Transformer Variants

노기섭 교수

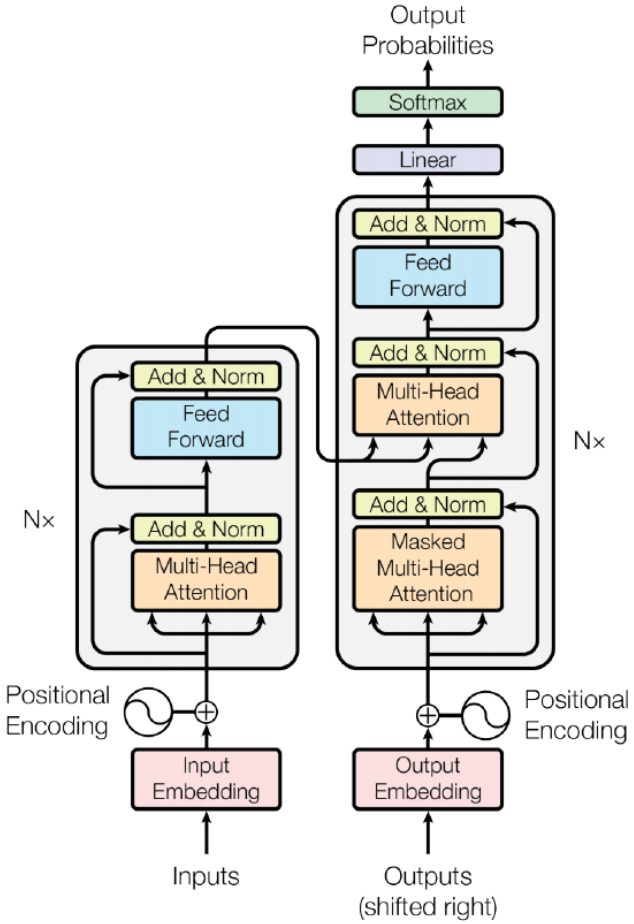
(kafa46@hongik.ac.kr)

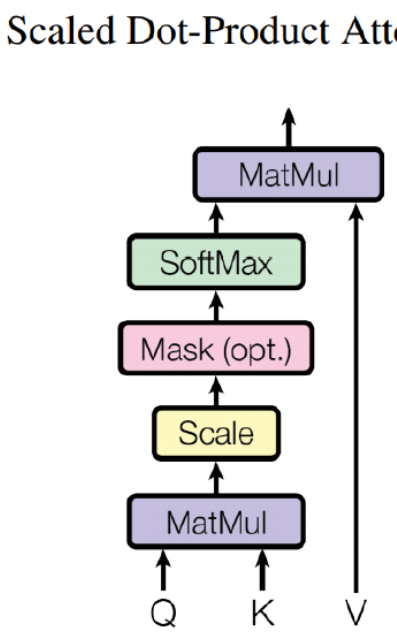
Contents

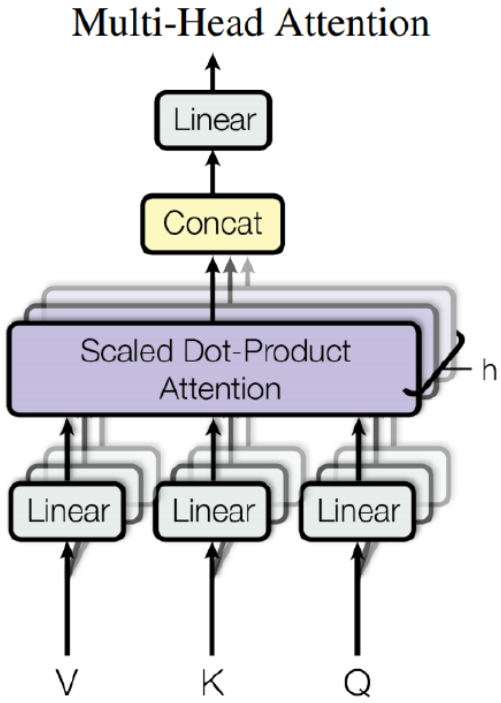
- Pre-trained Language Model
- Self-supervised Learning
- Auto-encoder Models
- Auto-regressive Models
- Encoder-Decoder Models

Pre-trained Language Model (PLM)

