

시각지능A – 자연어처리 기본

2. Transformer to LLM

목차 2. Transformer to LLM

1. Attention
2. Self-Attention
3. BERT
4. LLM

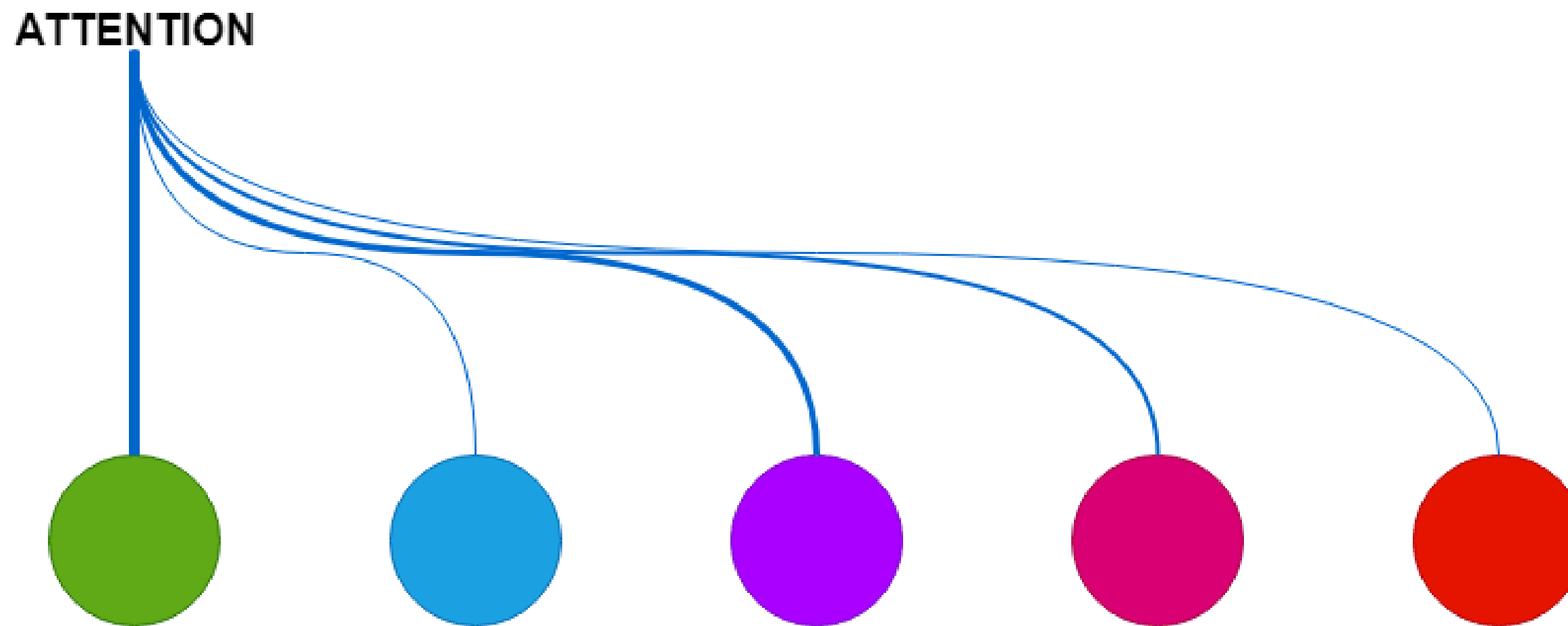
1. Attention과 Self-Attention

Attention Mechanism의 등장 배경

- 기본 Seq2Seq 모델의 한계
 - Context vector의 한계
 - 고정된 크기의 컨텍스트 벡터로 모든 정보 압축
 - 마지막 단어에 의존성이 지나치게 높음
 - 긴 입력 시퀀스의 경우 정보 손실 발생 가능성(Vanishing gradient)
- RNN 기반 모델의 한계
 - Task 수행 시, 각 단어가 갖는 영향력을 정확하게 반영할 수 없음
 - 기존에는 텍스트 전처리를 통해 많은 부분 해결(TF-IDF, W2V 등)

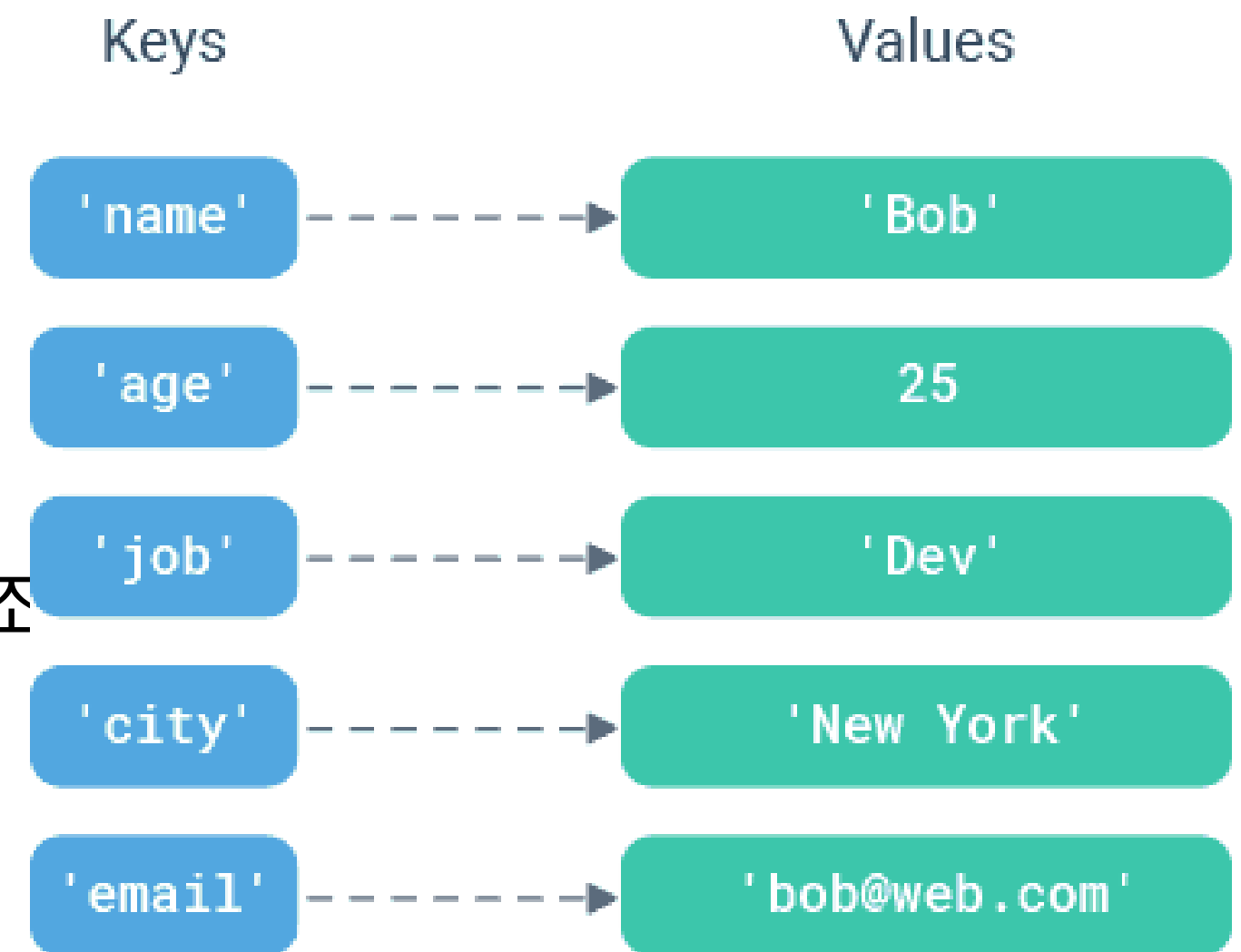
Attention Mechanism의 기본 아이디어

- 입력 시퀀스의 모든 부분이 동일한 중요성을 갖지 않음
- 디코더가 출력을 생성할 때 입력 시퀀스의 특정 부분에 "주목"
- 가중치를 통해 중요한 부분에 더 많은 주의를 기울임



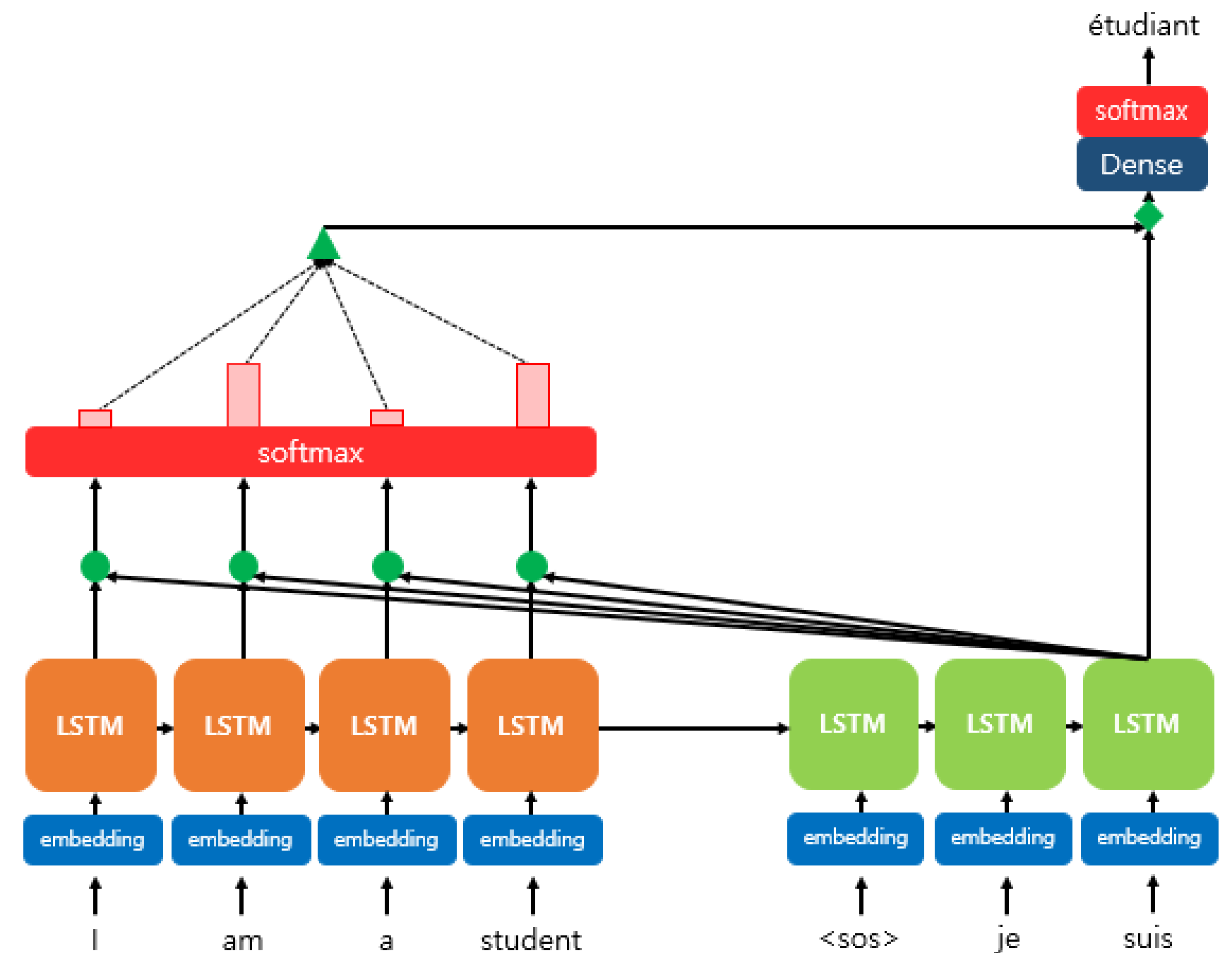
Key, Value

- Python 자료형 중 Dictionary에서는 데이터를 짝지어 저장
 - Key - Value
 - Key: 데이터를 호출하는 인덱스
 - Value: 호출한 인덱스의 실제 값
- 위 개념이 Attention에서 유사하게 사용
- Query: 질문
 - 디코더가 특정 단어를 생성하기 위해 인코더에 요구하는 참조



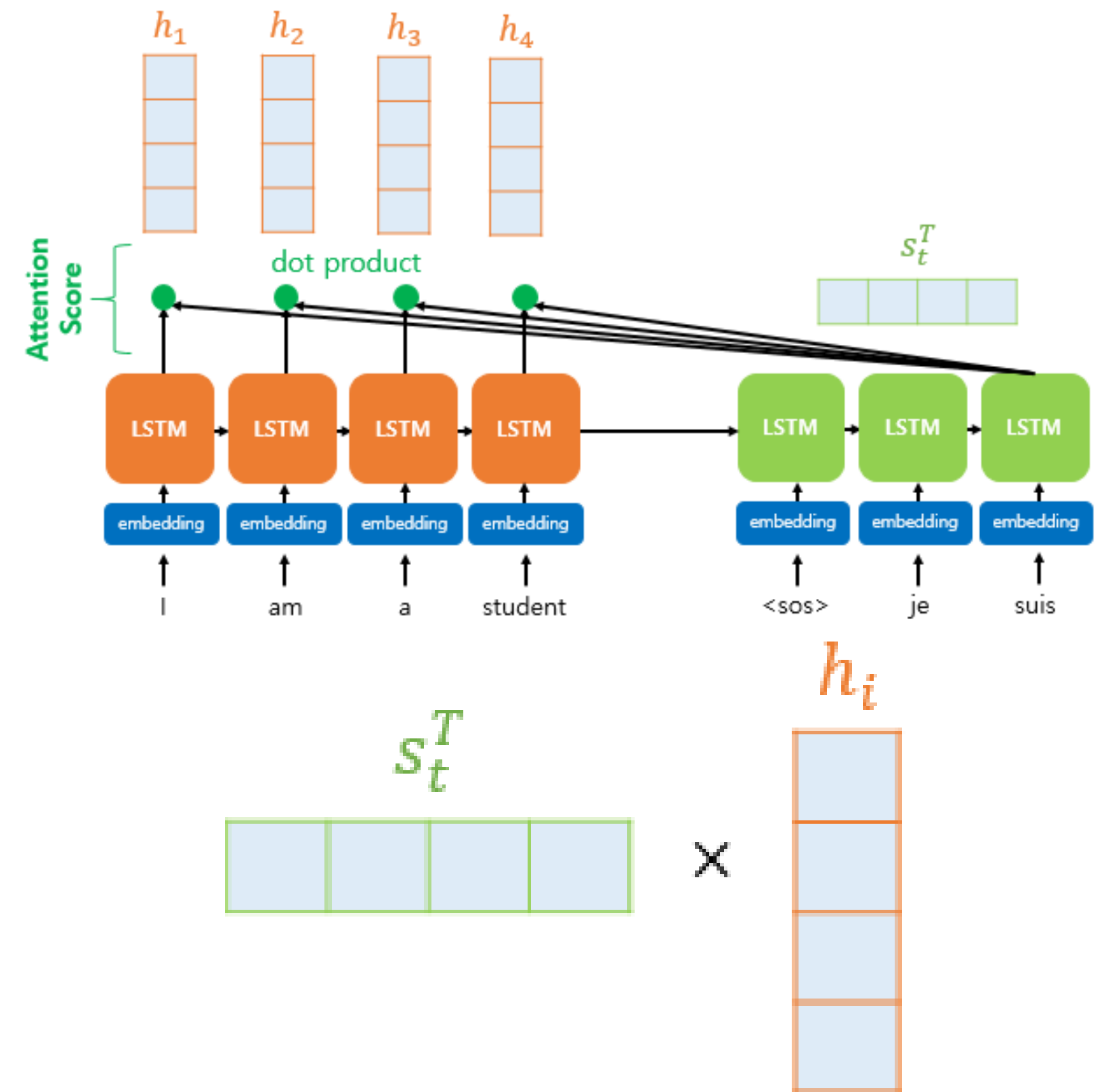
Attention의 작동 원리

- 인코더는 입력 문장의 단어를 읽고 내용을 학습
- 각 입력 단어에 대한 Hidden state(Key)가 생성됨
- 디코더는 문장을 구성하는 단어(Query) 생성 시, 입력된 여러 단어 Hidden state의 기여도(Value)를 참고함



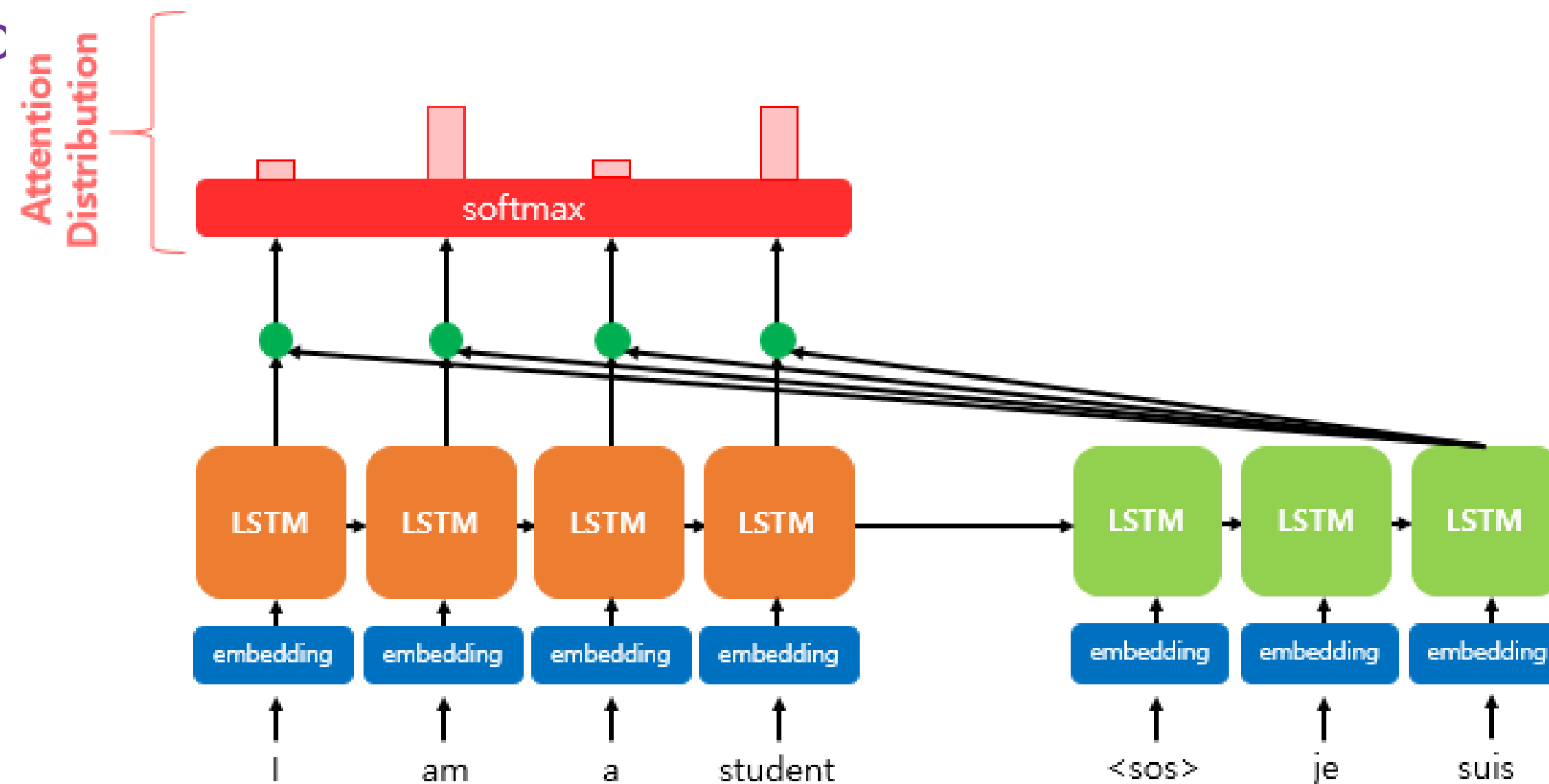
Attention Score

- 디코더에서 단어를 생성할 때, 인코더의 모든 Hidden state에 대해 각 요소가 디코더의 현재 Hidden state와 얼마나 유사한지를 측정한 점수 값
- 주로 내적을 통해 계산하며, 코사인 유사도 등 다른 연산을 사용하기도 함
 - Query와 Key의 내적 값



