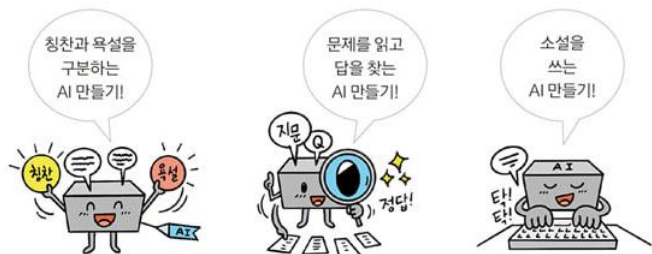
BERT와 GPT로 배우는

한국어
말뭉치!

자연어 처리

트랜스포머 핵심 원리와 허깅페이스 패키지 활용법

이기창 지음



네이버 영화평, 질의응답 말뭉치를 활용한 한국어 이해/생성 모델 만들기

이지스퍼블리싱

참고자료 및 코드

- (2024류지희) A4001 2일차 실습코드
<https://github.com/chrisjihee/DeepKNLP-24.03>
- (2021이기창) BERT와 GPT로 배우는 자연어처리
<https://ratsgo.github.io/nlpbook/>

주요 실습 내용

- 환경 설치
 - <https://github.com/chrisjihee/DeepKNLP-24.03/blob/master/setup-env.sh>
- 감성 분류 (학습, 평가, 서비스)
 - <https://github.com/chrisjihee/DeepKNLP-24.03/blob/master/1-cls.py>
- 개체명 인식 (학습, 평가, 서비스)
 - <https://github.com/chrisjihee/DeepKNLP-24.03/blob/master/2-ner.py>

추후 보강 예정

- 자율 숙제
 - <https://github.com/chrisjihee/DeepKNLP-24.03/blob/master/TODO.txt>
 - 5장 태스크 실습 및 수정 (4장과 매우 유사)
 - 7장 태스크 실습 및 수정 (Extractive QA 개념)
 - KLUE 태스크에 적용: YNAT, STS, NLI, MRC
 - T5 모델(seq-to-seq) 구현



감사합니다