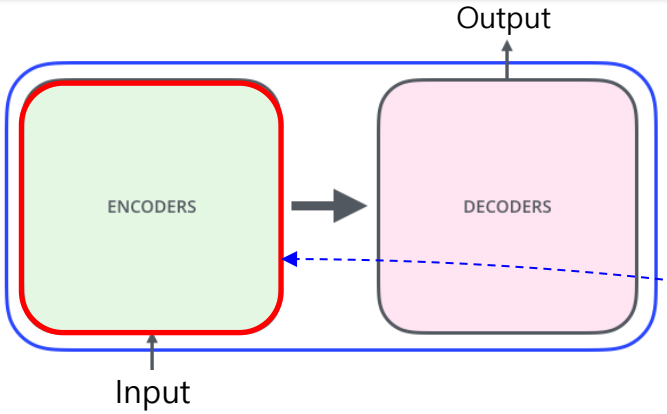
Recap Transformer



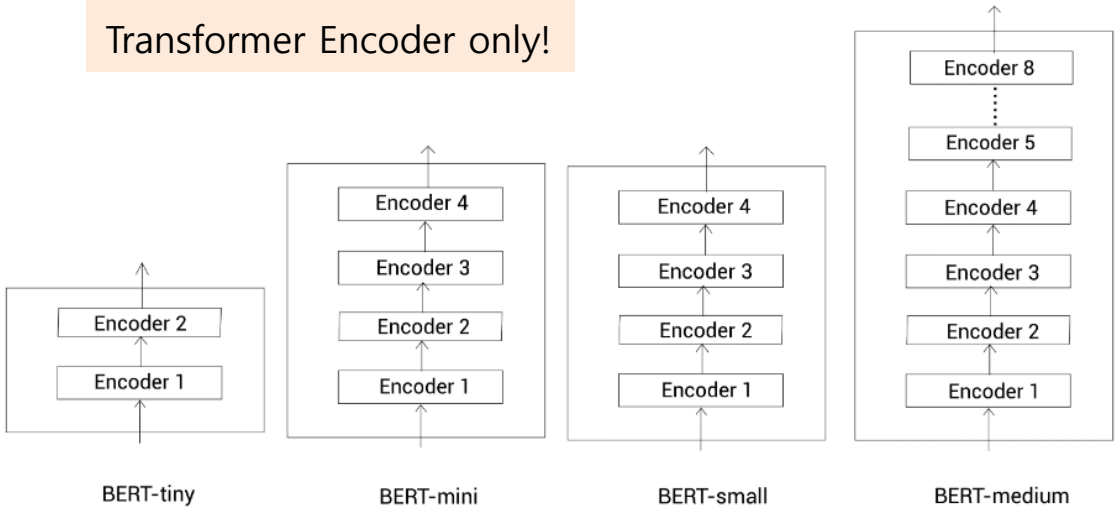


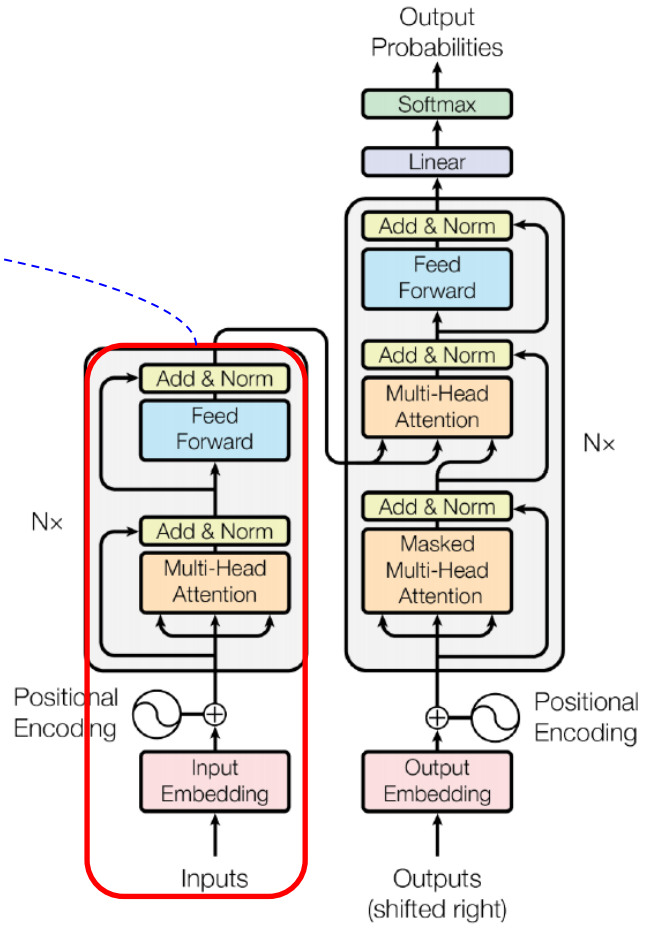


Bi-directional Encoder from Transformer

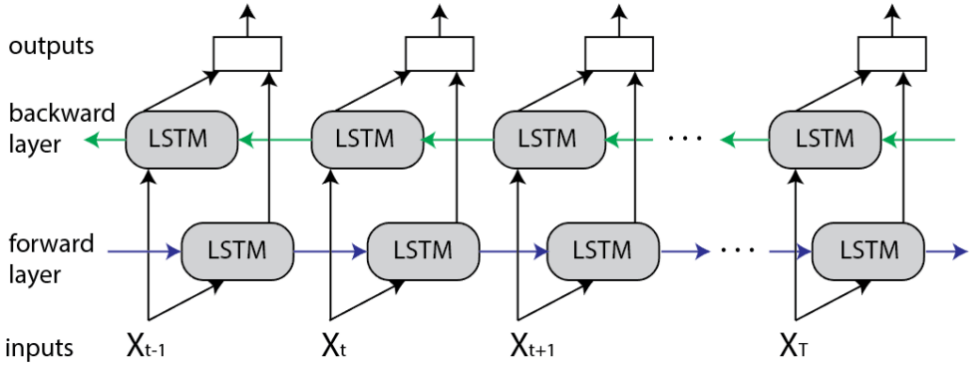


Transformer Encoder only!





Meaning of "Bi-directional"

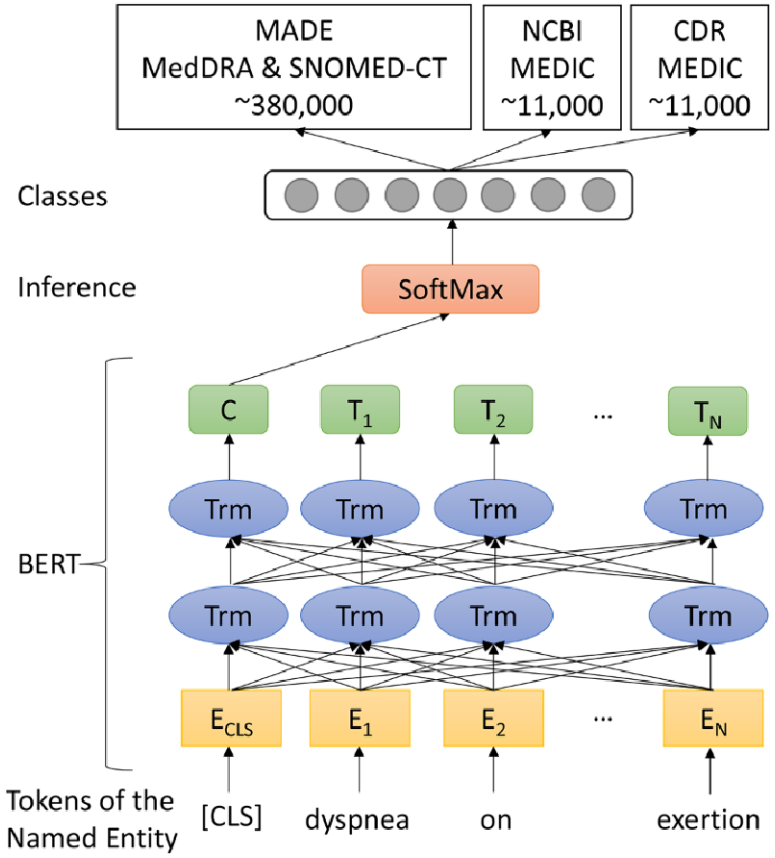


Apple is something that competitors simply cannot reproduce.

vs.

Apple is something that I like to eat.

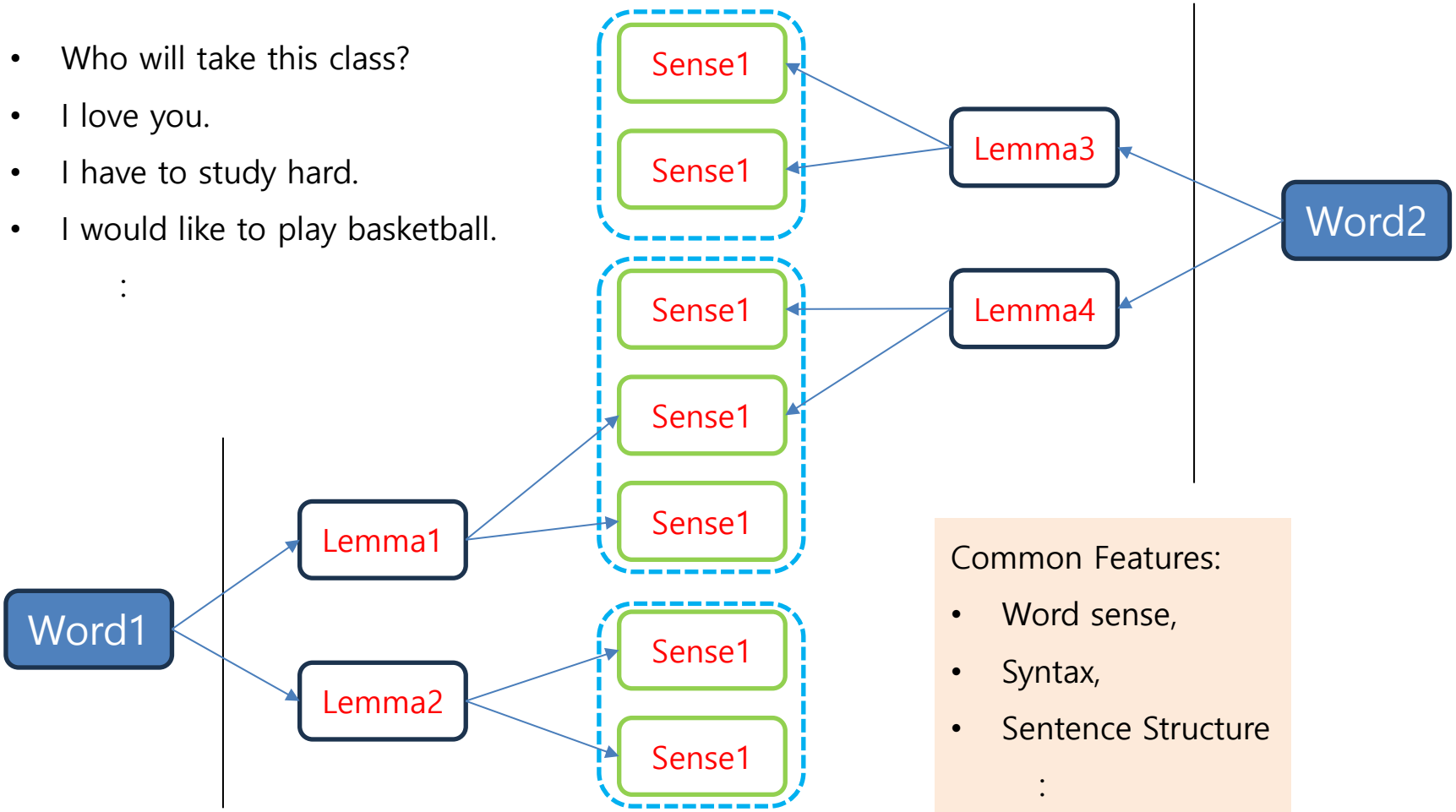
Bi-directional ➔ More meaningful representation!