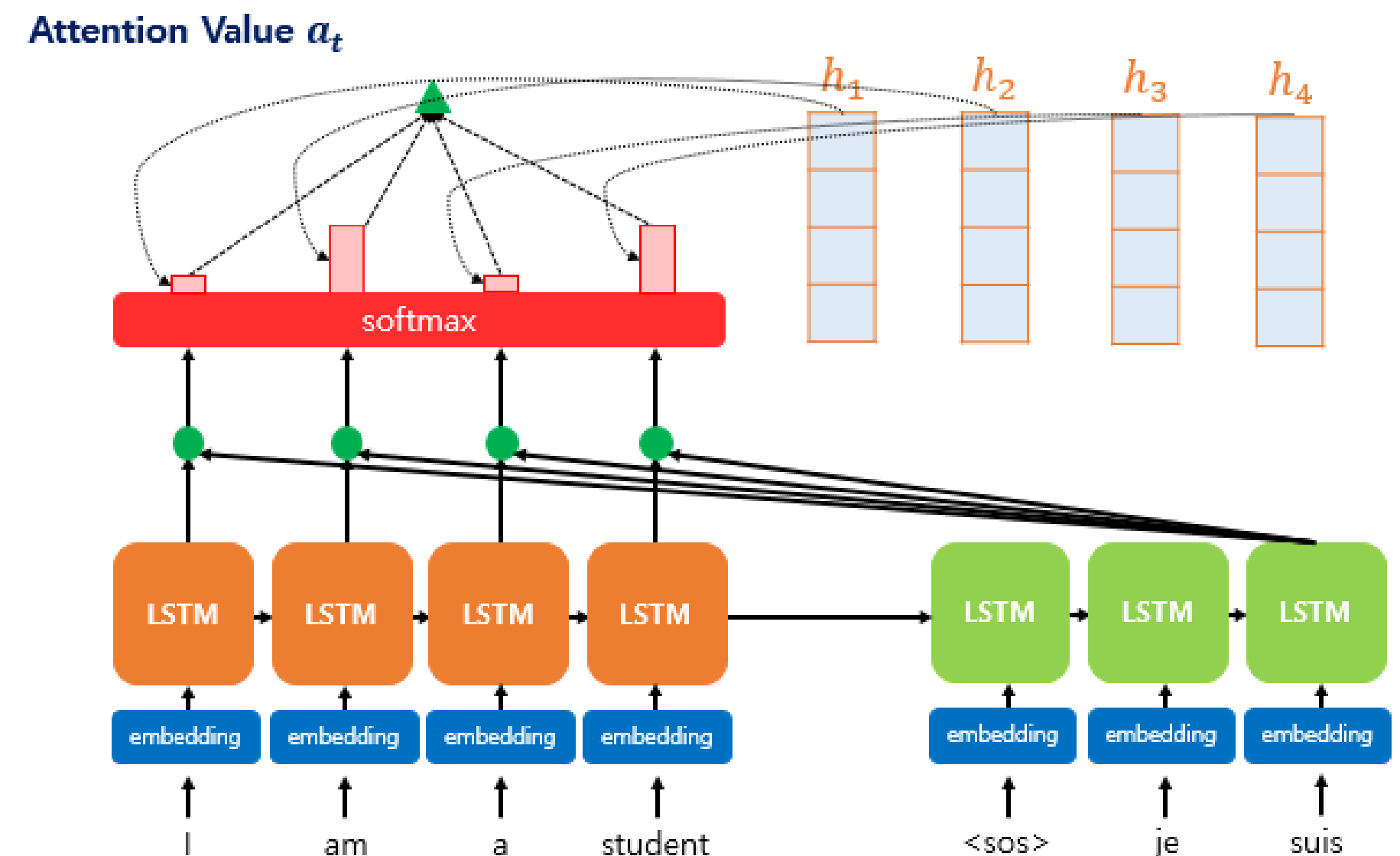
Attention Distribution

- 앞에서 구한 Attention score을 확률의 분포로 변환
 - Softmax 활용
 - 모든 인코더 Hidden state의 기여도가 확률 값으로 반환[0, 1]
- 각 기여도를 Attentive



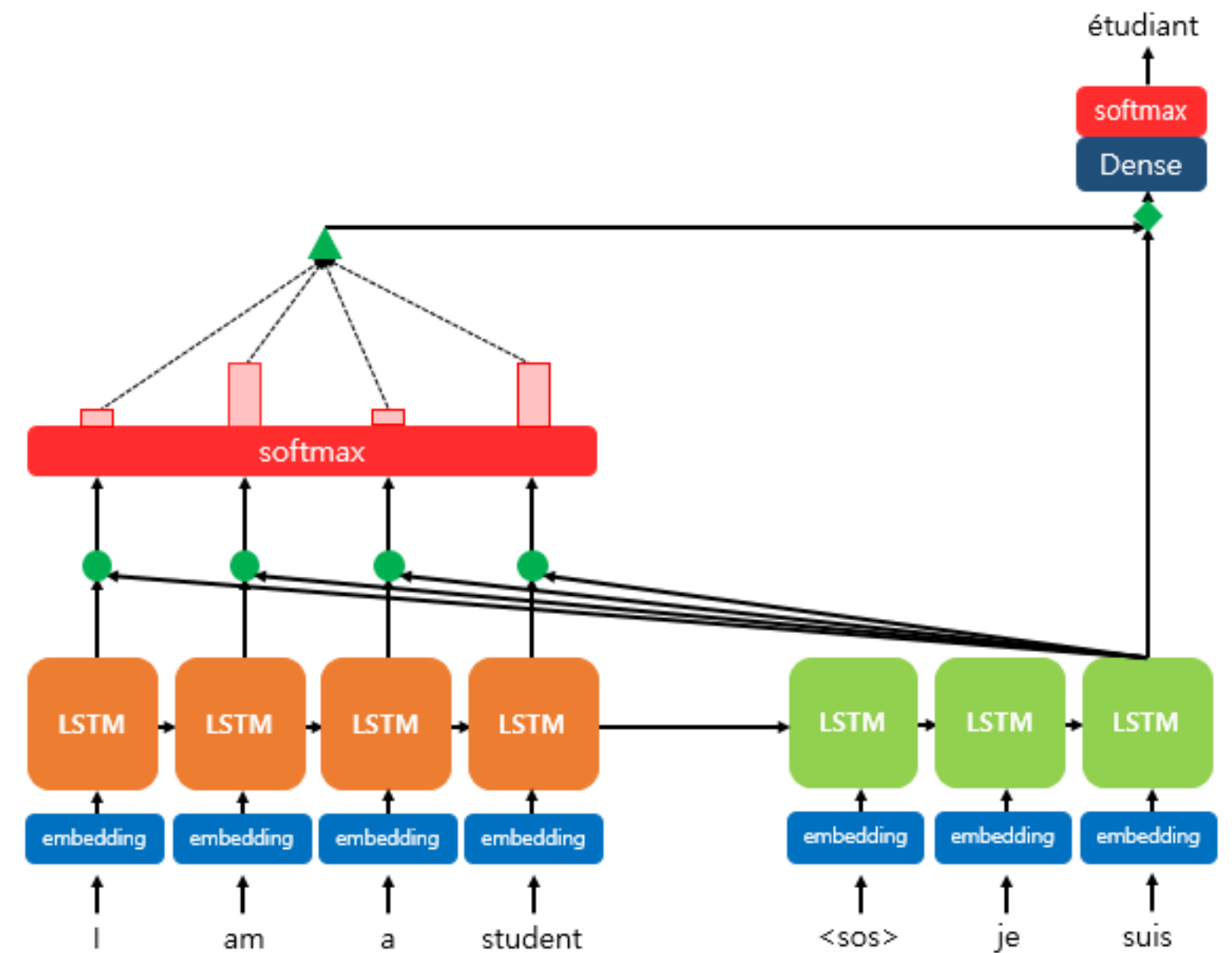
Attention Value

- 계산법
 - 각 인코더의 시점에서, 즉 매 단어마다
 - 각 단어의 Hidden state(Key)와
 - Attention distribution의 원소인 Attention weight(Value)를 곱한 후 모두 더함
- 의미
 - 디코더의 단어 생성 시 Hidden state 역할을 함
 - Context vector



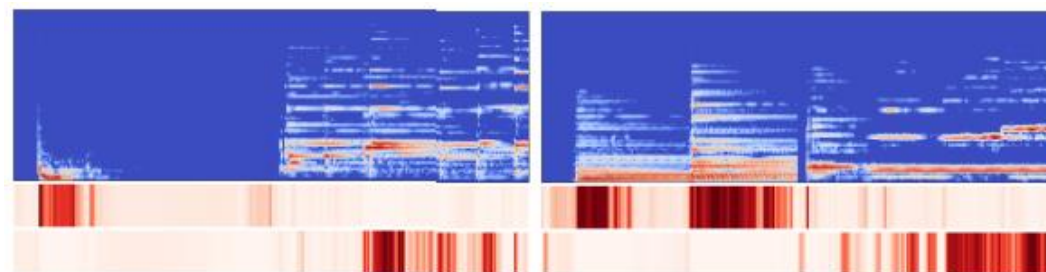
Attention의 결과

- 디코더는 주어진 컨텍스트 벡터(Attention Value)를 사용해 출력 토큰을 생성
- 이 과정은 디코더의 각 단어 생성 단계마다 반복
- 각 단계에서의 주목 대상이 다름



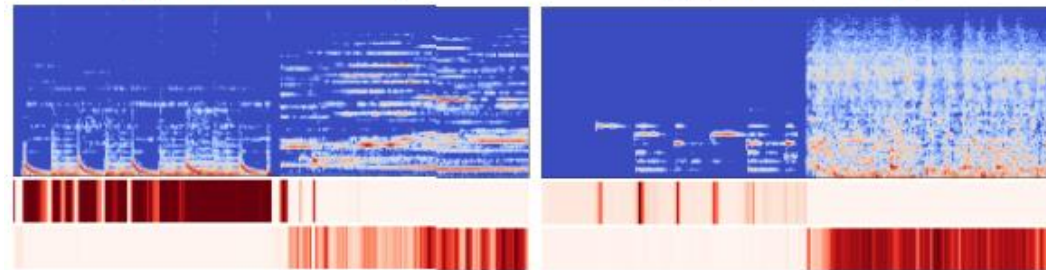
Attention Mechanism 응용 사례

- Attention을 이용한 Music tagging & Image captioning



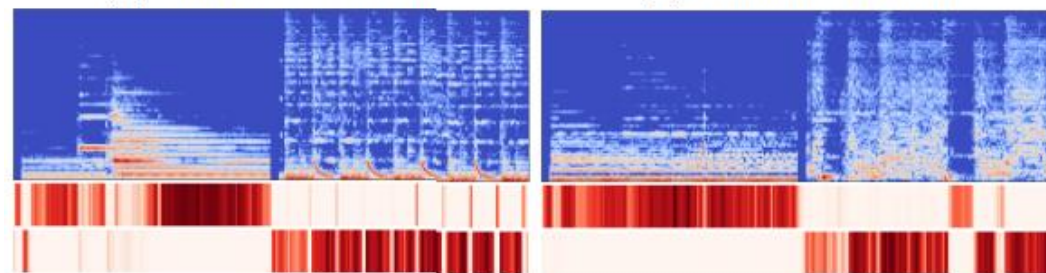
(c) Drums + Harp

(d) Piano + Flute



(e) Techno + Classic

(f) Classic + Metal



(g) Slow + Fast

(h) Quiet + Loud



A woman is throwing a frisbee in a park.



A dog is standing on a hardwood floor.



A stop sign is on a road with a mountain in the background.



A little girl sitting on a bed with a teddy bear.



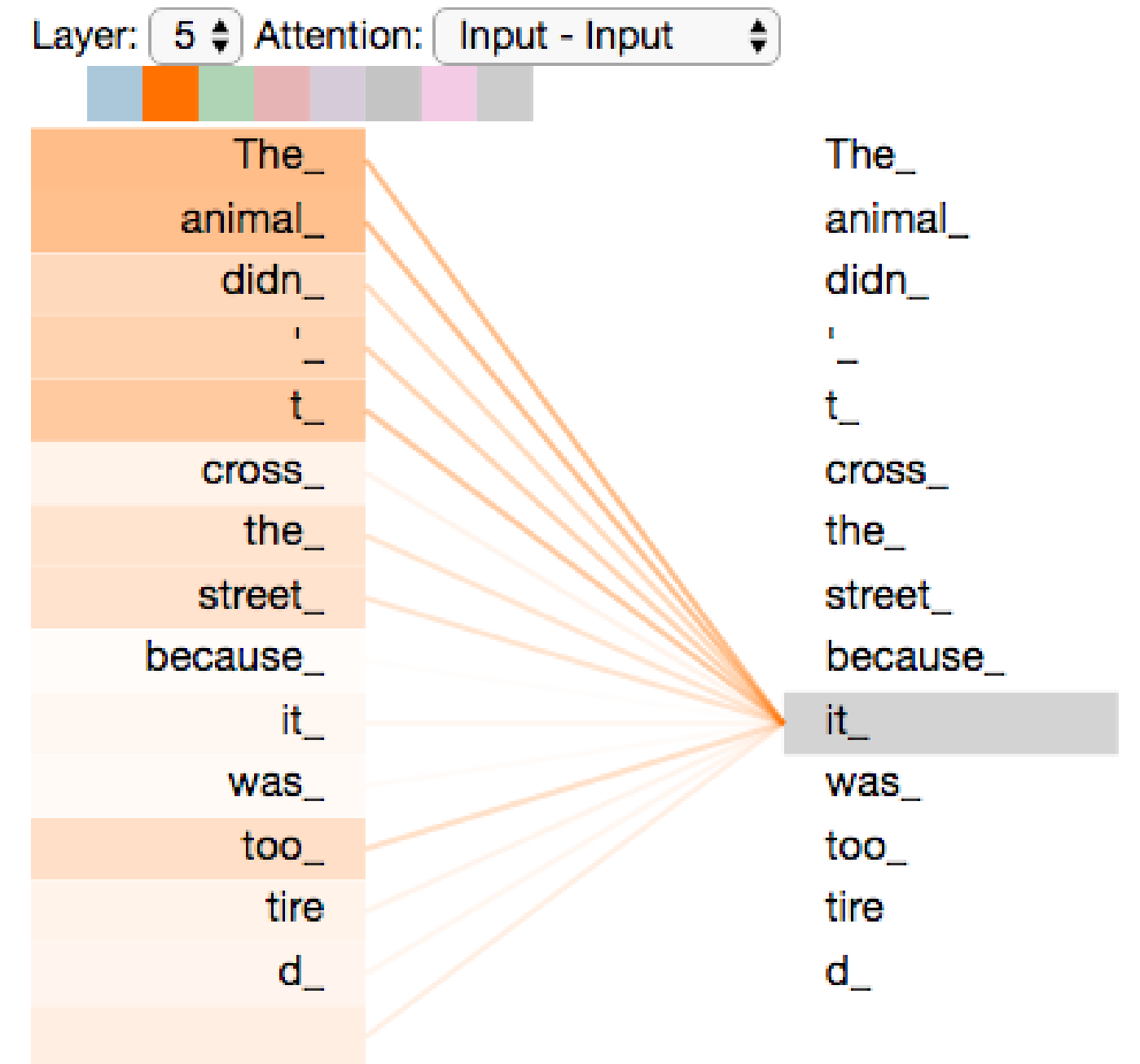
A group of people sitting on a boat in the water.



A giraffe standing in a forest with trees in the background.

Self-Attention

- Transformer의 핵심 원리
 - 시퀀스 내 모든 토큰 간의 관계 파악
 - Query, Key, Value의 개념을 사용하여 attention 점수 계산
 - 앞서 배운 Attention 개념과 유사
 - 각 토큰에 대한 가중치를 계산하여 새로운 표현 생성

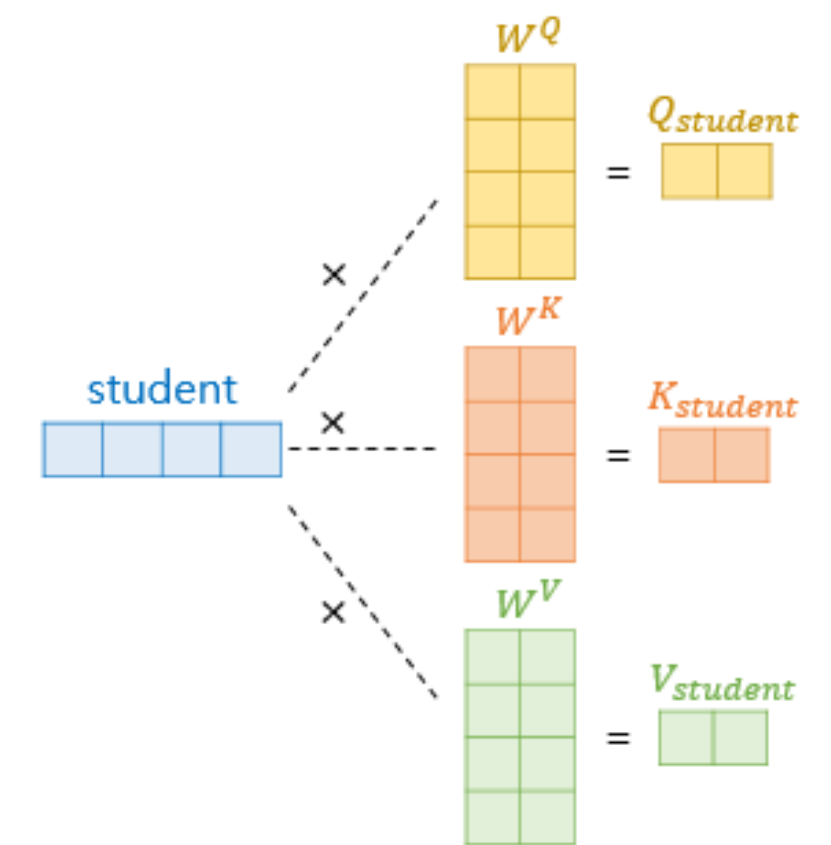
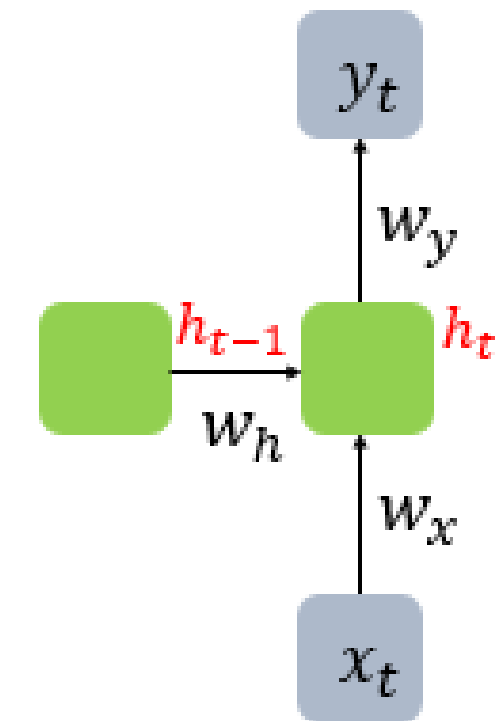


Self-Attention v.s. Attention

- 기존 Attention
 - Key와 Value는 인코더(입력 문장)의 Hidden state
 - Query는 디코더(출력 문장)의 Hidden state
 - 즉 영-프 번역기에서 K, V는 영어 문장의 단어, Q는 프랑스어 문장의 단어에서 유래
- Self attention
 - Q, K, V 모두 동일한 문장에서 유래
 - 인코더의 경우 동일한 영어 문장에서 Q, K, V 모두 계산
 - 디코더에서도 프랑스어 문장에서 Q, K, V 모두 계산