Common Features in NLP

- Who will take this class?
- I love you.
- I have to study hard.
- I would like to play basketball.

:



Common Features:

- Word sense,
- Syntax,
- Sentence Structure

:

Common Features in Images

Intuition

- There exist common features even in different datasets
- Example

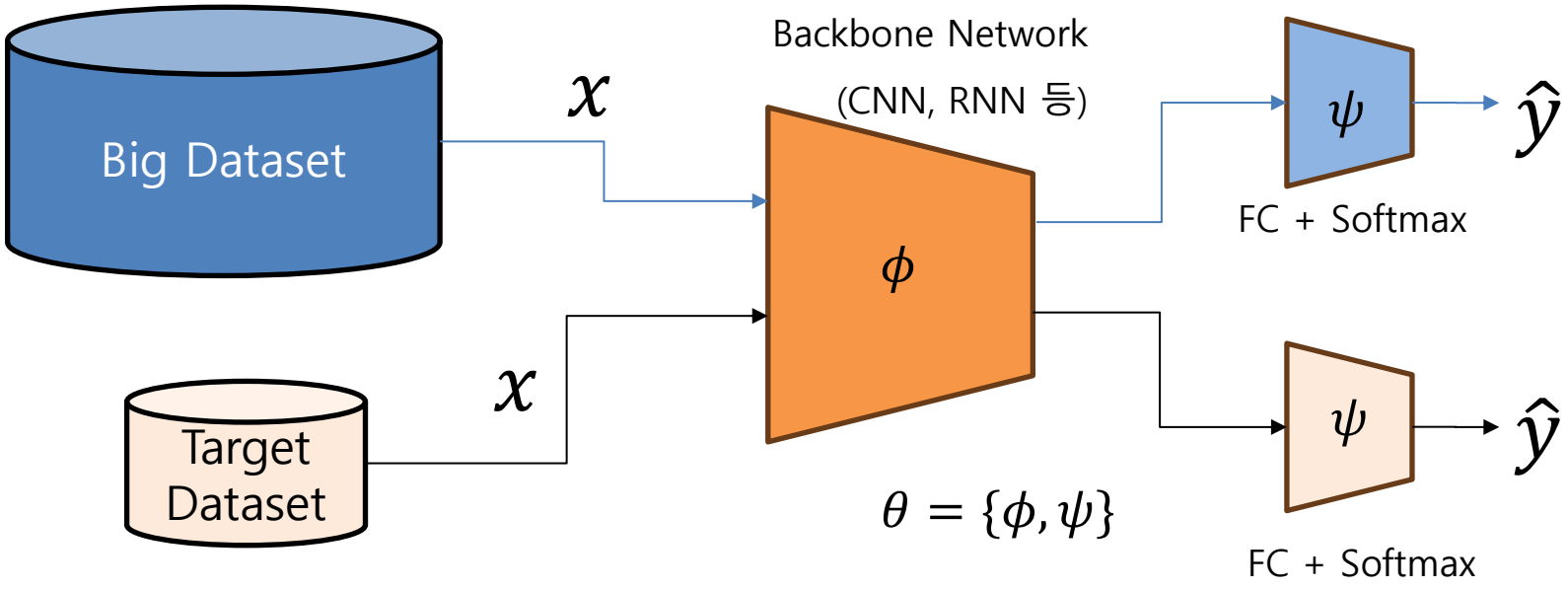
Goal

- Reuse pre-trained common features
- Improve learning efficiency

Concept of Transfer Learning

Transfer Learning

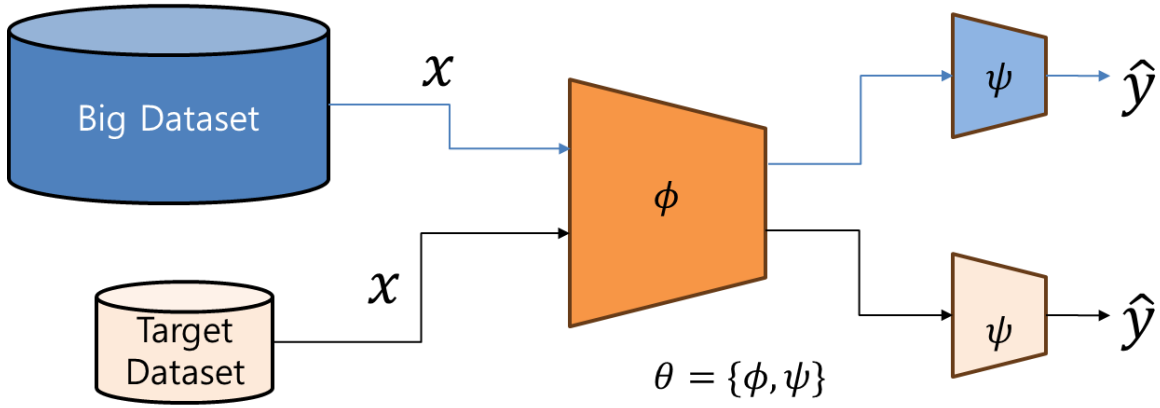
Transfer learning (TL) is a technique in machine learning (ML) in which knowledge learned from a task is re-used to boost performance on a related task.



Transfer Learning Approaches

Transfer Learning Approaches

- Load seed weight → Performing general optimization (learning)
- Freeze weights → Learn unloaded parameters
- Apply different learning rates

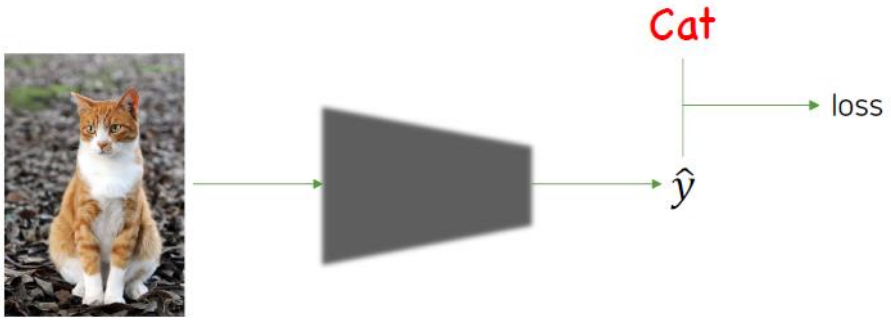


Self-supervised Leaning

Learning Types

Supervised Learning

- With labels
  - $D = \{(x_i, y_i)\}_{i=1}^{n}$
- Learn relation between x and y
- Almost commercial applications



Unsupervised Learning

- No labels
  - $D = \{(x_i)\}_{i=1}^{n}$
- Learn distribution in dataset, mainly used in generative models
- Autoencoders, GAN (Generative Adversarial Networks), Clustering etc.

Self-supervised Learning

- Supervised: need big labeled dataset
- Unsupervised: difficulty of achieving commercial level

Exploit inner(structure) of dataset → Train like labels exist

- Using partial information (x) of dataset, train to predict remain information (y) as label
- Once generate models → Apply different domain (Transfer Learning) → Maximize the performance of supervised learning.

In NLP, how to apply?

- NL data is everywhere in Internet.
- But no labels.
- Strategy: generate labels ➔ Using partial tokens as labels
  - 파이토치(PYTORCH) 한국어 _____에 오신 것을 환영합니다!
  - 이를 통해 얻은 모델을 다른 _____에 적용(전이 _____)

Self-supervised Learning + PLM

- **Supervised learning with unlabeled dataset**
- **Huge-scale learning is possible!**
- **Generate PLM**
  - seed of good weight parameter
  - large-scale training!
- **By applying transfer learning, the better performance is possible!**
- **Shadow of PLM**
  - Not a new model or algorithm → just scale-up
  - Environmental problems
  - Richer get richer
  - Not learn real-world knowledge, just mimic human behavior

Contrastive Learning

Contrastive loss in self-supervised learning

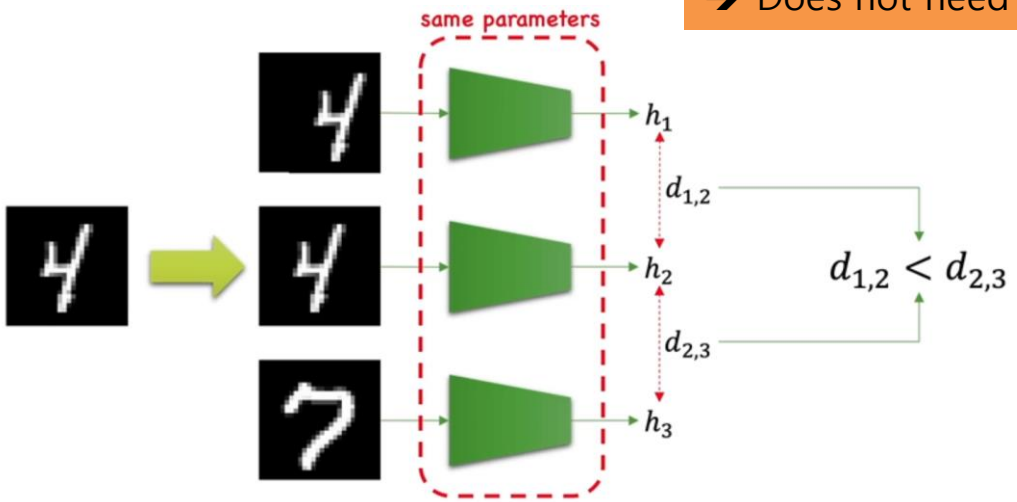
- Measure distance similar (positive) pair of hidden representation
- Measure distance different (negative) pair of hidden representation

Example

- Learning cross entropy in softmax

Learn parameters to be maximize distance between positive pair ($d_{1,2}$) and negative pair ($d_{2,3}$)

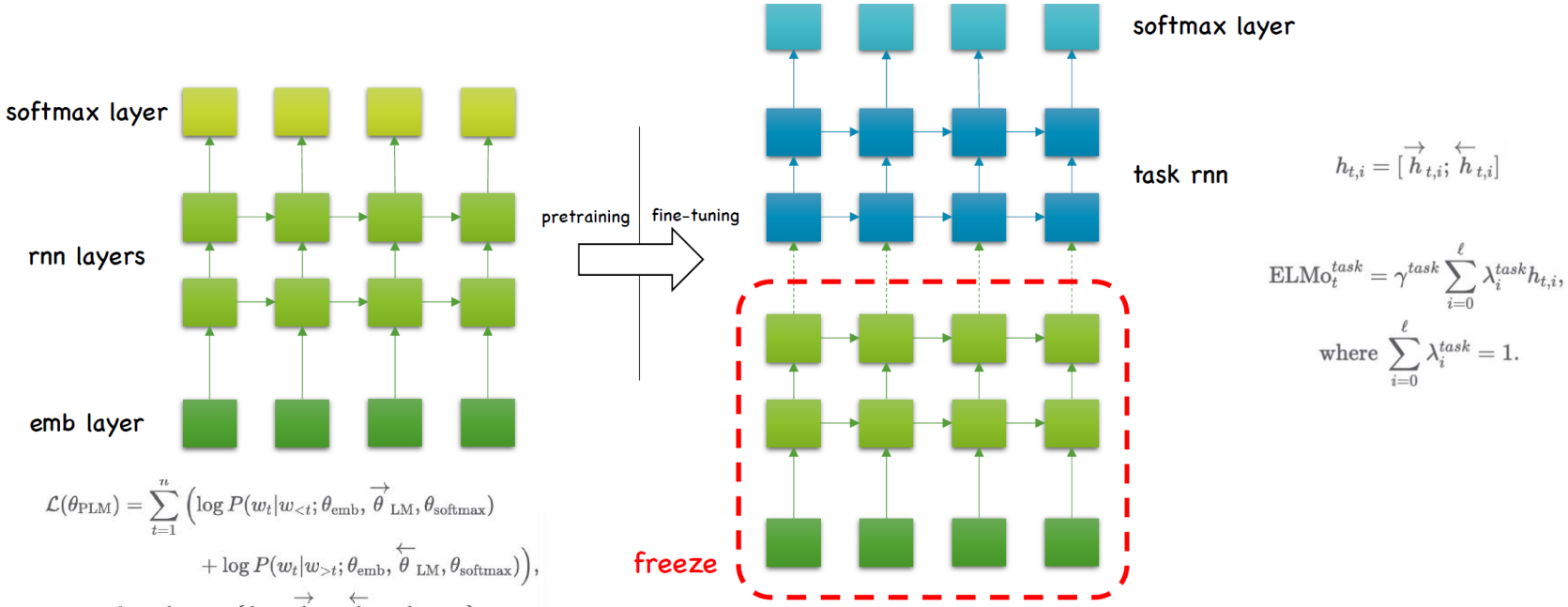
➔ Does not need labeled data



ELMo, PLM 시대의 서막을 올린 기술!

ELMo (Embedding from Language Models)

- Paper: https://arxiv.org/abs/1802.05365 [Matthew E. Peters et al., 2018]
- Design philosophy: Beyond Fixed Word Embedding (Word2Vec, Skip-gram 등), Context-based Embedding
- Existing embedding technologies: Cannot train under training objective (always same embedding)



$$\mathcal{L}(\theta_{\text{PLM}}) = \sum_{t=1}^{n} \Big(\log P(w_t | w_{<t}; \theta_{\text{emb}}, \overrightarrow{\theta}_{\text{LM}}, \theta_{\text{softmax}}) + \log P(w_t | w_{>t}; \theta_{\text{emb}}, \overleftarrow{\theta}_{\text{LM}}, \theta_{\text{softmax}}) \Big),$$

where $\theta_{\text{PLM}} = \{\theta_{\text{emb}}, \overrightarrow{\theta}_{\text{LM}}, \overleftarrow{\theta}_{\text{LM}}, \theta_{\text{softmax}}\}.$

$$\hat{\theta}_{\text{PLM}} = \underset{\theta_{\text{PLM}} \in \Theta}{\text{argmin}}\, \mathcal{L}(\theta_{\text{PLM}})$$

$$h_{t,i} = [\overrightarrow{h}_{t,i}; \overleftarrow{h}_{t,i}]$$

$$\text{ELMo}_t^{task} = \gamma^{task} \sum_{i=0}^{\ell} \lambda_i^{task} h_{t,i},$$

where $\sum_{i=0}^{\ell} \lambda_i^{task} = 1.$

자료출처: 패스트캠퍼스 김기현의 BERT, GPT-3를 활용한 자연어 처리

ELMo 성능과 한계

Task	Previous SOTA		Our baseline	ELMo + baseline	Increase (absolute/ relative)
SQuAD	Liu et al. (2017)	84.4	81.1	85.8	4.7 / 24.9%
SNLI	Chen et al. (2017)	88.6	88.0	88.7 ± 0.17	0.7 / 5.8%
SRL	He et al. (2017)	81.7	81.4	84.6	3.2 / 17.2%
Coref	Lee et al. (2017)	67.2	67.2	70.4	3.2 / 9.8%
NER	Peters et al. (2017)	91.93 ± 0.19	90.15	92.22 ± 0.10	2.06 / 21%
SST-5	McCann et al. (2017)	53.7	51.4	54.7 ± 0.5	3.3 / 6.8%

Dataset Goals

- SQuAD: Question & Answering
- SNLI: Textual Entailment
- SRL: Semantic Role Labeling
- Coref: Coreference Resolution
- NER: Named Entity Recognition
- SST-5: Semantic Analysis