Self Attention의 가중치

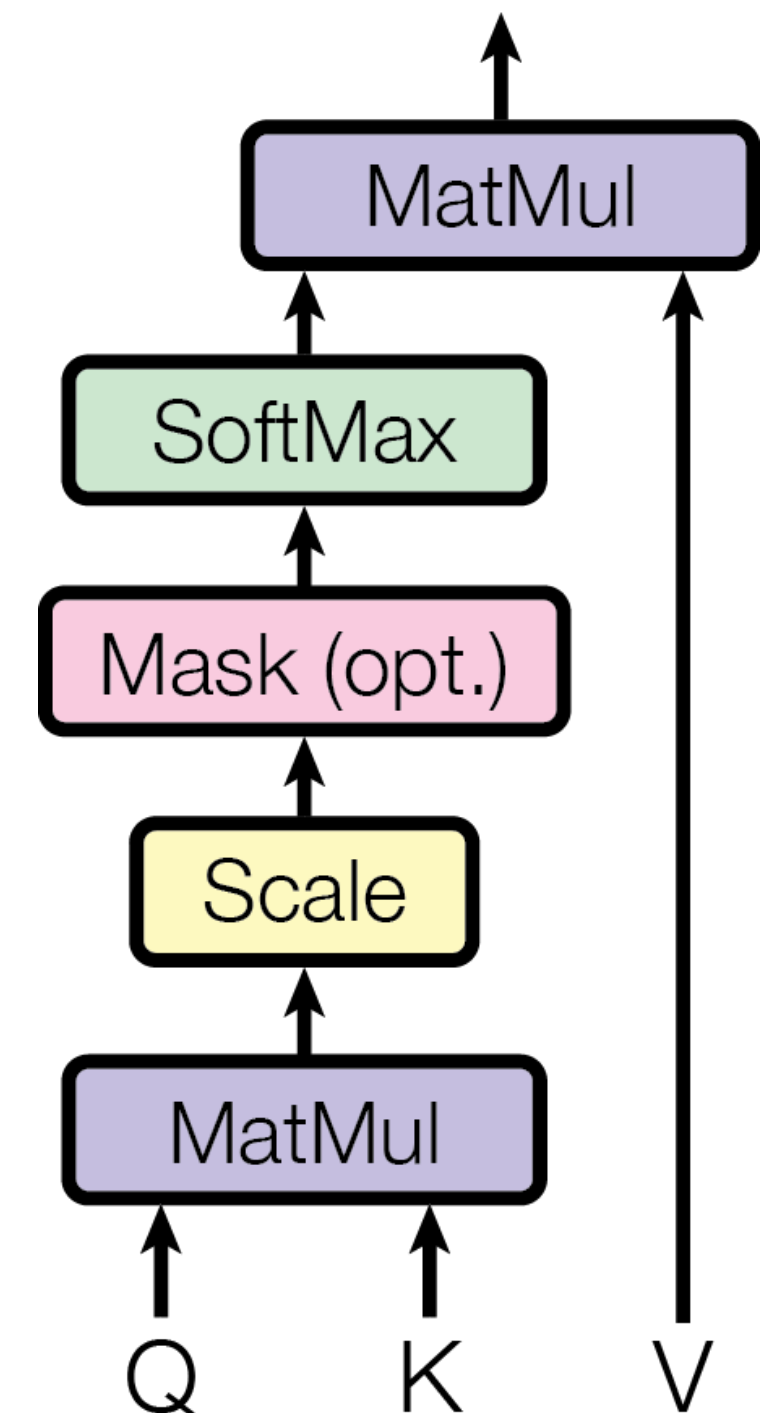
- RNN에서의 가중치:
 - 입력값을 읽어오는 가중치 W_x
 - 이전 시점을 받아 다음 시점으로 넘겨줄 때 연산되는 가중치 W_h
 - 경우에 따라 출력 값을 생성할 때 사용하는 가중치 W_y
- Self Attention에서의 가중치
 - 임베딩된 입력 문장에서 Q, K, V 를 생성하기 위한 행렬
 - WQ, WK, WV



Self Attention

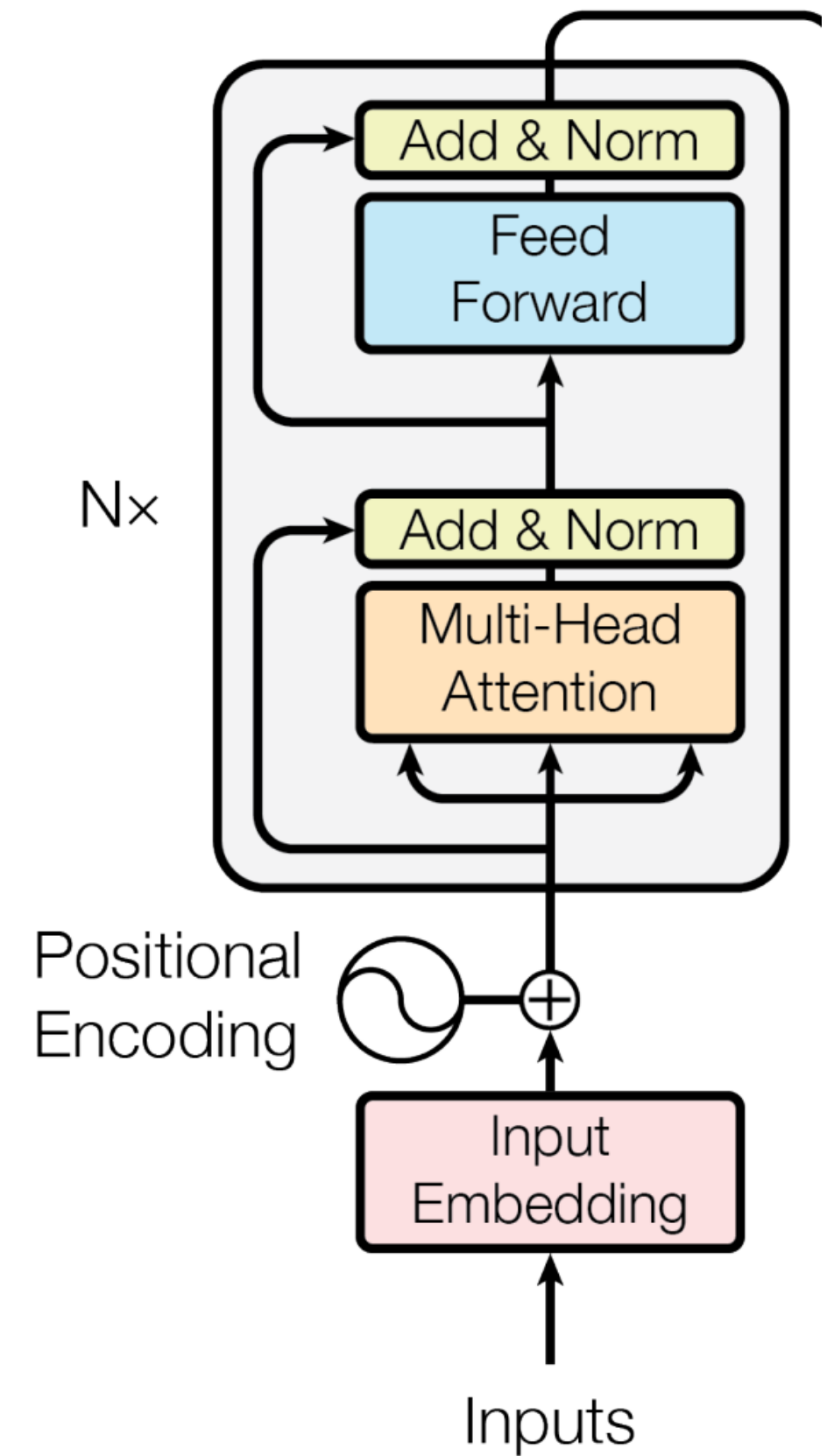
- 기존 Attention과 연산 방식에도 차이가 있음
 - Attention score
 - Q와 K의 내적은 동일하지만
 - K의 차원 수 제곱근으로 나누어 **Scaling** 진행
 - Attention value
 - 기존 Attention은 softmax의 결과와 V를 합했지만
 - Self attention에서는 이 둘을 **내적**함
 - 그런 이유로 **Scaled dot-product attention** 이라고도 함

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



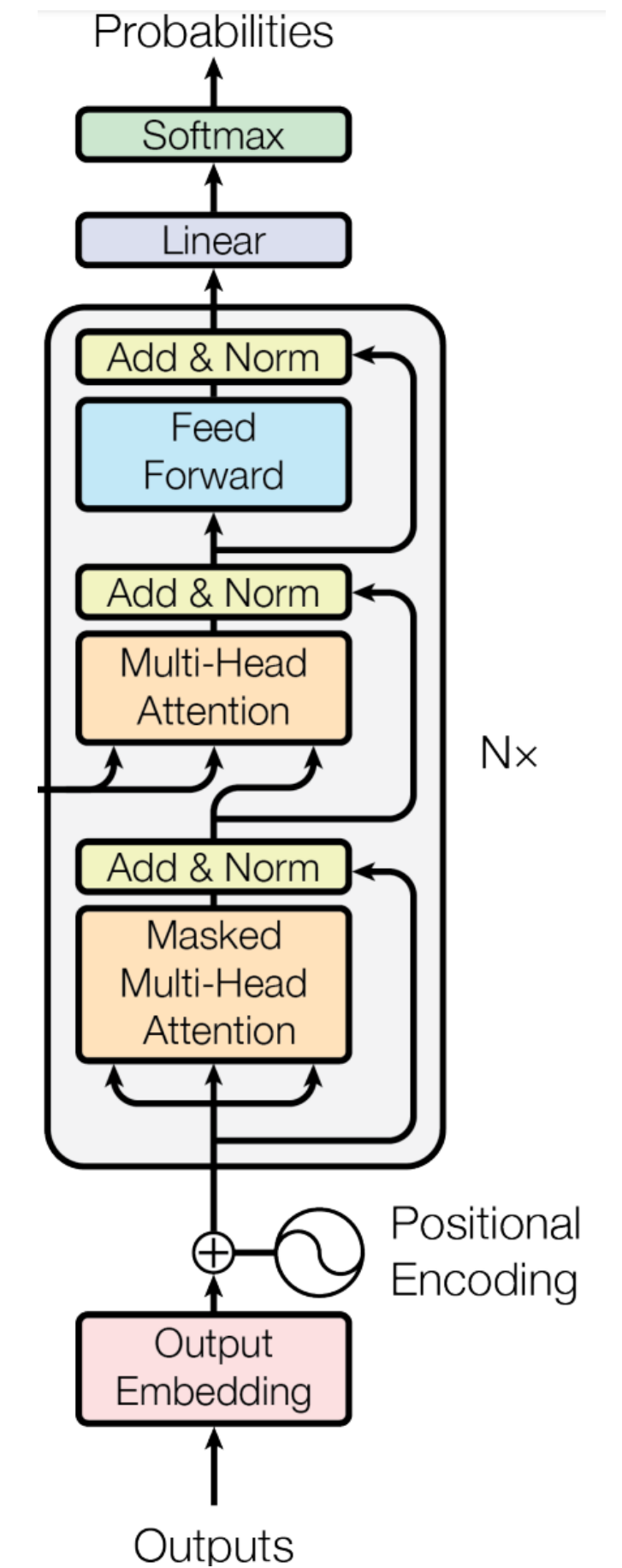
Multi-Head Attention(Encoder)

- 8개의 Head로 구성
- Self attention 연산을 통해 입력 문장의 정보를 학습
- 학습된 정보는 디코더의 Multi-Head Attention에 전달됨



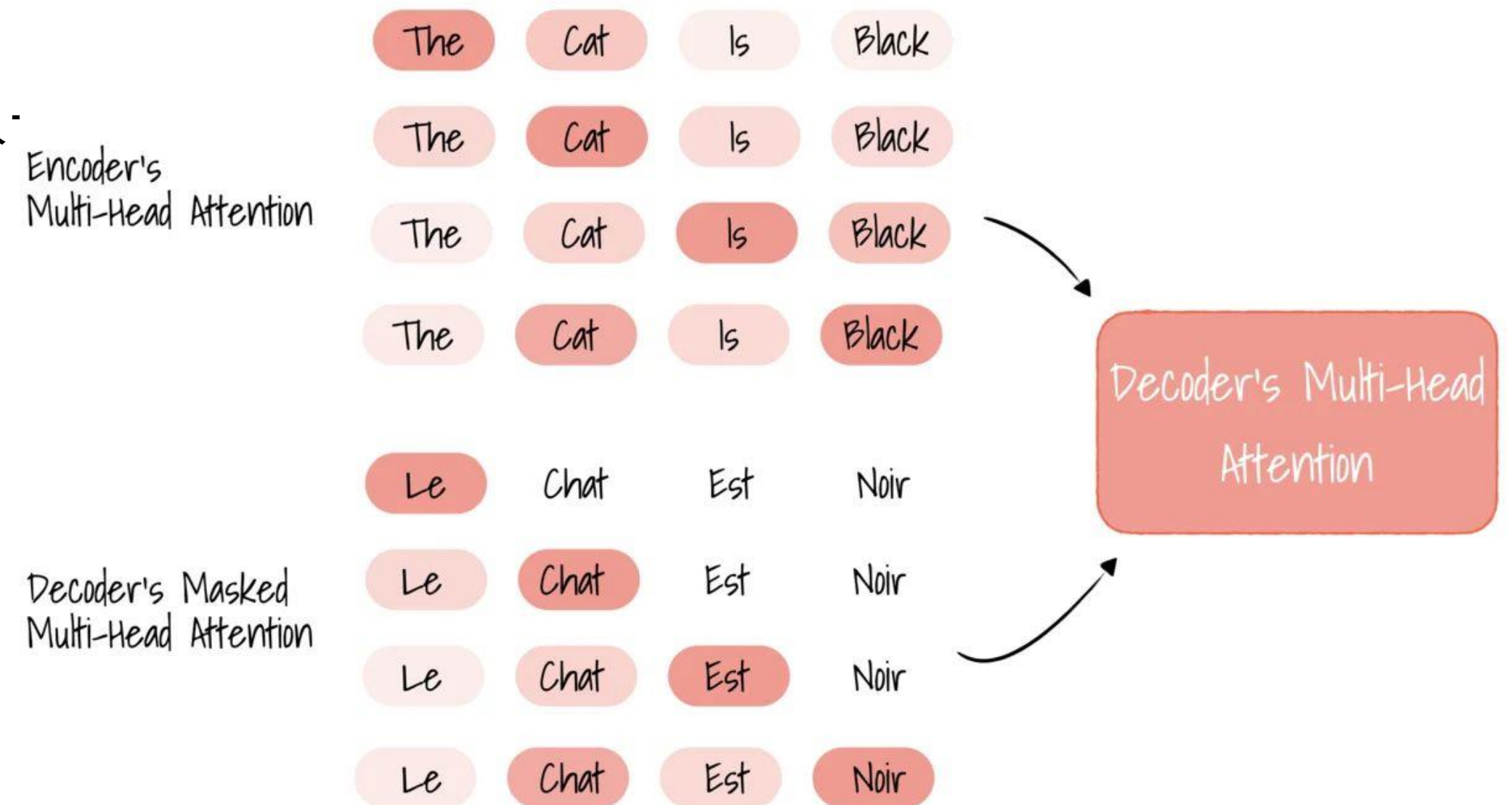
Masked Multi-Head Attention(Decoder)

- 디코더에만 존재하는 유일한 구조
- 디코더는 LM(Language Model)의 역할을 수행해야 함
- 즉 앞 단어를 바탕으로 뒷단어를 예측할 수 있어야 함
- 해당 기능을 구현할 수 있는 장치 필요



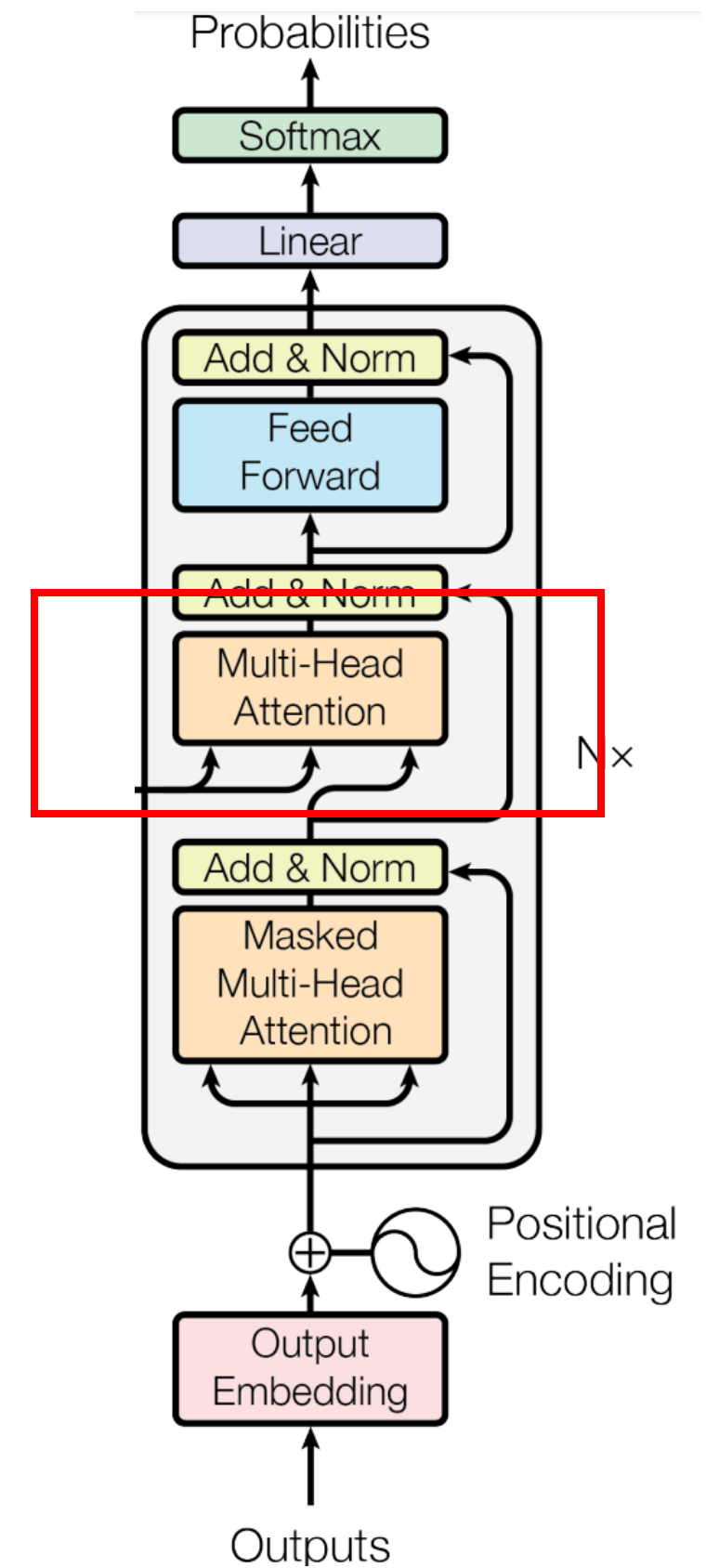
Masked Multi-Head Attention(Decoder)

- 디코더에서 문장을 입력받을 경우, Self attention 계산 시 문장의 뒷부분을 의도적으로 가림
- 문장의 제일 앞 단어부터 하나씩 마스크를 해제하며 Self attention 계산
- 결과적으로 단어 간 순서 정보를 학습할 수 있.