Achievement

- Performance enhancement using context word embedding

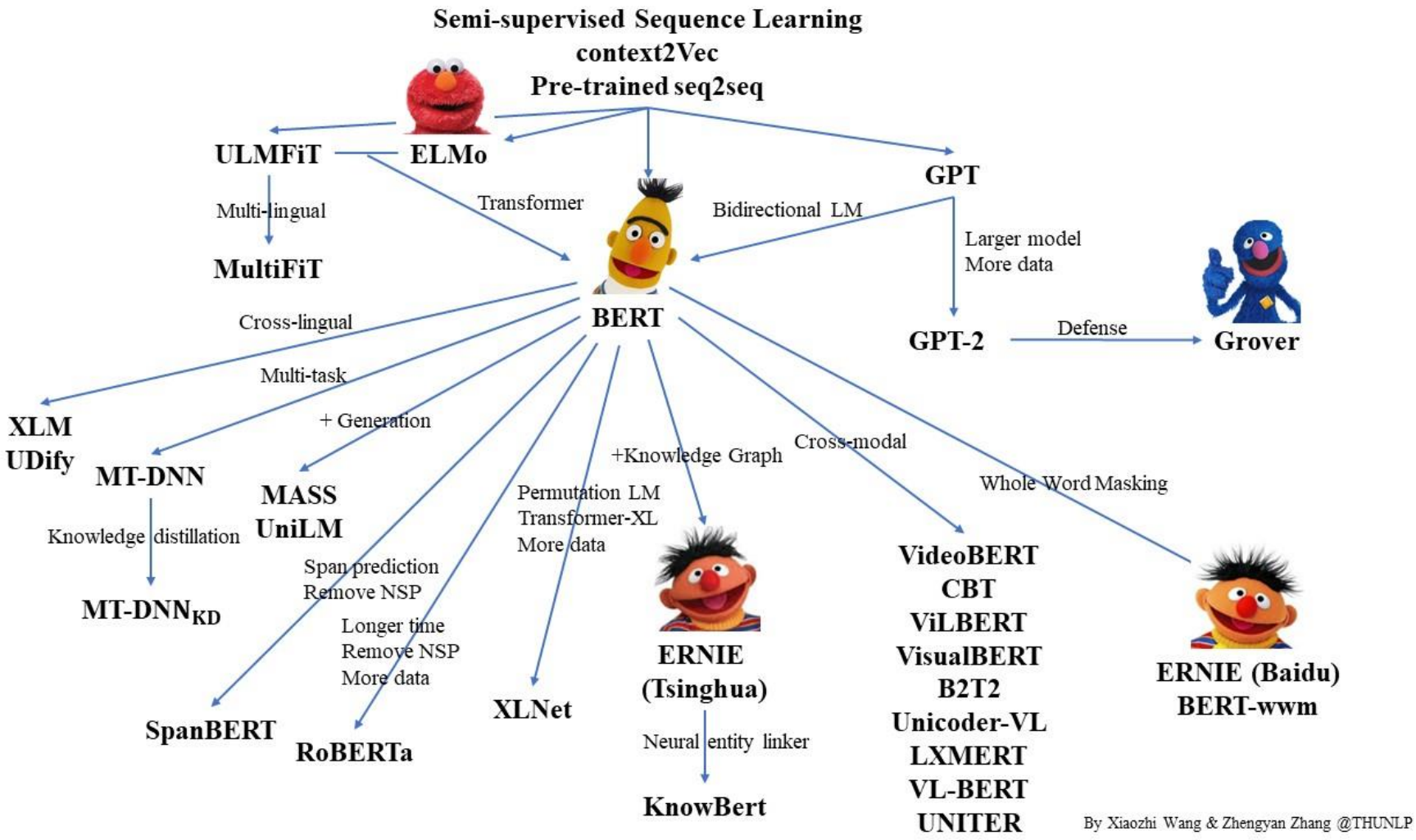
Limitation

- Additional model is need for each task
- Bi-directional LM, but only contain one-directional information

Implication

- Additional enhancement using Transformer's encoder, no longer use.
- However, this paper is monumental paper that opened the future door of PLM approach

PLM Descendants from ELMo



이미지 출처: https://github.com/thunlp/PLMpapers

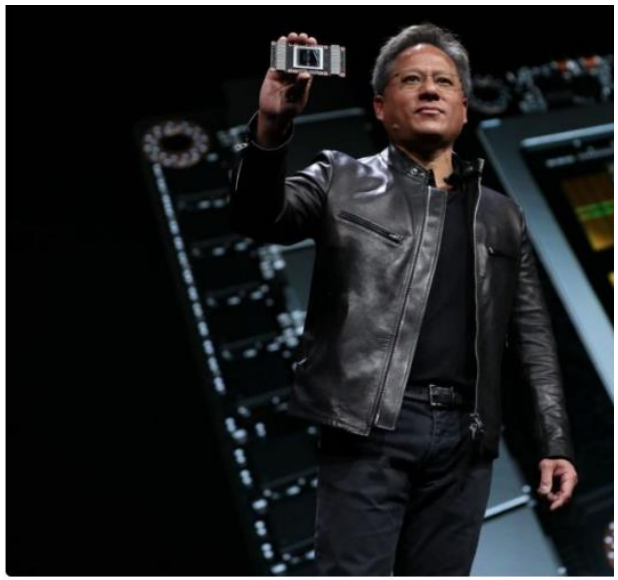
Bigger, Bigger, and Bigger!!!

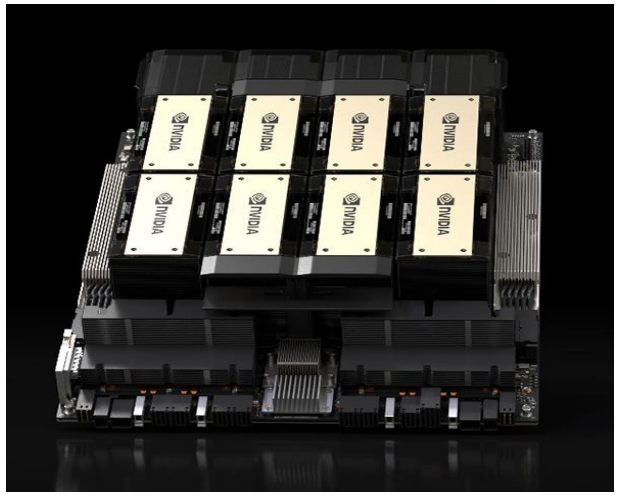
Bn: billion 파라미터 수
십억(10^9) 개의 파라미터

Tbps
= 테라비트/초
= 1조 비트/초

이미지 출처: https://endplan.ai/%EC%96%91%EC%9E%90%ED%99%94-quantization-1-58bit/

빈익빈 부익부 ㅠㅠ





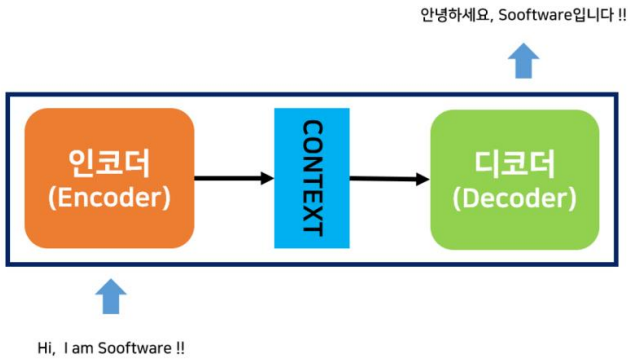
NVIDIA H200 GPU

Graphics Card Comparison for ML/AI

Feature	NVIDIA H200 (SXM)	NVIDIA H100 (SXM)	AMD Instinct MI300X	NVIDIA A100 (PCIe/SXM)	NVIDIA RTX 4090 (Consumer)
Architecture	Hopper (Enhanced)	Hopper	CDNA 3	Ampere	Ada Lovelace
VRAM	141 GB HBM3e	80 GB HBM3	192 GB HBM3	40 GB or 80 GB HBM2e	24 GB GDDR6X
Memory Bandwidth	4.8 TB/s	3.35 TB/s	5.3 TB/s	~2.0 TB/s (80GB)	~1.0 TB/s
FP8 Support	Yes, Native	Yes, Native	Yes, Native	No	Yes (via Tensor Cores)
Tensor Cores	4th Gen	4th Gen	N/A (AMD equivalent)	3rd Gen	4th Gen
Interconnect	NVLink, PCIe Gen 5	NVLink, PCIe Gen 5	AMD Infinity Fabric	NVLink, PCIe Gen 4	PCIe Gen 4
Typical Price (Approx.)	$40,000+	$25,000 - $35,000	N/A (Enterprise)	$10,000 - $15,000	$1,600 - $2,000
Best Use Case	Large-scale LLM training/inference, maximum bandwidth	Large-scale AI training & HPC, top performance	LLM inference, large batch sizes, massive VRAM capacity	Cost-efficient enterprise AI training/inference, virtualization	Individual research, fine-tuning, excellent cost/performance for smaller models

Auto-encoder Models

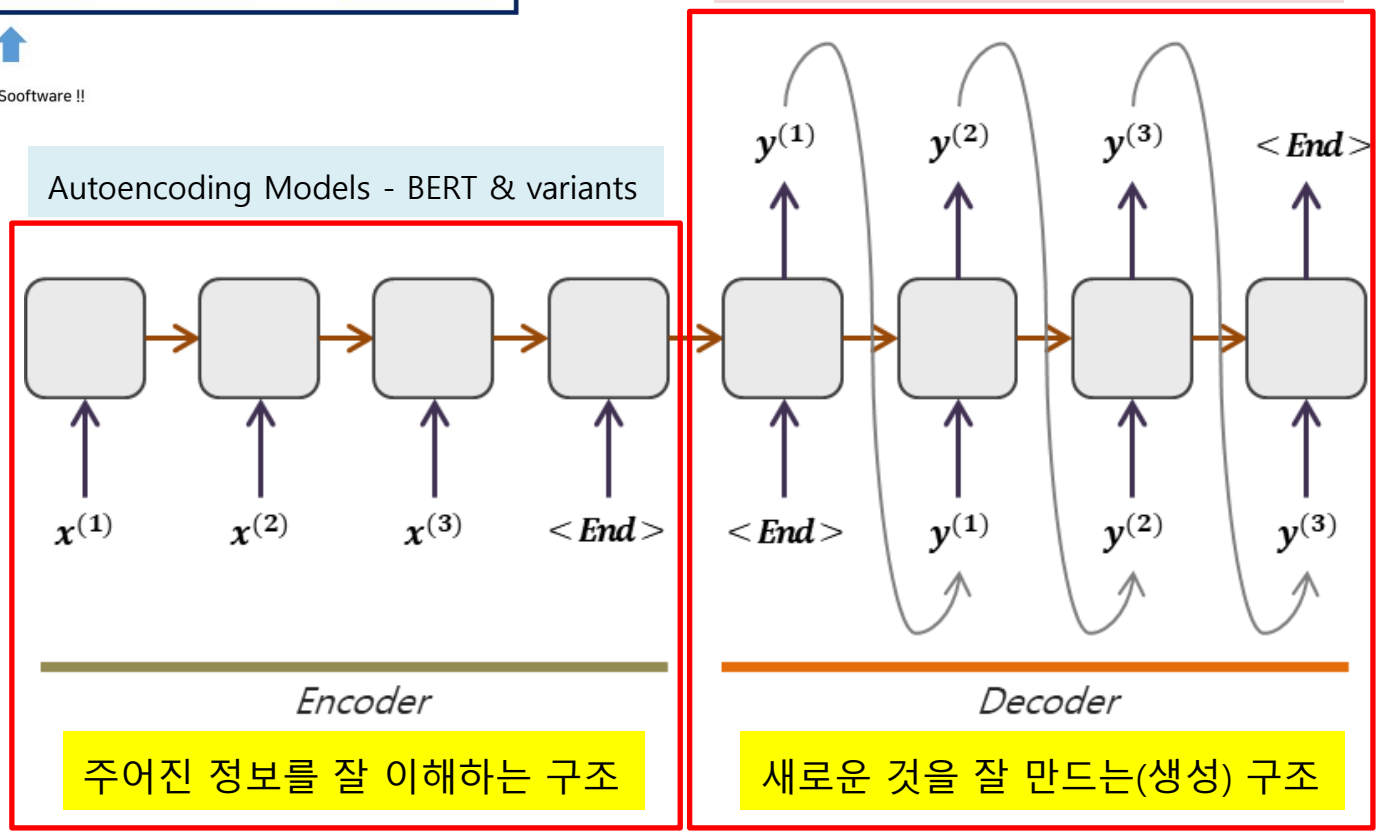
Autoencoding vs. Autoregressive



Transformer's encoder & decoder simultaneously

Autoregressive Models - GPT & variants

Autoencoding Models - BERT & variants



주어진 정보를 잘 이해하는 구조

새로운 것을 잘 만드는(생성) 구조

Transformer Encoder Models

Transformer의 진화

- Autoencoding Models ➔ BERT (Bidirectional Encoder Representation from **Transformer**) family
  - Only use Transformer' Encoder → Bi-directional LM pretrain
- Autoregressive Models ➔ GPT (Generative Pre-trained **Transformer**) family

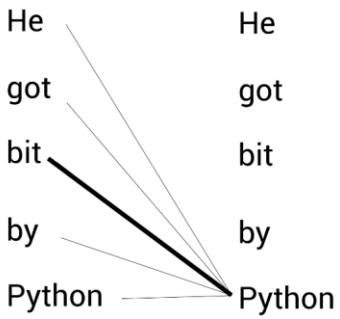
BERT's idea

- If Transformer is good, why don't we just use encoder only?
- Plus, train via bi-direction
  - Bi-directional Pre-Train!
- After pre-train → transfer learning (Fine-tuning)

What is better????

How to train???