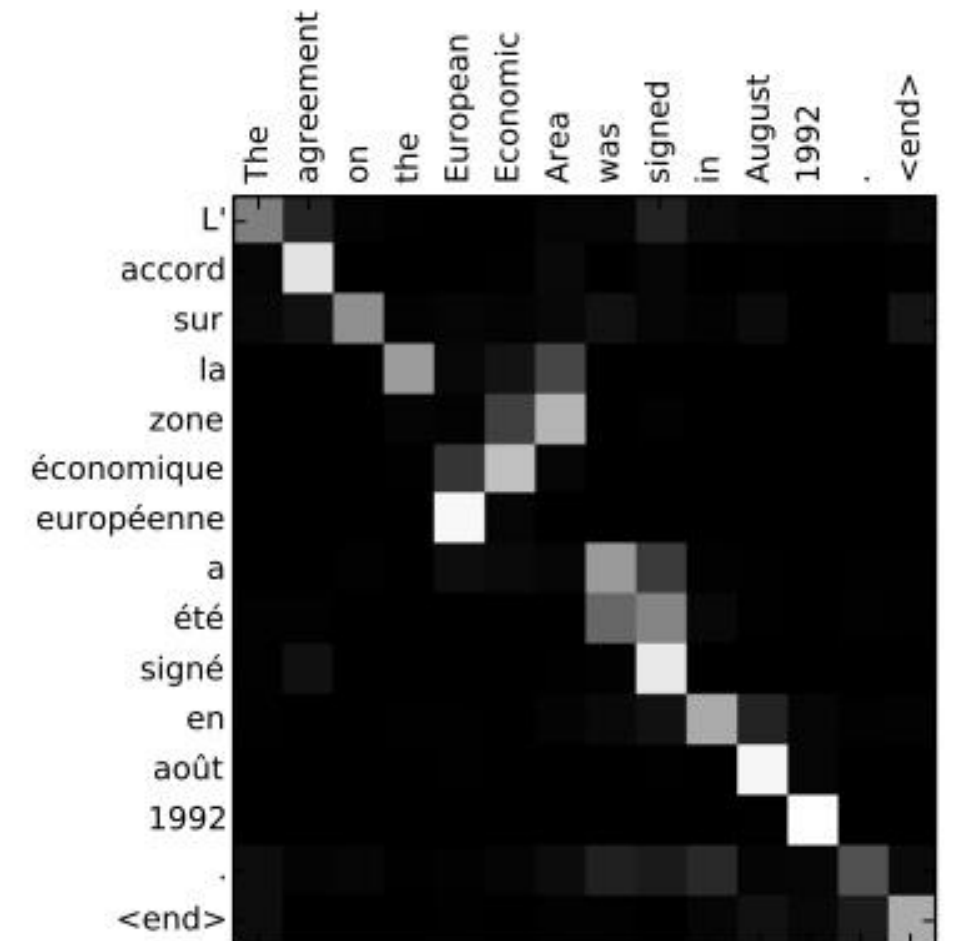
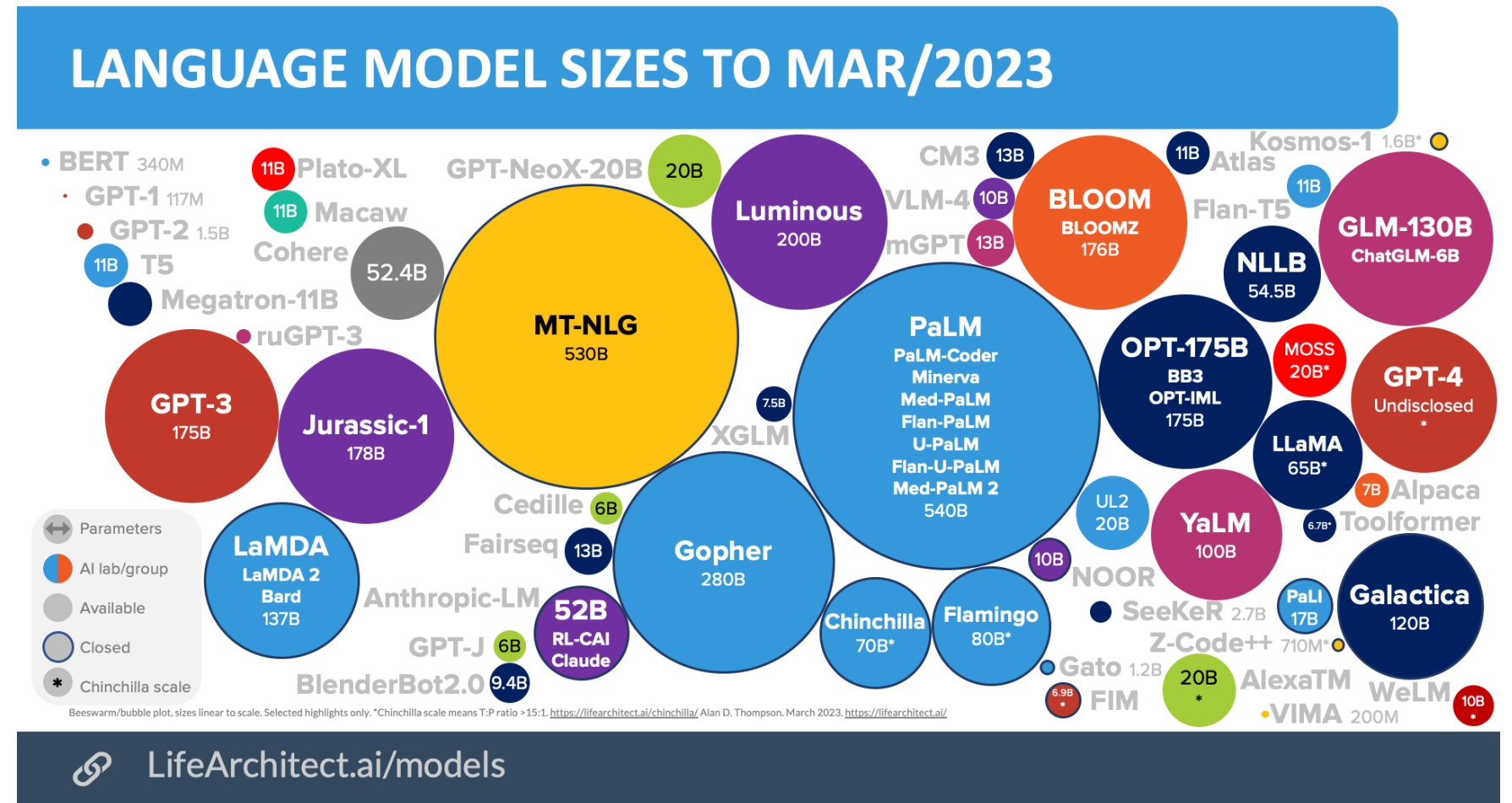
Multi-Head Attention(Decoder)

- 인코더의 Multi-head attention과 유사한 구조
 - 다만, 이 과정에서 K, V 는 인코더에서, Q 는 디코더에서 받음
- 사실상 이전 Attention 구조와 동일한 목적을 위해 연산
 - 디코더의 단어 생성(Q)을 위해
 - 인코더의 Hidden state(K, V)를 참조



Transformer의 영향

- 대규모 데이터로 병렬 학습
- 모델 크기에 비례하여 성능 향상
- 길이가 긴 문장도 잘 이해
- 하나의 모델이 여러 기능 수행
 - 다양한 문제를 해결하는 문장 생성
 - 다개국어 학습 가능
- 모델의 내부 연산 확인 가능



Transformer의 영향

- 거대 언어 모델(LLM)의 등장
 - 대규모 데이터와 컴퓨팅 파워의 결합
 - Ex) GPT-4, LLaMA
 - 일반적인 태스크에서 높은 성능, 다양한 응용 7

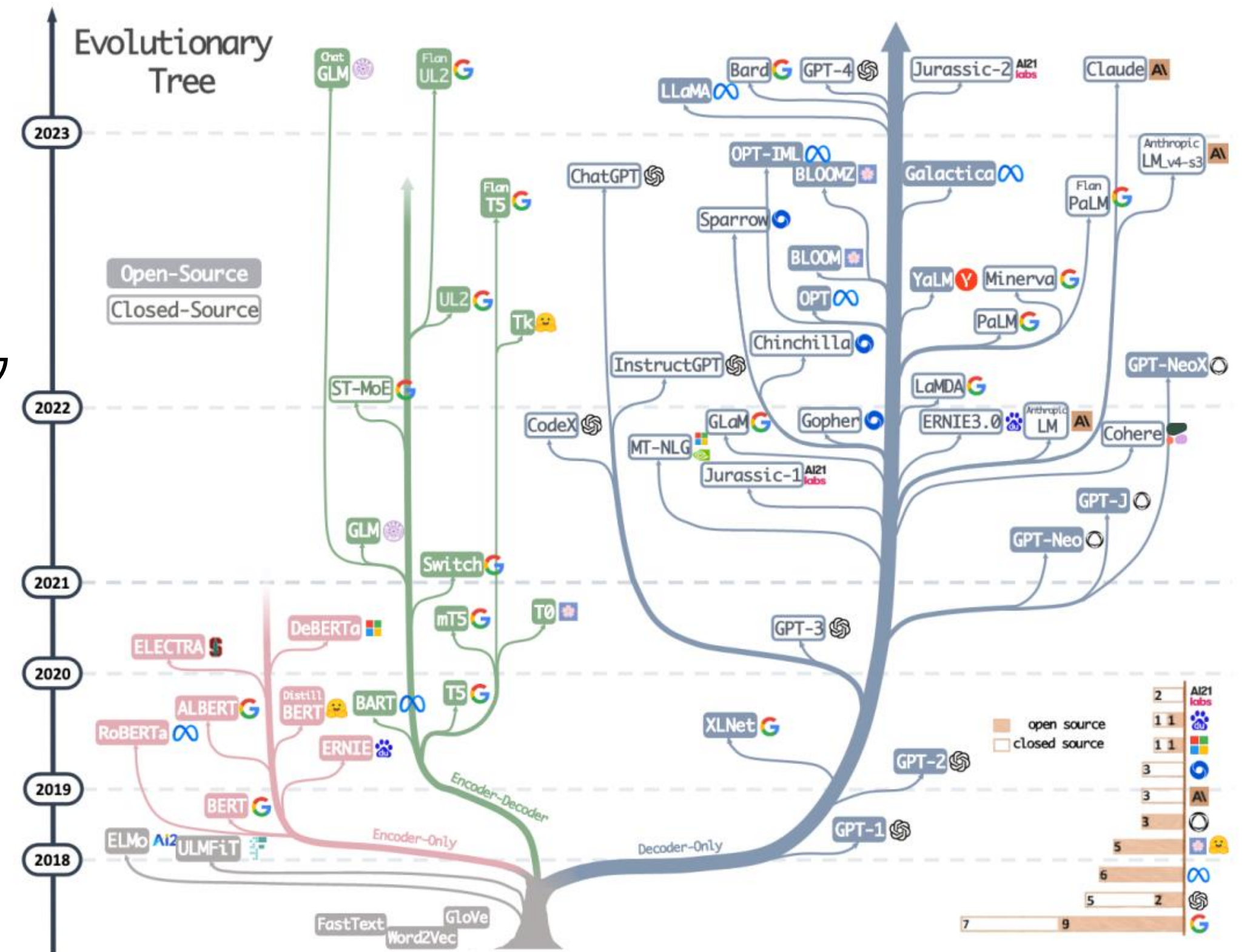
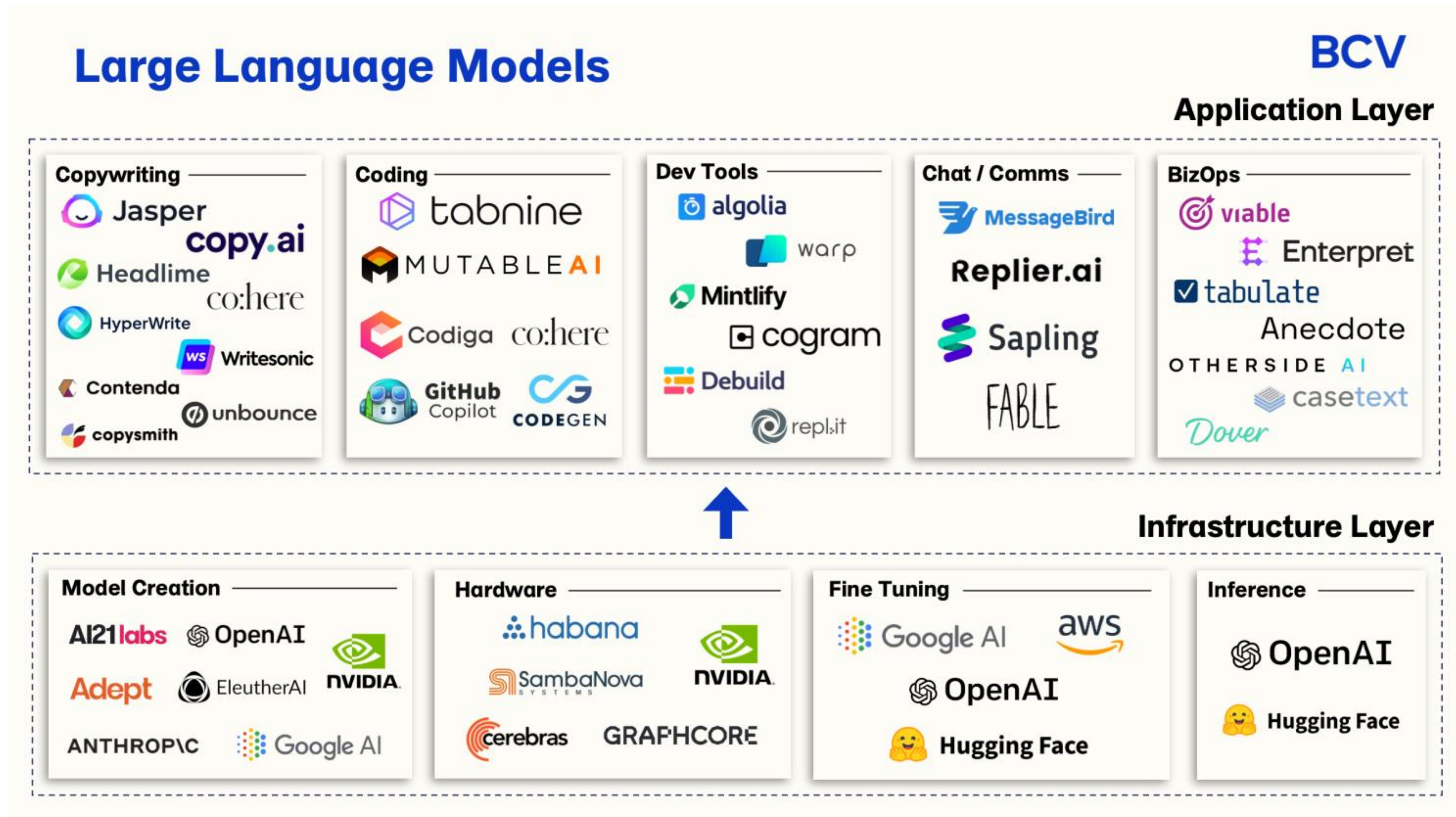
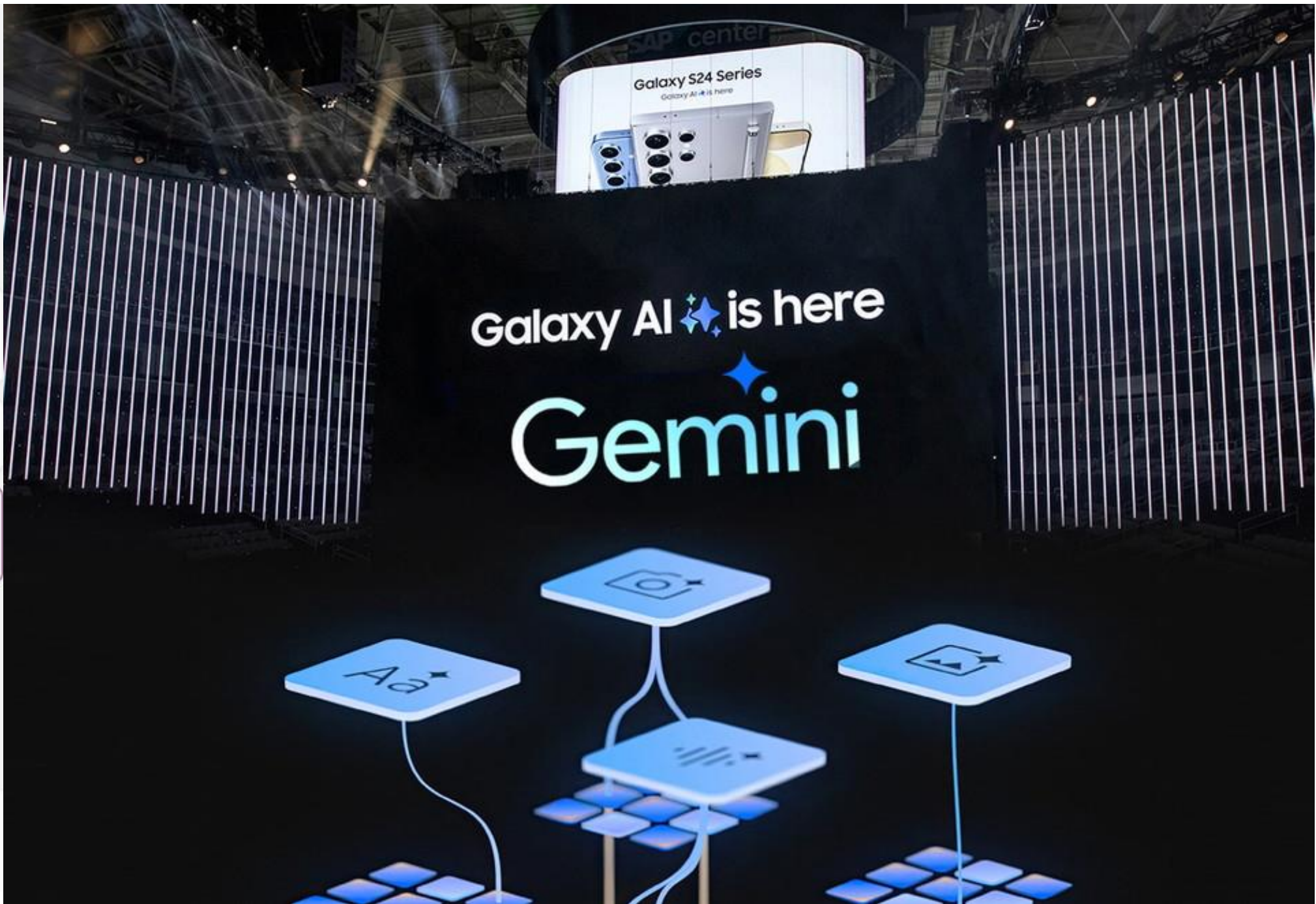
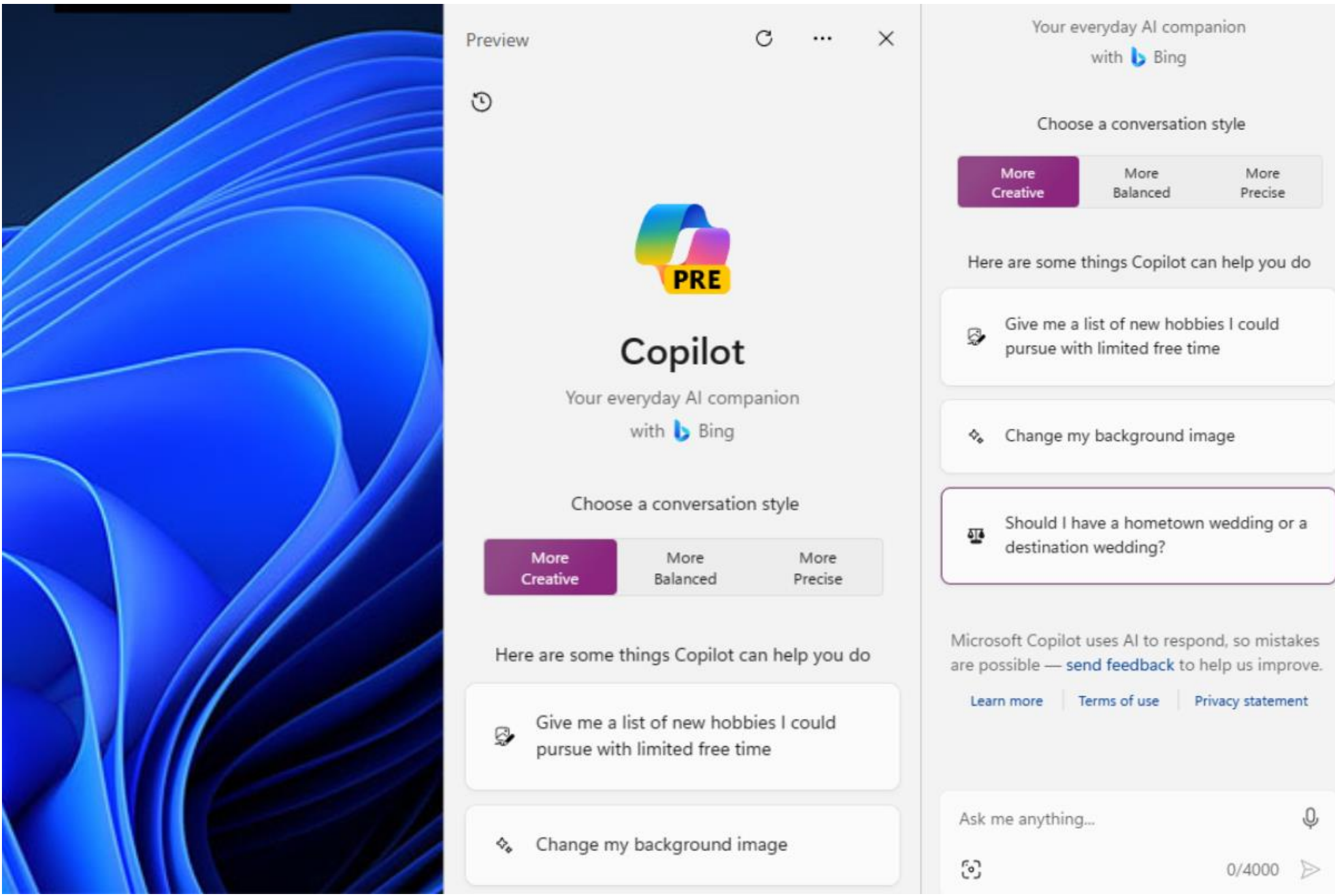


Fig. 1. The evolutionary tree of modern LLMs traces the development of language models in recent years and highlights some of the most well-known models. Models on the same branch have closer relationships. Transformer-based models are shown in non-grey colors: decoder-only models in the blue branch, encoder-only models in the pink branch, and encoder-decoder models in the green branch. The vertical position of the models on the timeline represents their release dates. Open-source models are represented by solid squares, while closed-source models are represented by hollow ones. The stacked bar plot in the bottom right corner shows the number of models from various companies and institutions.



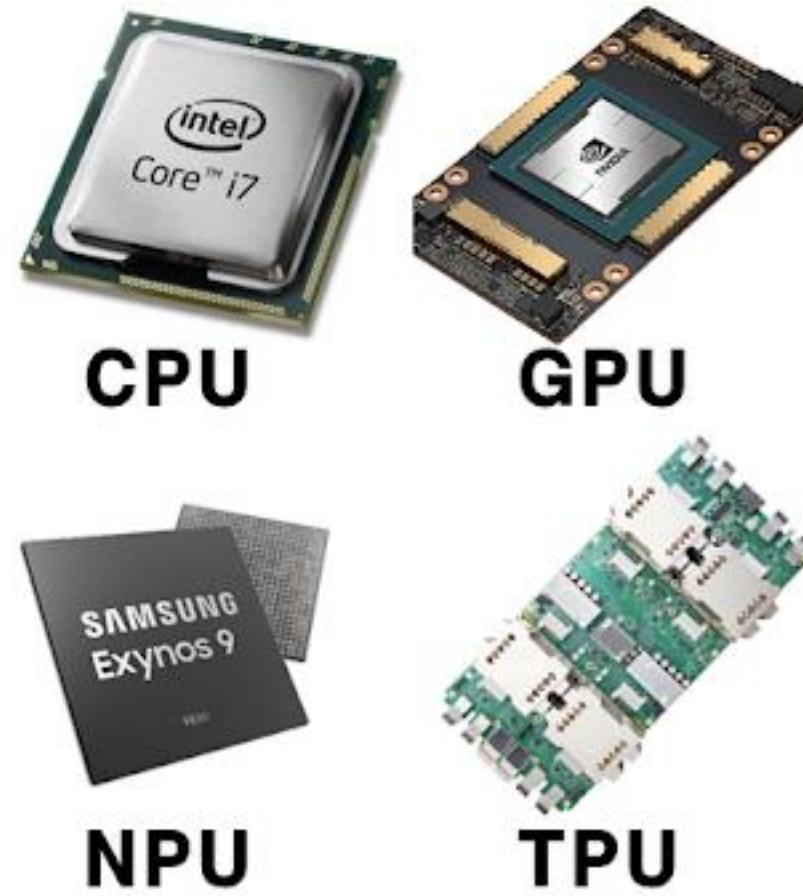
LLM 소프트웨어



LLM과 하드웨어

엔비디아 vs 애플 시가총액

(단위 : 조달러) ※ 현지시간 기준



연합뉴스

FT "위기의 삼성전자, SK하이닉스에 엔지니어 뺏길 위험"

FT "위기의 삼성전자, SK하이닉스에 엔지니어 뺏길 위험" · 엔지니어 "HBM은 SK하이닉스, 파운드리만 TSMC에 밀려 분위기 어둡다" · 전문가 "여러...

1개월 전

중앙일보

[단독] '마누라·자식 빼고 바꿔' 31살 삼성전자 위기...주 64시간 근무

"마누라와 자식 빼고 다 바꿔라"며 고(故) 이건희 삼성전자 전대회장이 '신경영'을 선언한 지 31주년이 되는 가운데, 삼성전자 내부에선 위...

2024. 6. 6.

한국경제

"위기 상황인 것 같다"...경고 쏟아진 삼성 반도체[황정수의 반도체 이슈 짚어보기]

위기 상황인 것 같다...경고 쏟아진 삼성 반도체, 기रो에 선 삼성전자 반도체 (1) 흔들리는 30년 1위 신화.

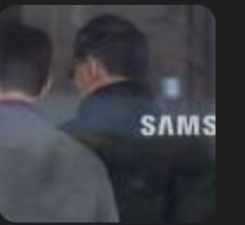
2024. 2. 3.

동아일보

[사설]삼성전자 창사 이래 첫 파업... '반도체 위기 탈출' 발목 잡나

삼성전자 최대 노동조합인 전국삼성전자노동조합(전삼노)이 어제부터 사흘간 총파업에 들어갔다. 삼성전에서 파업이 발생한 것은 1969년 창사 이래...

1개월 전

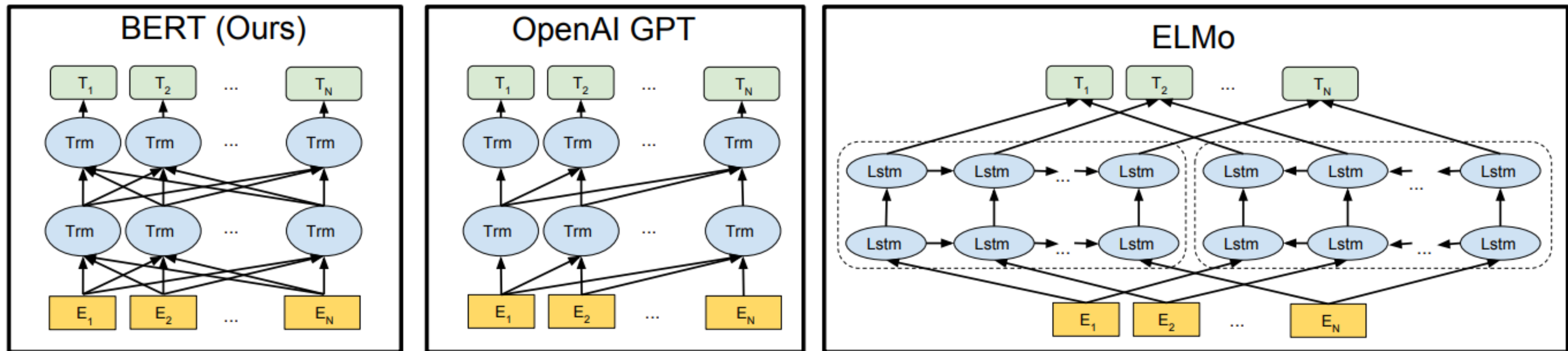


3. BERT

- Bidirectional Encoder Representation from Transformers
 - Transformer의 인코더 구조를 기반으로 한 모델
 - 양방향으로 문맥을 고려하여 토큰을 인코딩
 - 주어진 텍스트에서 일부 토큰을 마스킹하고 이를 예측하는 방식으로 사전 학습
 - QA, 문장 분류 등의 다양한 NLP 문제에서 SOTA 달성
 - 일반적으로 분류 문제에 사용(적은 출력 값의 수)
 - 이어질 LLM 분야의 패러다임을 바꿈

BERT의 특징 – 양방향 문맥 인식

- 양방향 문맥 인식
 - 이전 모델들은 주로 단방향(왼쪽에서 오른쪽 또는 그 반대) 문맥만 고려
 - BERT는 양쪽 방향의 문맥을 동시에 고려하여 더 정확한 토큰 표현 생성



BERT 이전 모델의 학습 문제

- 지도 학습
 - 데이터(X)와 이에 대한 레이블(y_true)를 제공
 - 데이터를 통해 모델이 예측한 값(y_pred)과 정답 간 차이를 계산
 - 차이가 수치화되므로 모델이 어느 방향으로 학습해야할 지 지시할 수 있음
 - 그러나 지도학습 데이터를 만드는 것은 매우 고된 일
 - 노동력
 - 비용
 - 시간
 - 텍스트 데이터의 경우 사실상 불가능

BERT 이전 모델의 학습 문제

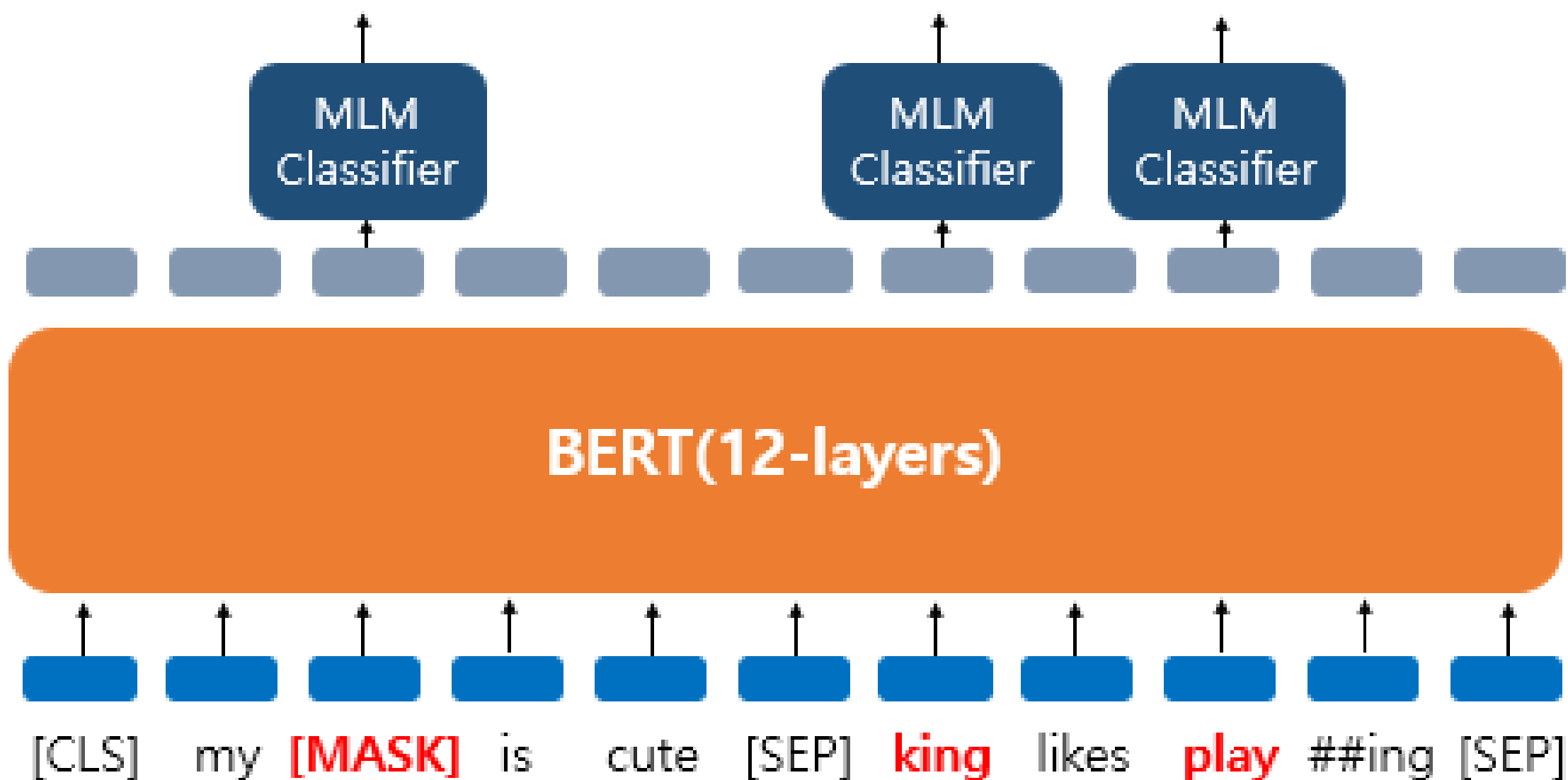
- 비지도 학습
 - 데이터만으로 모델을 학습
 - 비용이 매우 저렴
 - 방법이 매우 모호하며 추상적
 - 모델의 학습 방향을 제시하기 어려움
 - BERT 이전 자연어처리 모델은 비지도 학습을 통해 의미있는 결과를 만들어내지 못함

BERT의 특징 – 사전 학습(Pretraining)

- BERT에는 무한한 양의 지도학습 데이터가 투입
 - 지도학습의 형태로 비지도학습을 수행
 - 즉 레이블이 없는 데이터를 이용하여 지도학습을 수행
- 두 가지 방식의 사전 학습이 진행
 - MLM(Masked Language Model)
 - NSP(Next Sentence Prediction)

BERT의 특징 – 사전 학습(Pretraining)

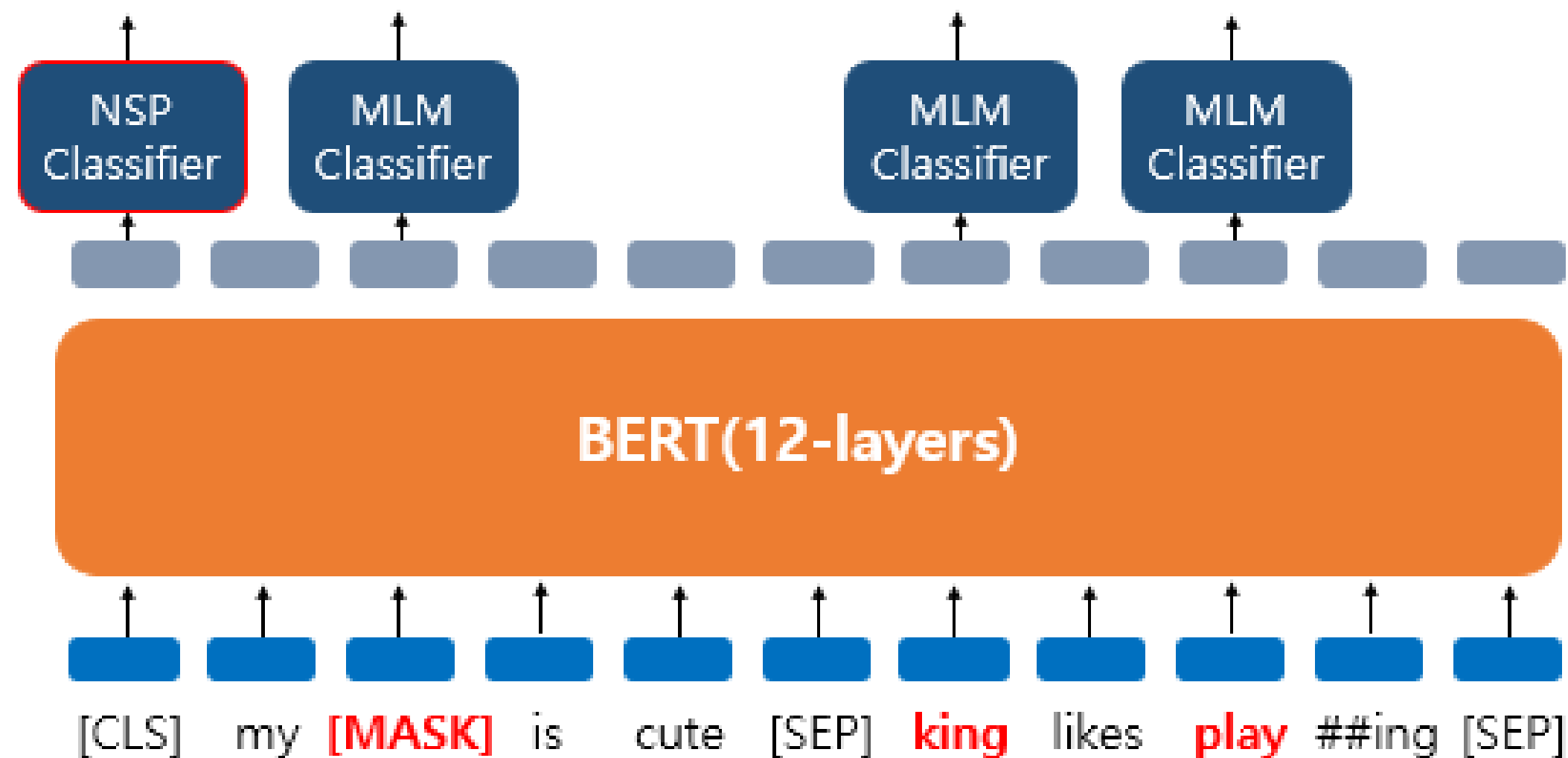
- BERT의 사전학습 - MLM(Masked Language Modelling)
 - 일부(15%) 토큰을 임의로 마스킹하고 해당 토큰을 예측하는 방식으로 학습
 - 이 과정을 통해 모델은 문맥을 기반으로 한 토큰의 의미를 깊게 이해



I am eating an ice cream.
I **am eating an** ice cream.
I am **eating an ice** cream.
I am eating **an ice cream**.
The boy is drinking milk.
The boy is drinking milk.
The **boy is drinking** milk.
The boy **is drinking** milk.