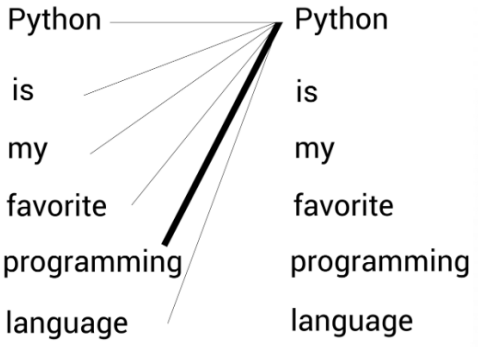
Core Idea in BERT

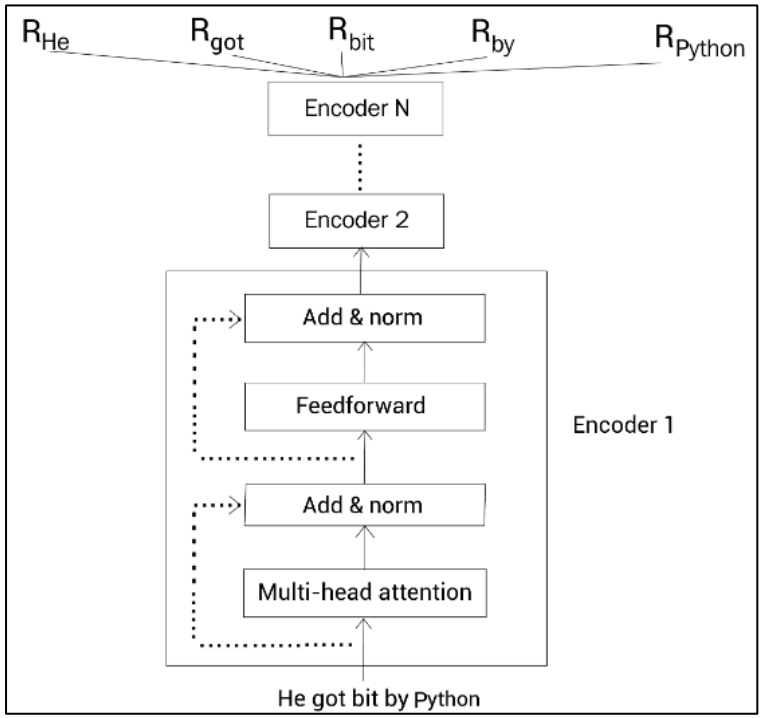


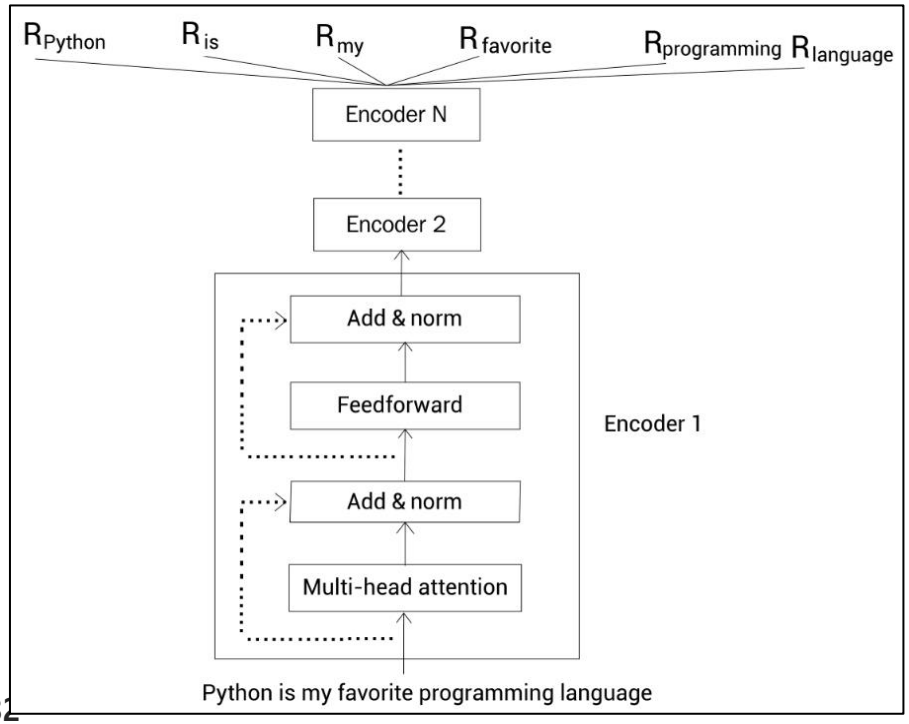
Exploiting Encoders' self-attention Mechanism

- better understanding of context
- reduce context mis-understanding via Multi-head attention

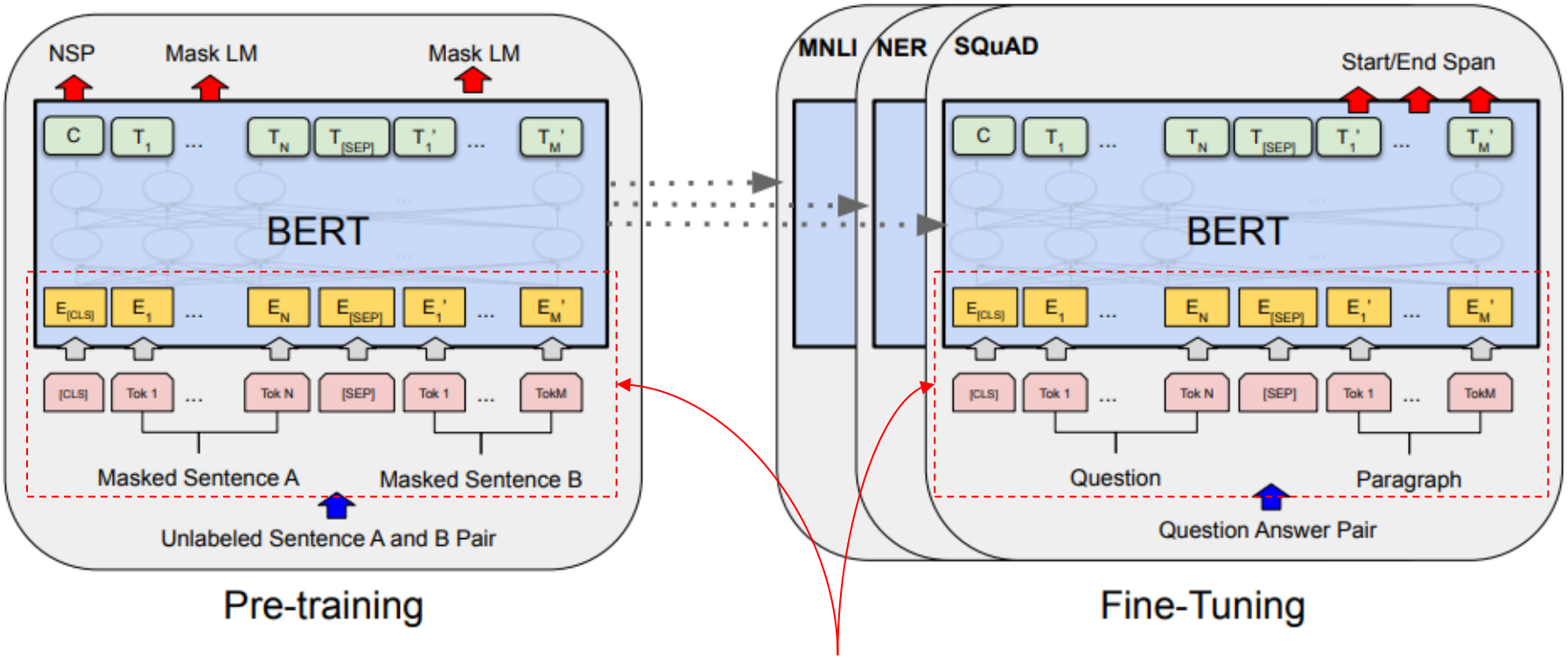
better than all existing Embedding or DNN(CNN, RNN, …) !







BERT Architecture



Characteristic:

NLU task only!

No generative task!

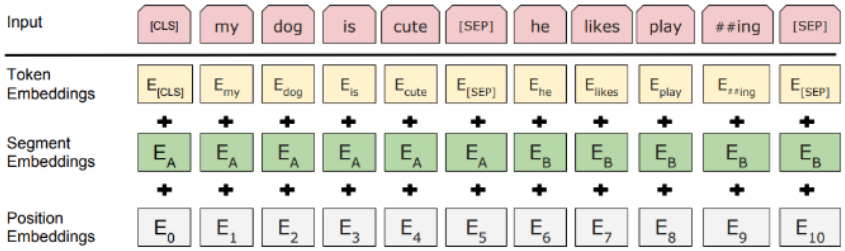
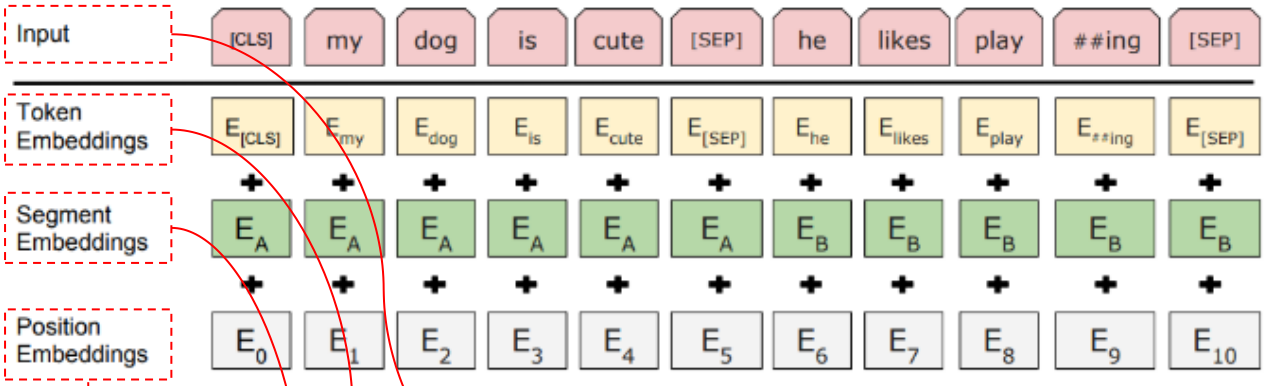


Figure 2: BERT input representation. The input embeddings is the sum of the token embeddings, the segmentation embeddings and the position embeddings.