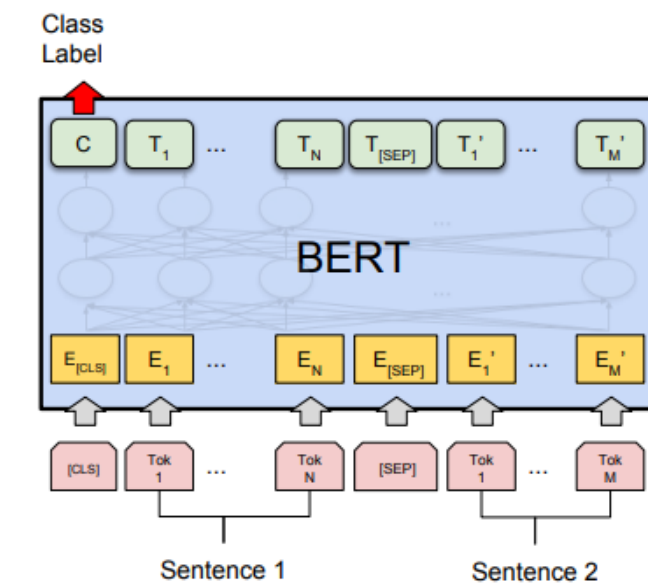
BERT의 특징 – 사전 학습(Pretraining)

- BERT의 사전학습 – NSP(Next Sentence Prediction)
 - 두 개의 문장이 붙은 상태로 모델에 입력됨
 - 문장을 구분하기 위해 [SEP] 토큰을 구분자로 사용
 - 이진 분류를 통해, 두 문장의 연속성 여부를 학습함
 - [CLS]토큰의 위치에서 이진 분류 문제를 해결

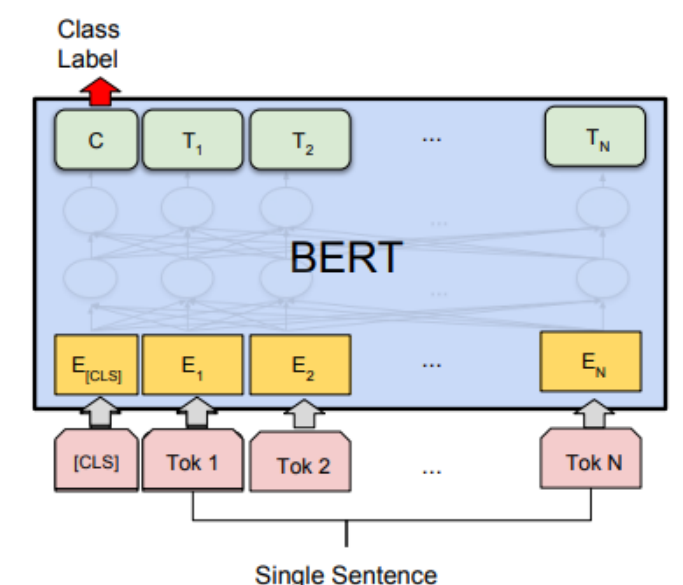


BERT의 특징 – 전이 학습

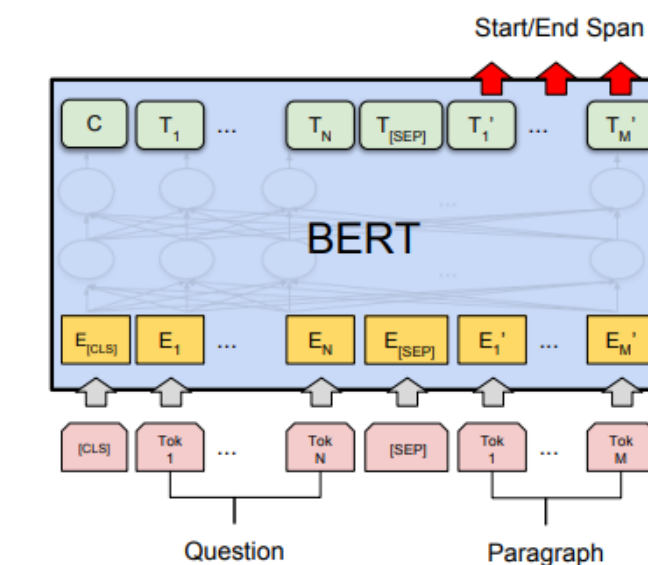
- 전이 학습(Transfer learning and Fine-tuning)
 - BERT는 큰 텍스트 코퍼스에서 사전 학습 후, 특정 작업에 미세 조정 가능
 - Wikipedia(2.5B), BooksCorpus(8M)
 - 11가지 NLP task에서 SOTA 달성
 - 각 Task마다 모델 말단의 구조만 다름
 - 대규모 텍스트로 사전학습된 모델을 Foundation model이라 함



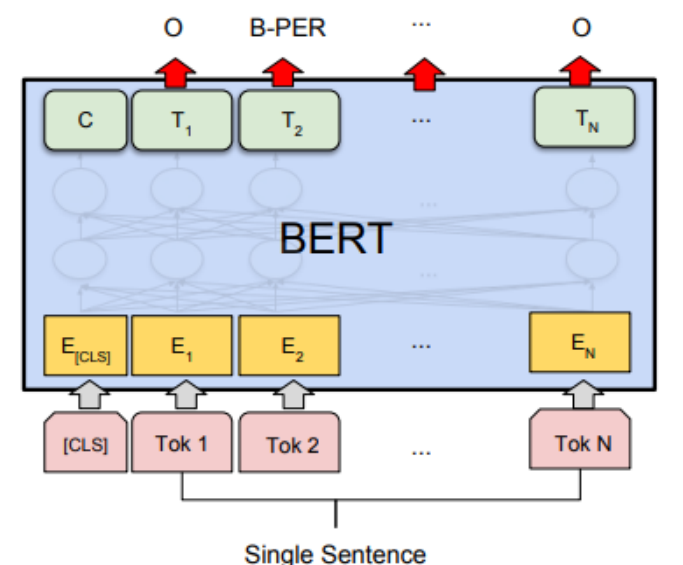
(a) Sentence Pair Classification Tasks:
MNLI, QQP, QNLI, STS-B, MRPC,
RTE, SWAG



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



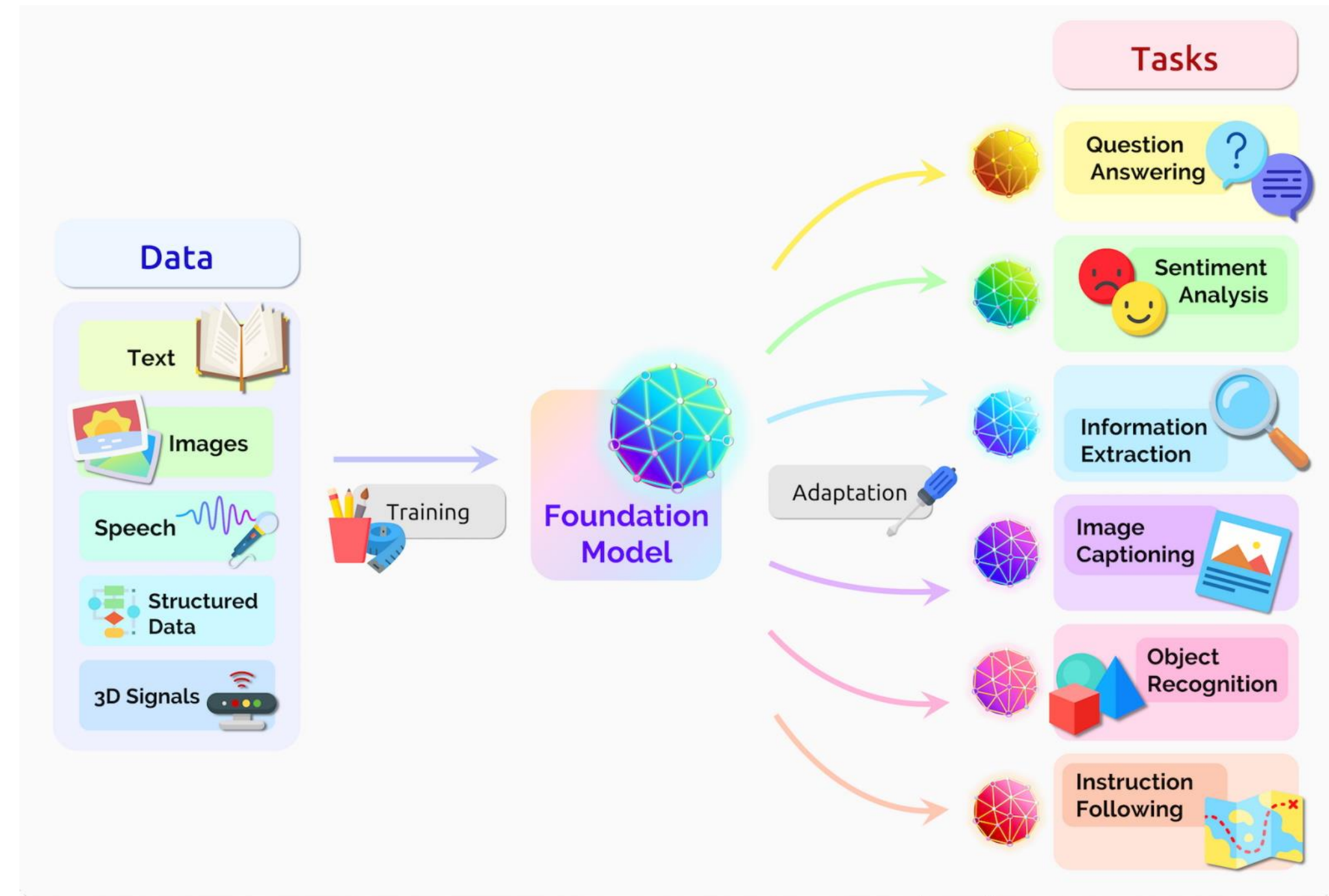
(c) Question Answering Tasks:
SQuAD v1.1



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

BERT의 특징 – 사전 학습(Pretraining)

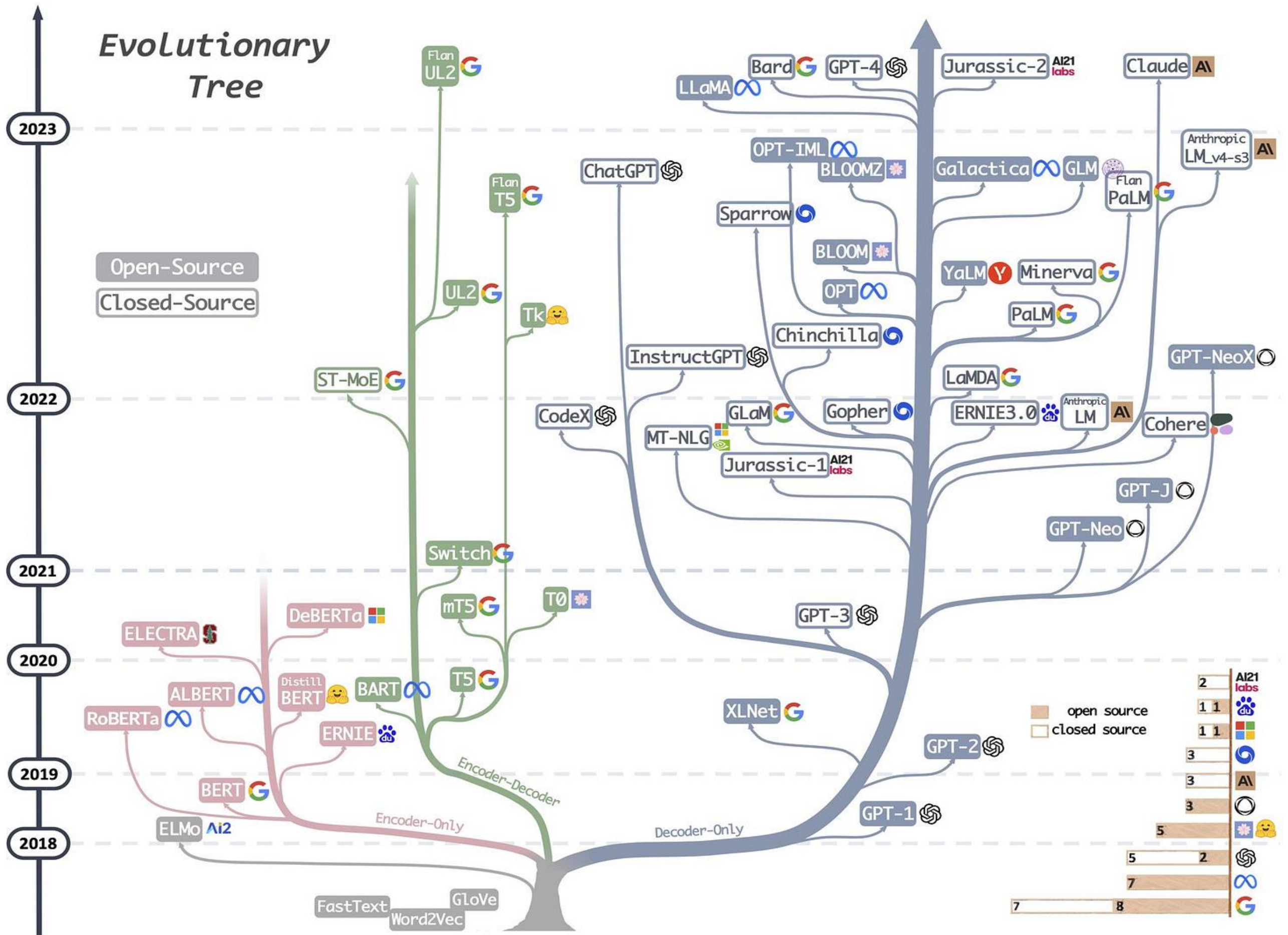
- Foundation 모델
 - 과거에는 1작업 1모델
 - BERT 이후 모델 하나가 여러 작업을 담당하도록 훈련
 - 비용 부담 감소 및 상용화



4. LLM

LLM의 구분

- 구조에 따라
 - Encoder-Decoder
 - Encoder only
 - Decoder only
- 소스 코드 오픈 여부에 따라
 - 오픈소스 모델
 - 사유화 모델



BERT 개선 및 확장

- 모델의 확장과 변형
 - RoBERTa, DistilBERT, ALBERT 등 BERT의 변형 모델 등장
 - 학습 방법, 모델 크기, 아키텍처의 변화를 통한 성능 향상 및 최적화

Comparison	BERT October 11, 2018	RoBERTa July 26, 2019	DistilBERT October 2, 2019	ALBERT September 26, 2019
Parameters	Base: 110M Large: 340M	Base: 125 Large: 355	Base: 66	Base: 12M Large: 18M
Layers / Hidden Dimensions / Self-Attention Heads	Base: 12 / 768 / 12 Large: 24 / 1024 / 16	Base: 12 / 768 / 12 Large: 24 / 1024 / 16	Base: 6 / 768 / 12	Base: 12 / 768 / 12 Large: 24 / 1024 / 16
Training Time	Base: 8 x V100 x 12d Large: 280 x V100 x 1d	1024 x V100 x 1 day (4-5x more than BERT)	Base: 8 x V100 x 3.5d (4 times less than BERT)	[not given] Large: 1.7x faster
Performance	Outperforming SOTA in Oct 2018	88.5 on GLUE	97% of BERT-base's performance on GLUE	89.4 on GLUE
Pre-Training Data	BooksCorpus + English Wikipedia = 16 GB	BERT + CCNews + OpenWebText + Stories = 160 GB	BooksCorpus + English Wikipedia = 16 GB	BooksCorpus + English Wikipedia = 16 GB
Method	Bidirectional Transformer, MLM & NSP	BERT without NSP, Using Dynamic Masking	BERT Distillation	BERT with reduced parameters & SOP (not NSP)

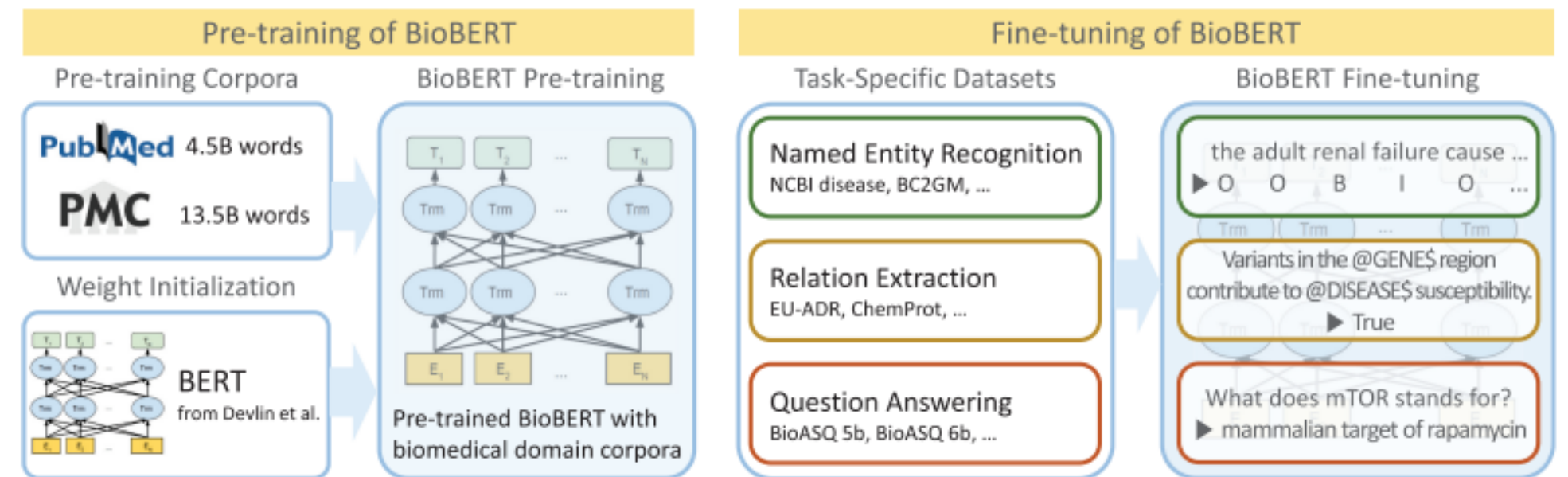
BERT 개선 및 확장

- 다양한 언어와 도메인에 적용
 - 다양한 언어의 BERT (예: KoBERT, MultiLingual BERT)
 - 특정 도메인에 최적화된 BERT (예: BioBERT, SciBERT)

SK텔레콤 언어처리 AI기술

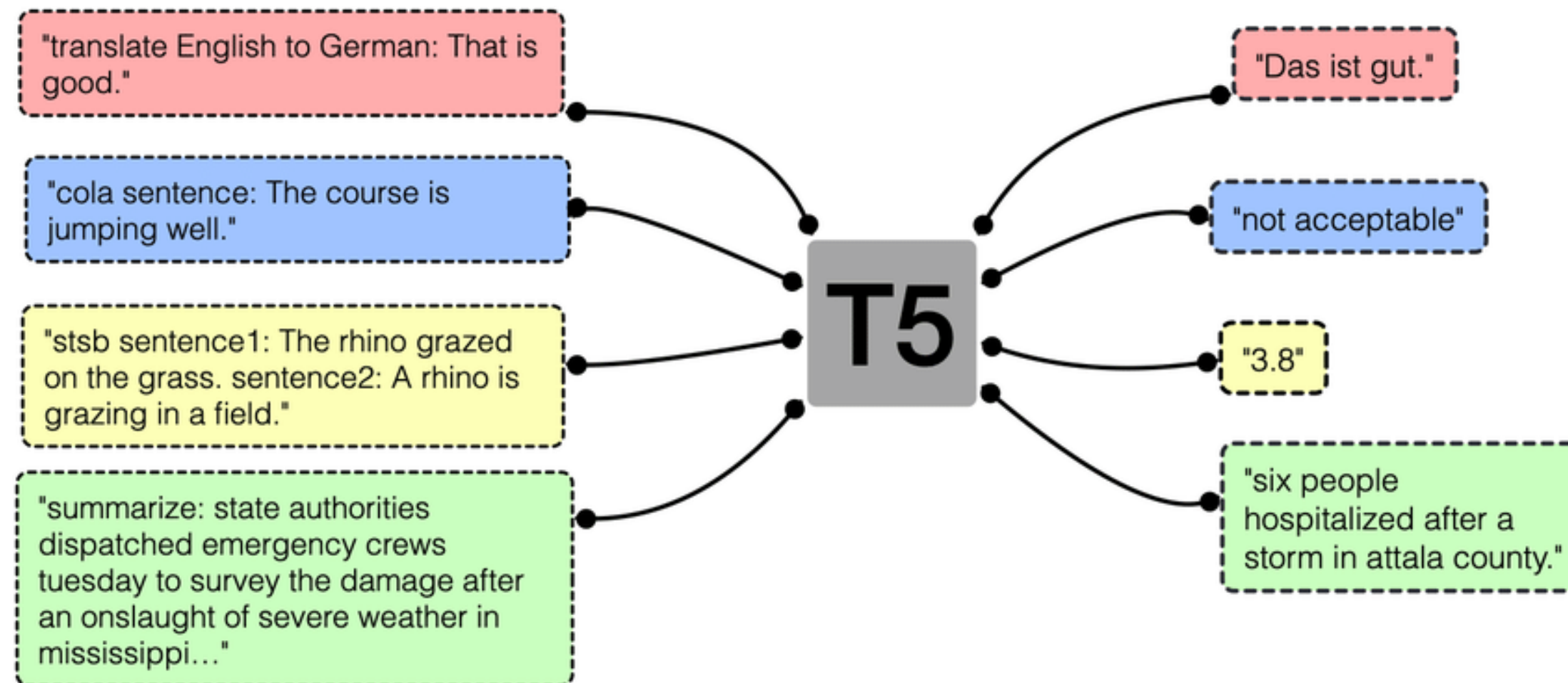
	KoBERT		KoGPT2
공개시기	2019년 10월		2020년 2월
한국어 학습 데이터	위키(500만문장 5400만단어), 뉴스(2000만문장 2억7000만단어)		위키(500만문장 5400만단어), 뉴스(1억2000만문장 16억단어), 기타(940만문장 8800만단어 · 1800만문장 8200만단어) 등 20GB 크기 원시문장
내부 활용처	챗봇(콜센터 상담 보조), AI검색(법무 · 특허등록 지원), 기계독해(내부 마케팅 자료 정보추출)		챗봇(대화형 인터페이스 자연어생성 최적화)
원형 기술	구글 BERT(2018년 10월 공개)		오픈AI GPT-2(2019년 2월 공개)
개발배경	BERT 한국어성능 한계 개선		GPT-2 한국어성능 한계 개선

[자료=깃허브 KoBERT, KoGPT2 프로젝트 소개]



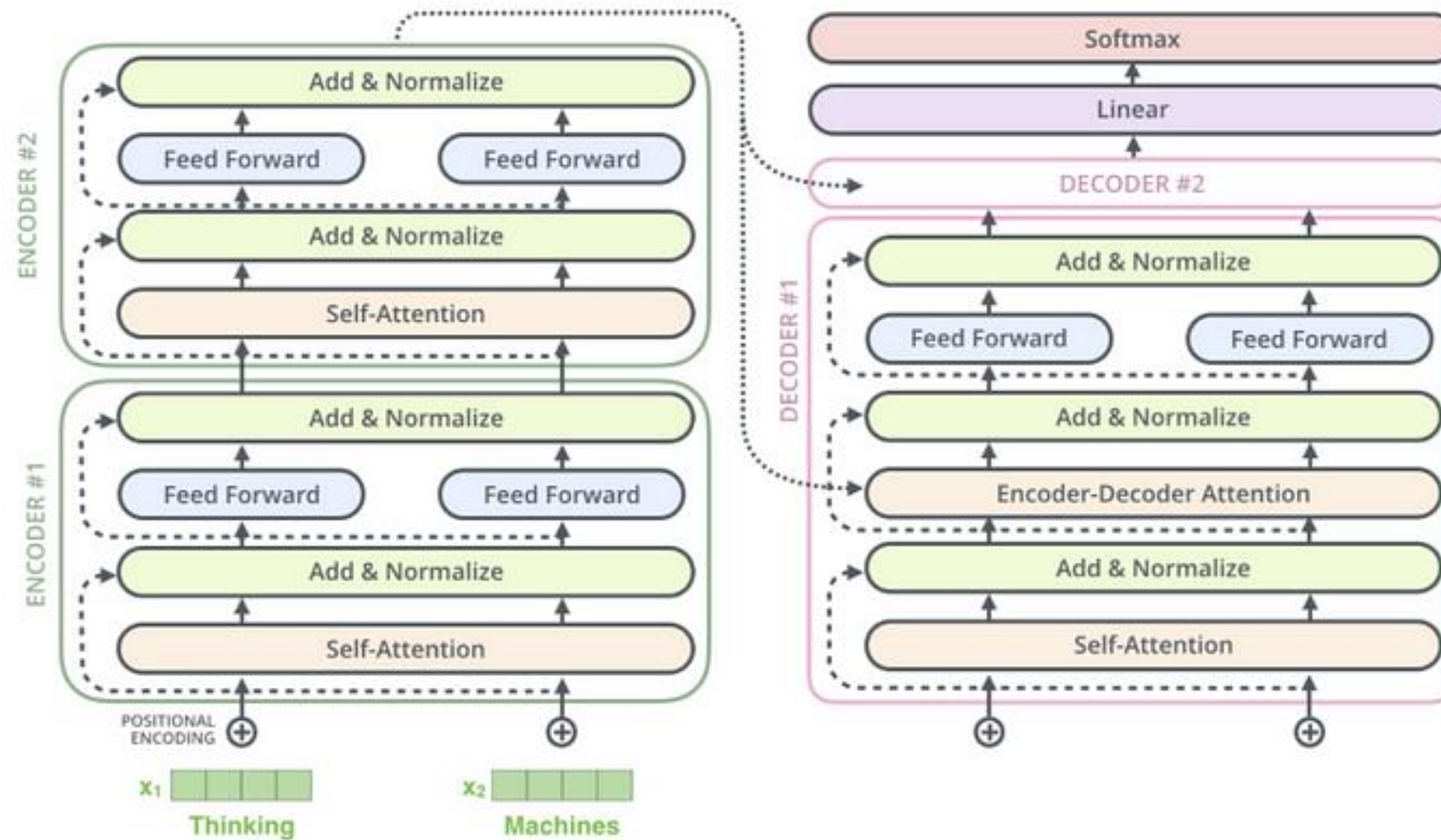
BERT 이후 주요 언어 모델 – T5

- T5 (Text-to-Text Transfer Transformer): 텍스트를 통한 모든 것
- 기본 아이디어
 - 모든 NLP 태스크를 "텍스트를 입력받아 텍스트를 출력하는" 문제로 변환
 - 예: "번역: Hello, World!" → "안녕, 세상!"



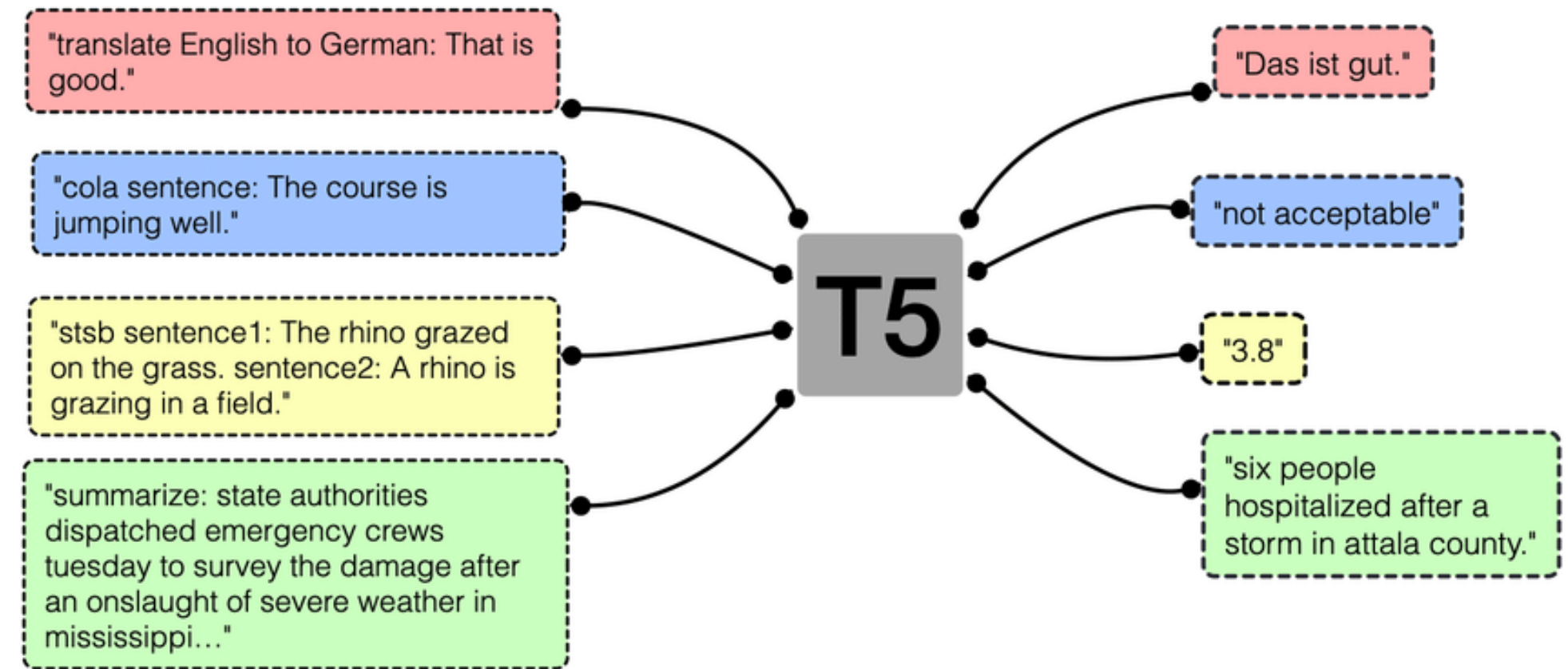
BERT 이후 주요 언어 모델 – T5

- T5 모델 구조
 - Transformer 기반의 인코더-디코더 구조
 - BERT나 GPT와는 달리, 인코더와 디코더 모두 사용
 - 사실상 마지막



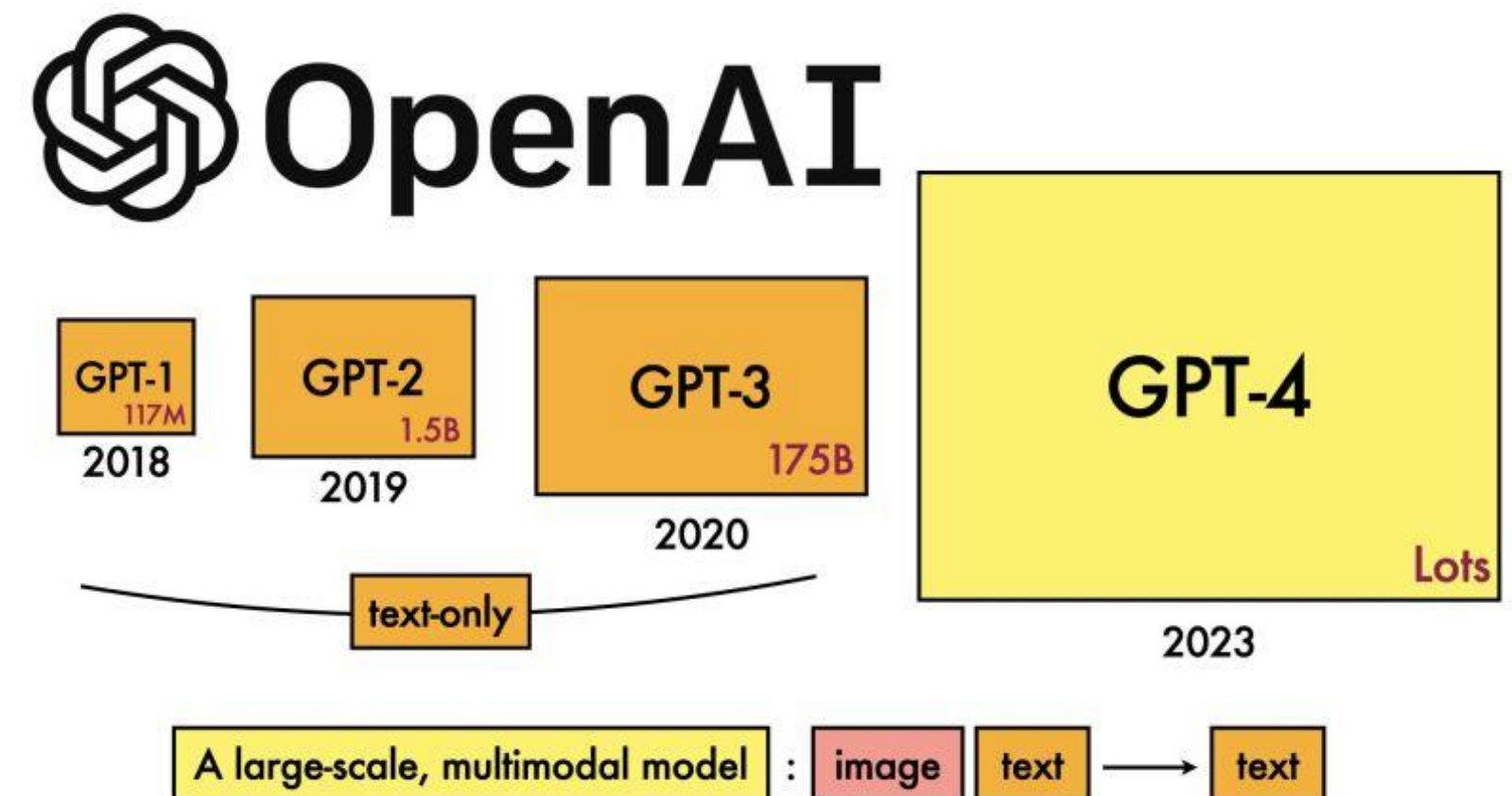
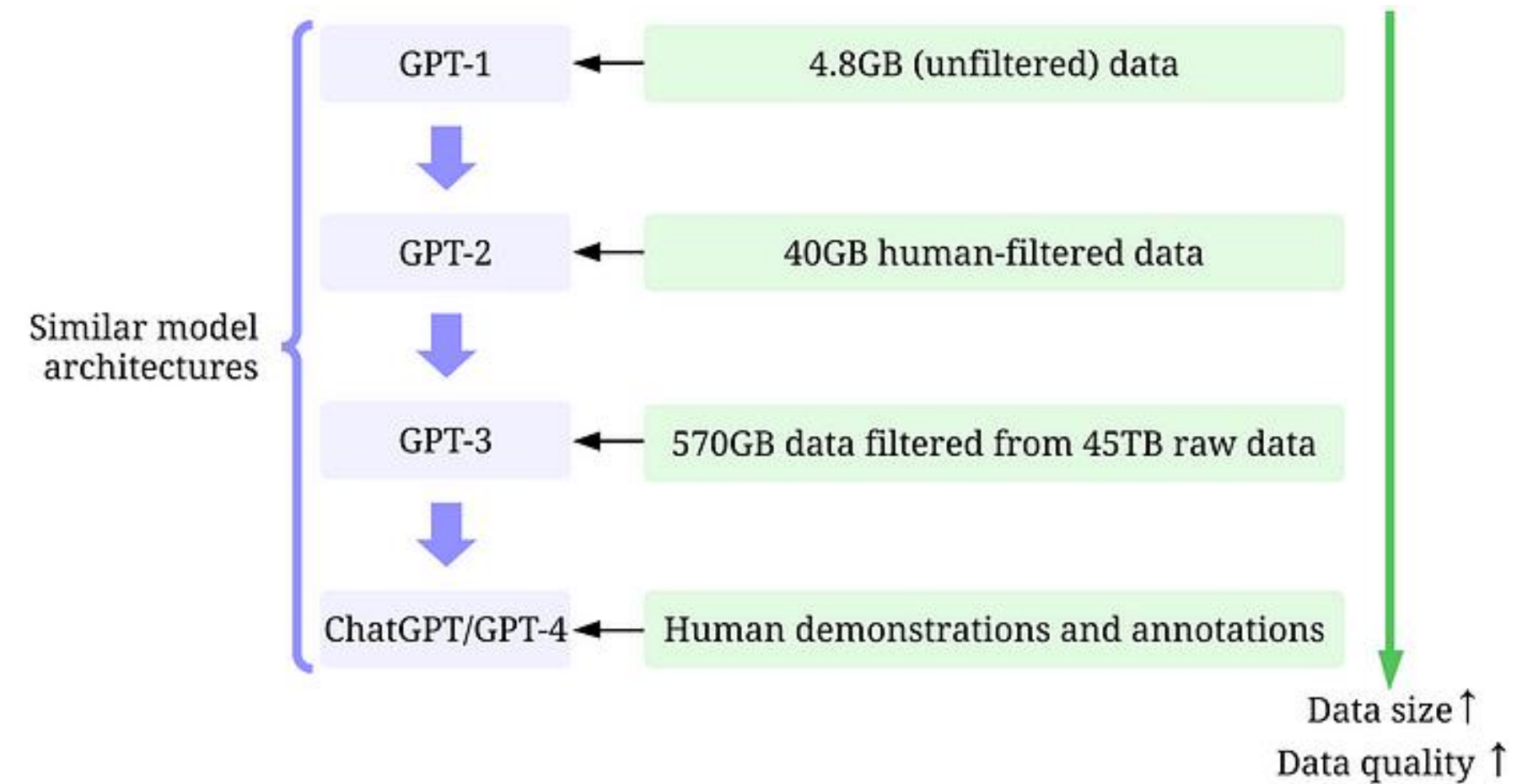
BERT 이후 주요 언어 모델 – T5

- T5 학습 방식
 - 사전 학습 (Pre-training)
 - 대규모 텍스트 데이터(C4)를 사용하여 언어 모델링
 - 마스킹된 텍스트 복원 등의 방법 활용
 - 미세 조정 (Fine-tuning)
 - 특정 태스크의 데이터를 사용하여 모델 미세 조정
 - 태스크 Prefix를 입력 문장에 포함하여 학습



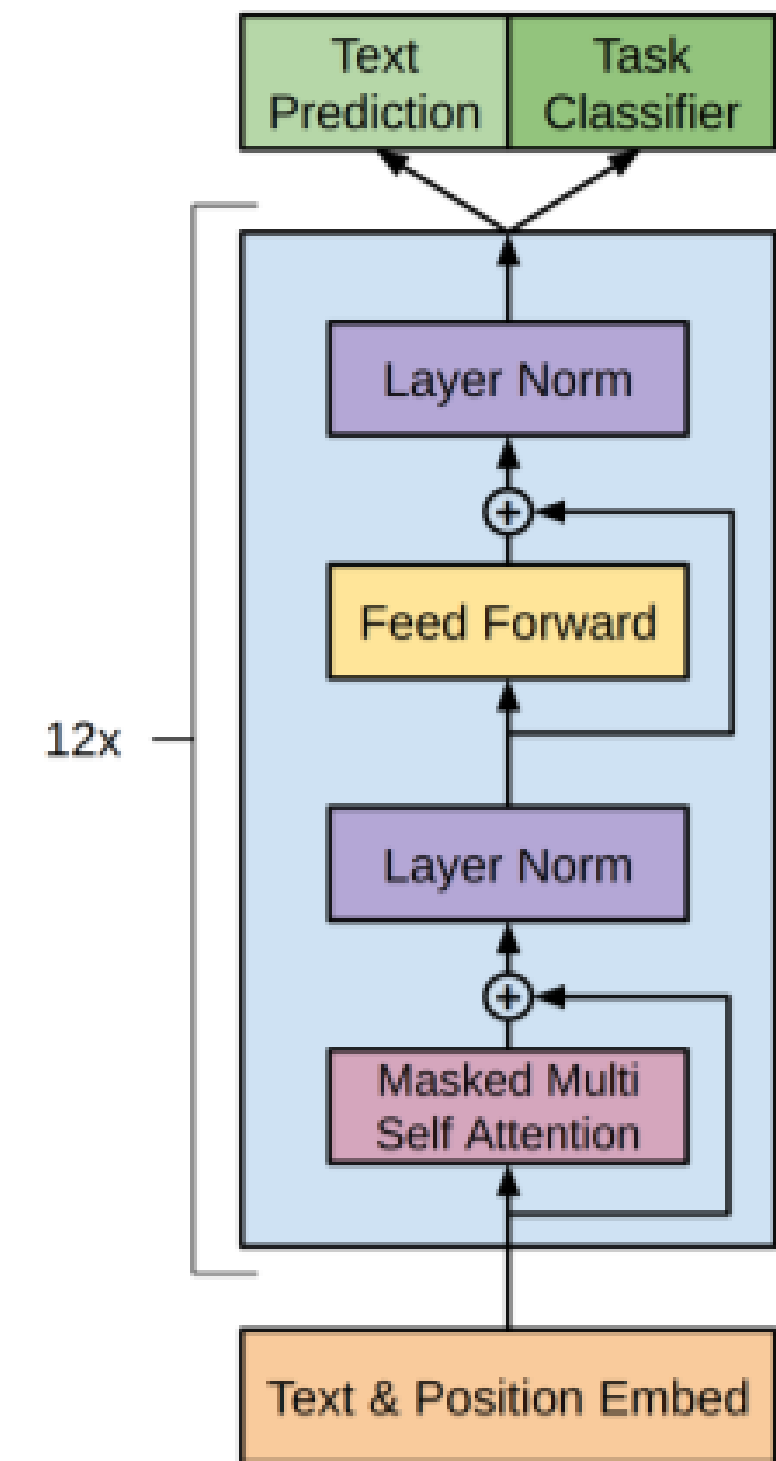
BERT 이후 주요 언어 모델 – GPT

- GPT: Generative Pre-trained Transformer
 - OpenAI에서 개발된 Text-generation 모델
 - Transformer 기반의 디코더만을 사용한 아키텍처
 - 대규모 텍스트 데이터를 바탕으로 훈련
 - 준지도학습(Semi-supervised learning) 사용
 - 모델의 버전이 올라갈 수록
 - 데이터의 수 증가
 - 파라미터의 수도 증가



BERT 이후 주요 언어 모델 – GPT

- GPT-1: Generative Pre-trained Transformer 1
 - 논문: [Improving Language Understanding by Generative Pre-Training]
 - 디코더만으로 구성된 모델
 - 문장 생성이 목표
 - 문장 생성을 통해 모든 문제 해결 가능
 - 지도학습을 통한 모델 훈련
 - 비지도학습을 통한 Pretraining
 - 지도학습을 통한 Fine-tuning
 - 비지도학습은 Language Modelling을 통해 진행
 - MLM, NSP 등과 비슷한 방식

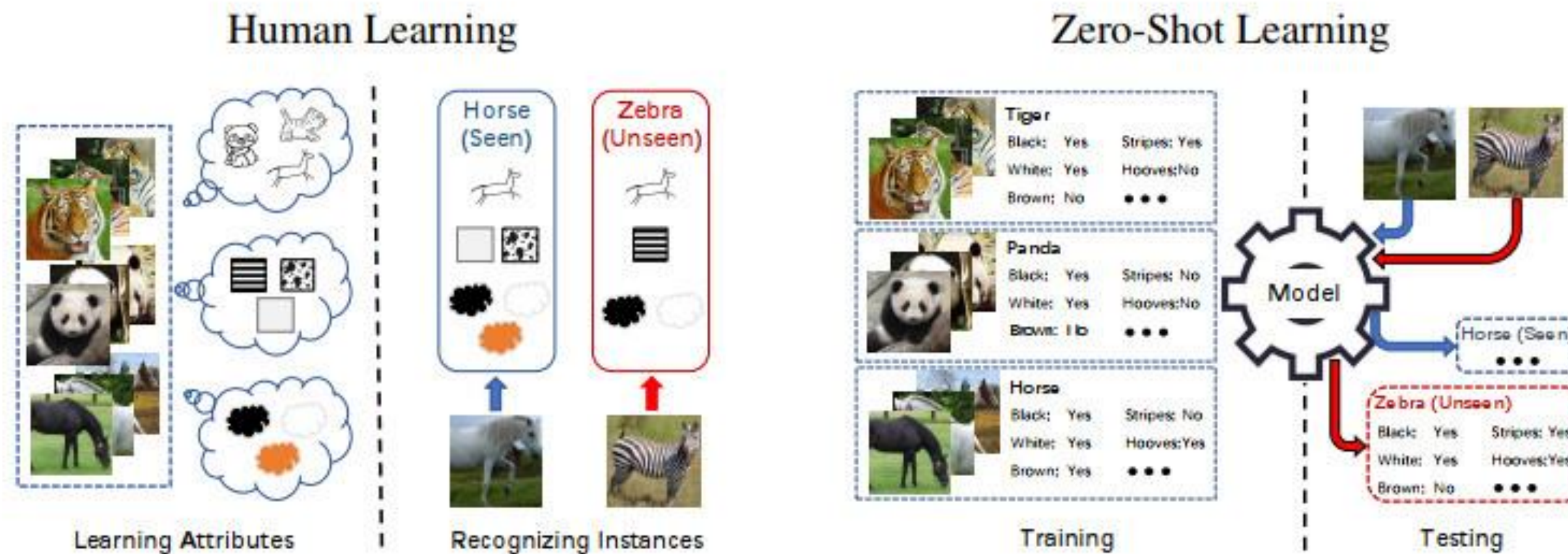


BERT 이후 주요 언어 모델 – GPT

- GPT-2: Generative Pre-trained Transformer 2
 - 논문: [Language Models are Unsupervised Multitask Learners]
 - 학습 방식
 - 지도학습: 대규모 텍스트 데이터를 사용하여 언어 모델링
 - 특징
 - 구조와 학습 방식이 GPT-1과 동일
 - 다양한 NLP 태스크에서 Few-shot 학습 가능
 - 고도의 문장 생성 능력
- 초기에는 모델의 크기와 생성 능력 때문에 공개를 주저함

N-shot learning

- N-Shot: 학습 과정 중 참조한 데이터의 수
 - Few-shot: 학습 과정 중 적은 수의 데이터를 참조
 - 1-shot: 학습 샘플 중 특정 데이터가 하나만 존재
 - 0-shot: 학습 샘플에 특정 데이터가 포함되지 않음
- 텍스트 생성 모델은 학습하지 않은 문장도 생성할 수 있음



BERT 이후 주요 언어 모델 – GPT

- GPT-3: Generative Pre-trained Transformer 3
 - 논문: [Language Models are Few-Shot Learners]
 - 모델 크기
 - 1750억 개의 파라미터를 가진 거대한 모델
 - 학습 방식
 - GPT-2와 유사한 준지도학습, 하지만 더 큰 데이터와 모델로 학습
 - 특징
 - 매우 다양한 태스크에서 Zero-shot, Few-shot 학습 능력
 - 자연어 질의응답, 문장 생성, 번역, 요약 등 다양한 작업 수행 가능
 - API 및 상용화
 - GPT-3 기반의 API가 제공되어 다양한 애플리케이션 개발에 활용

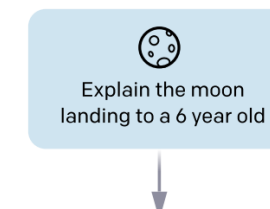
BERT 이후 주요 언어 모델 – GPT

- ChatGPT
 - OpenAI에서 만든 GPT를 활용한 **챗봇**
 - GPT-3.5, GPT-4를 활용하여 대답
 - GPT에게 대화하는 법을 지시
 - 완성된 문장으로 기술
 - 민감하거나 유해한 내용 배제

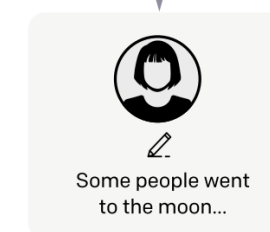
Step 1

Collect demonstration data, and train a supervised policy.

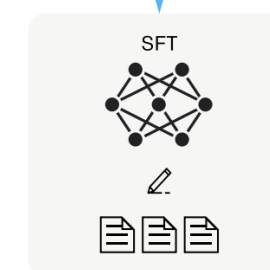
A prompt is sampled from our prompt dataset.



A labeler demonstrates the desired output behavior.



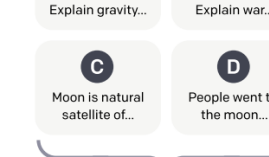
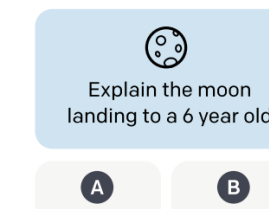
This data is used to fine-tune GPT-3 with supervised learning.



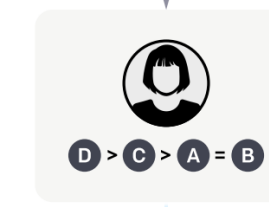
Step 2

Collect comparison data, and train a reward model.

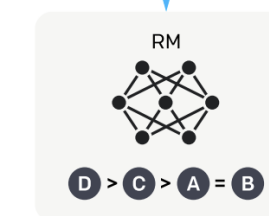
A prompt and several model outputs are sampled.



A labeler ranks the outputs from best to worst.



This data is used to train our reward model.



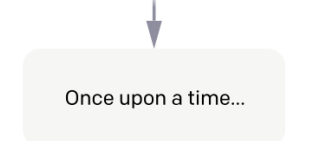
Step 3

Optimize a policy against the reward model using reinforcement learning.

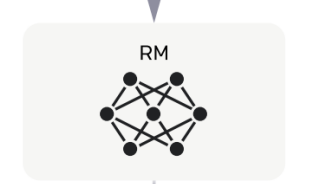
A new prompt is sampled from the dataset.



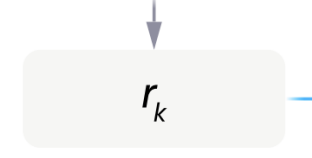
The policy generates an output.



The reward model calculates a reward for the output.



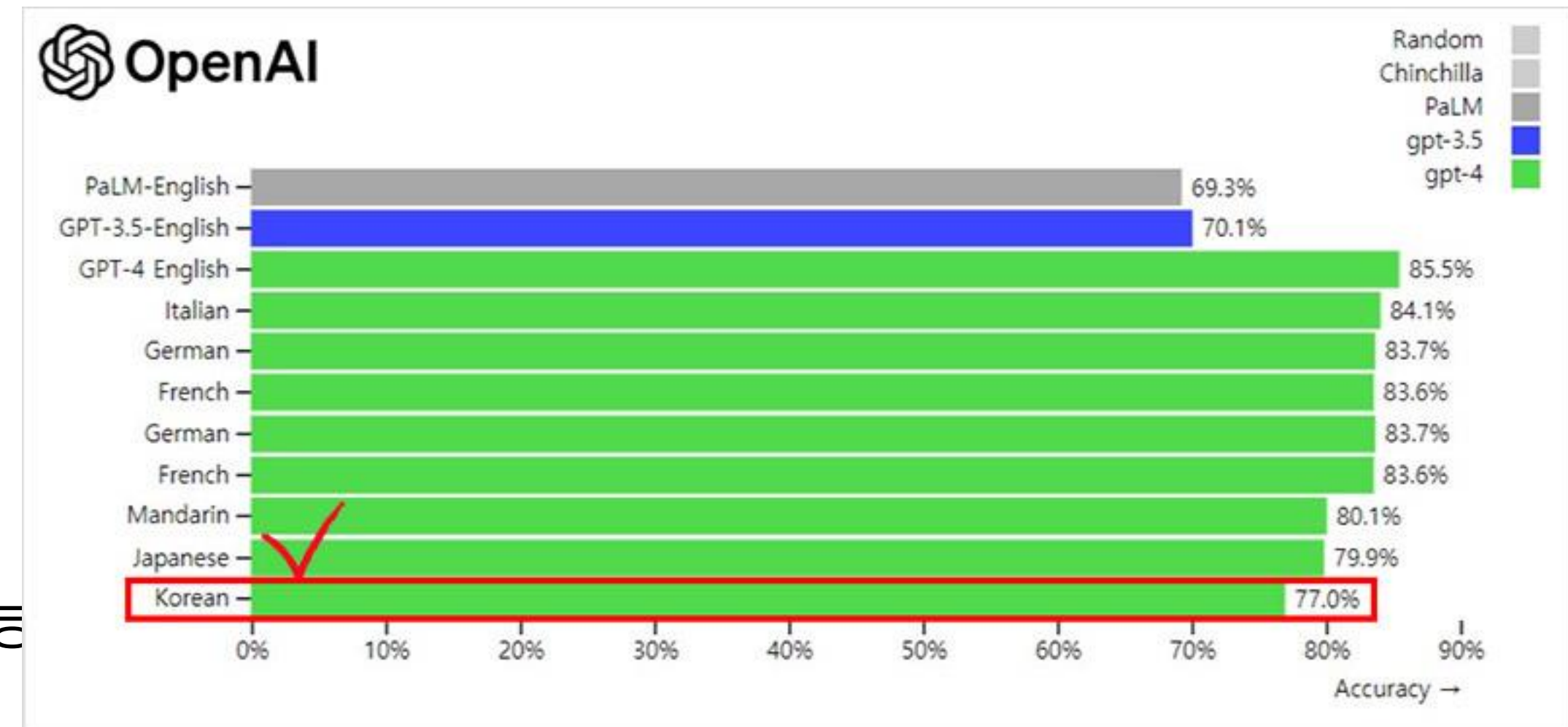
The reward is used to update the policy using PPO.



BERT 이후 주요 언어 모델 – GPT

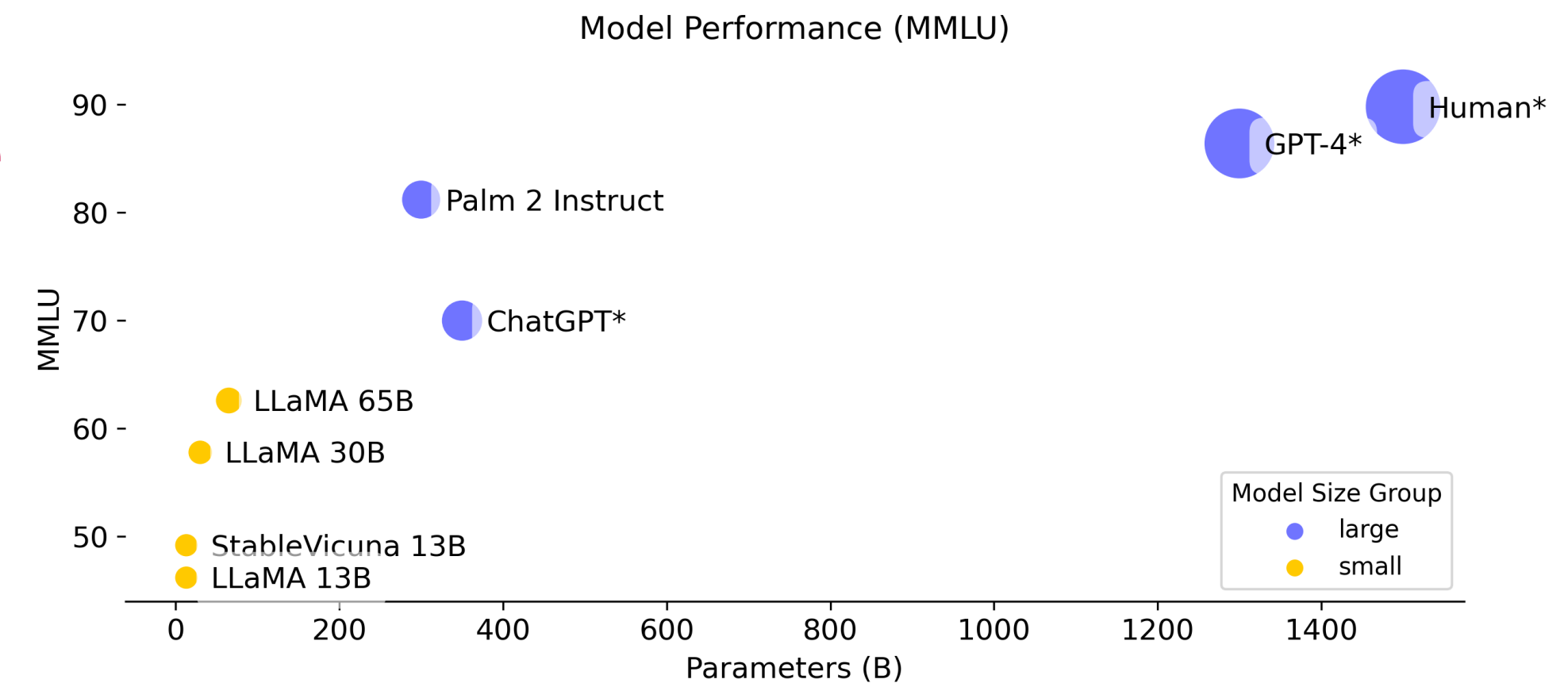
- GPT-4

- 엄청나게 큰 모델로 많은 양의 데이터를 오랜 기간 학습
- 추정치(풍문)
 - 모델 파라미터 수 1.8tr(1.8조)
 - Nvidia A100 * 25,000 사용
 - \$6300만(=850억 원)
 - 학습 기간 90 – 100일
- 이미지, 음성 등 다양한 데이터 동시 이해 가능



BERT 이후 주요 언어 모델 – LLaMA

- GPT-4
 - 엄청나게 큰 모델로 많은 양의 데이터를 오랜 기간 학습
 - 과연 모델의 파라미터 수는 데이터의 양에 적합할까?
 - 모델 파라미터 공간이 데이터에 비해 작다면 Underfitting
 - 모델 파라미터 공간이 데이터에 비해 크다면 Overfitting
- Meta에서 LLaMA를 공개
 - [Open and Efficient Foundation Language Model]
 - 데이터 크기를 고려하여, 이에 적합한 모델 구축



*Exact model size is unknown. | Data from InstructEval GitHub.

BERT 이후 주요 언어 모델 – LLaMA

- LLaMA
 - 파라미터가 데이터 학습 능력을 낭비하지 않게끔 계산하여 설계
 - 네 가지 크기의 모델을 공개
 - 7B, 13B, 33B, 65B
 - 13B 모델로 GPT-3(175B)과 유사한 성능을 보임
 - 다만 다국어 성능에서 살짝 아쉬운 점수를 보임

		BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA
GPT-3	175B	60.5	81.0	-	78.9	70.2	68.8	51.4	57.6
Gopher	280B	79.3	81.8	50.6	79.2	70.1	-	-	-
Chinchilla	70B	83.7	81.8	51.3	80.8	74.9	-	-	-
PaLM	62B	84.8	80.5	-	79.7	77.0	75.2	52.5	50.4
PaLM-cont	62B	83.9	81.4	-	80.6	77.0	-	-	-
PaLM	540B	88.0	82.3	-	83.4	81.1	76.6	53.0	53.4
LLaMA	7B	76.5	79.8	48.9	76.1	70.1	72.8	47.6	57.2
	13B	78.1	80.1	50.4	79.2	73.0	74.8	52.7	56.4
	33B	83.1	82.3	50.4	82.8	76.0	80.0	57.8	58.6
	65B	85.3	82.8	52.3	84.2	77.0	78.9	56.0	60.2

BERT 이후 주요 언어 모델 – LLaMA

- LLaMA
 - 오픈 소스로 모델 공개
 - 모델 & 사전학습 가중치 사용 가능
 - 모델의 경우 윤리&보안 서약 동의 후 사용 가능
 - 이용자가 모델을 바탕으로 Fine-tuning 가능
 - 상업적 사용 전면 허용
 - LLaMA2, LLaMA3 도 동일
 - 지속적인 성능 업데이트
 - LLaMA2는 2배 많은 토큰을 입력 가능(2k, A4 기준 6페이지)
 - 코딩 등 다양한 측면에서 발표 당시 최고 성능을 기록

BERT 이후 주요 언어 모델 – LLaMA

- LLaMA 3.1

- 입력 컨텍스트 128k로 확장(~A4 200p)
- 8개국어 지원(한국어 미포함)
- 세 가지 라인업 발표
 - 8B, 70B, 405B
 - 8B > Mistral7B, Gemma2 9B
 - 70B > GPT3.5, 4o와 비슷
 - 405B는 GPT4와 유사한 성능

Category Benchmark	Llama 3.1 405B	Nemotron 4 340B Instruct	GPT-4 (0125)	GPT-4 Omni	Claude 3.5 Sonnet
General					
MMLU (0-shot, CoT)	88.6	78.7 (non-CoT)	85.4	88.7	88.3
MMLU PRO (5-shot, CoT)	73.3	62.7	64.8	74.0	77.0
IFEval	88.6	85.1	84.3	85.6	88.0
Code					
HumanEval (0-shot)	89.0	73.2	86.6	90.2	92.0
MBPP EvalPlus (base) (0-shot)	88.6	72.8	83.6	87.8	90.5
Math					
GSM8K (8-shot, CoT)	96.8	92.3 (0-shot)	94.2	96.1	96.4 (0-shot)
MATH (0-shot, CoT)	73.8	41.1	64.5	76.6	71.1
Reasoning					
ARC Challenge (0-shot)	96.9	94.6	96.4	96.7	96.7
GPQA (0-shot, CoT)	51.1	-	41.4	53.6	59.4
Tool use					
BFCL	88.5	86.5	88.3	80.5	90.2
Nexus	58.7	-	50.3	56.1	45.7
Long context					
ZeroSCROLLS/QuALITY	95.2	-	95.2	90.5	90.5
InfiniteBench/En.MC	83.4	-	72.1	82.5	-
NIH/Multi-needle	98.1	-	100.0	100.0	90.8
Multilingual					
Multilingual MGSM (0-shot)	91.6	-	85.9	90.5	91.6