BERT Embedding



Input: Huge dataset for Self-supervised leaning

- Add special token (CLS) at the starting location, insert SEP token where sentence is separated

Token Embeddings: Inner product of Token index & matrix

Segment Embeddings: Inner product of sentence index & matrix

Position Embeddings: Applying trigonometric function in token location

WordPiece

WordPiece 소개

- Very similar to BPE
- Difference: Choose pair with high likelihood in merging phase

$$\frac{p(st)}{p(s) \times p(t)}, where\ p\ and\ t\ is\ symbol\ pair, respectively = \frac{\text{동시에 등장할 확률}}{\text{전체 글자 중 따로 등장할 확률}}$$

Example

- Corpus: 'aab', 'ab', 'abc', 'bcd', 'b', 'c'
- Symbol pair: ('a', 'b') $= \frac{\frac{3}{13}}{\frac{4}{13} \times \frac{5}{13}} = \frac{3}{20} = 0.15$
- Symbol pair: ('b', 'c') $= \frac{\frac{2}{13}}{\frac{5}{13} \times \frac{3}{13}} = \frac{2}{10} = 0.20$

WordPiece Tokenizer

BERT's special type of tokenizer, 'WordPiece'

- WordPiece: same approach in subword tokenization

"Let us start pretraining the model."

- Step 1. check whether the word is present in vocabulary

- Step 2.
  - If word is in vocab,
    - Use it as a token
  - else,

tokens = [let, us, start, pre, ##train, ##ing, the, model]

'##' indicates it is subword and preceded by other words.

[1] Split into subword & check whether the subword in vocab

[2] If the subword in vocab, use the subword as a token

[3] else, repeat [1] until the subword is appears in vocab

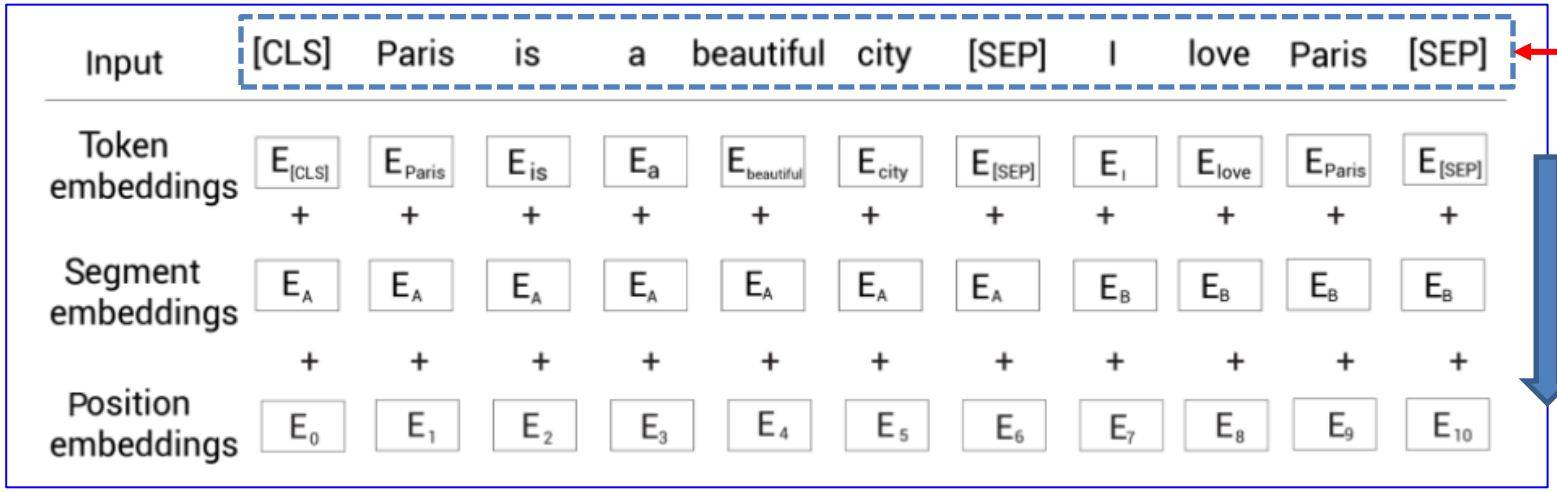
After Subword Tokenization

■ **Tokenization visualization**

"Let us start pretraining the model."

tokens = [let, us, start, pre, ##train, ##ing, the, model]

tokens = [[CLS], let, us, start, pre, ##train, ##ing, the model, [SEP]]



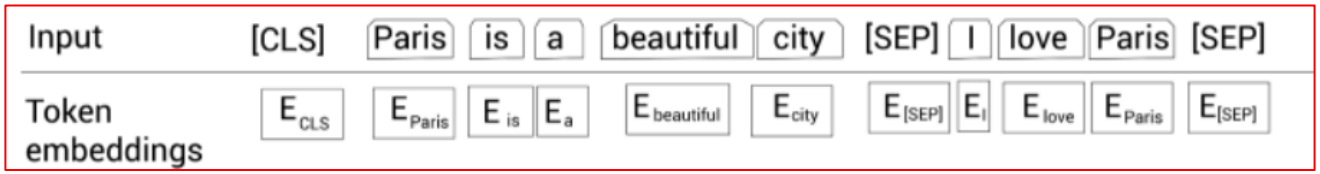
Token Embedding

The 1st layer of BERT embedding

Sentence A: Paris is a beautiful city.

Sentence B: I love Paris

- Tokenization



  - tokens = [Paris, is, a, beautiful, city, I, love, Paris]
    - Cased: maintain characters 'as is' (Upper/Lower case)
    - Uncased: transform all characters into lowercase)
- Adding **[CLS]** token ← used for classification task
  - tokens = [[CLS], Paris, is, a, beautiful, city, I, love, Paris]
- Adding **[SEP]** token ← used for NSP task
  - tokens = [[CLS], Paris, is, a, beautiful, city, [SEP], I, love, Paris, [SEP]]

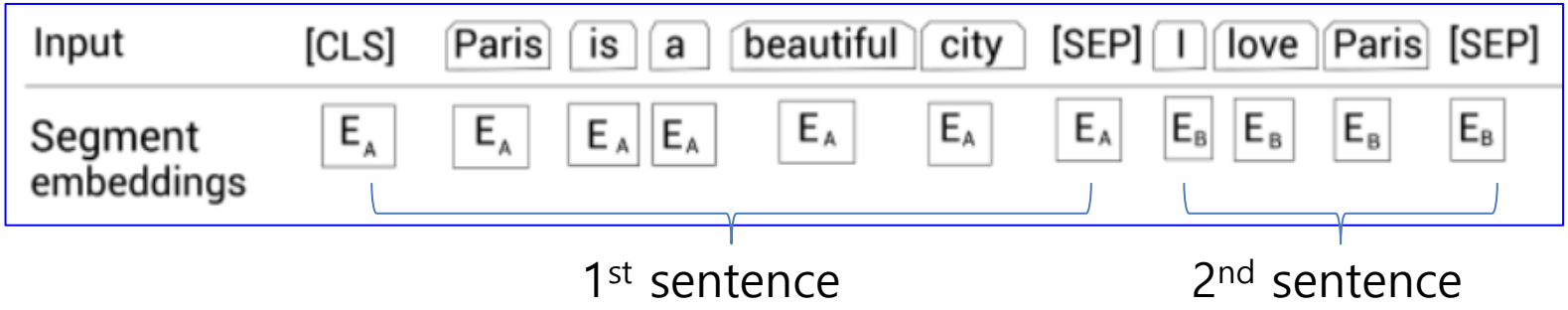
Segment Embedding

Segment Embedding

- Distinguish between two given sentences
- Feed input tokens into segment embedding layer

  tokens = [[CLS], Paris, is, a, beautiful, city, [SEP], I, love, Paris, [SEP]]

- segment embedding layer returns only either of the two embeddings, E_A and E_B
  - E_A: the 1st sentence
  - E_B: the 2nd sentence



Position Embedding

Position Embedding

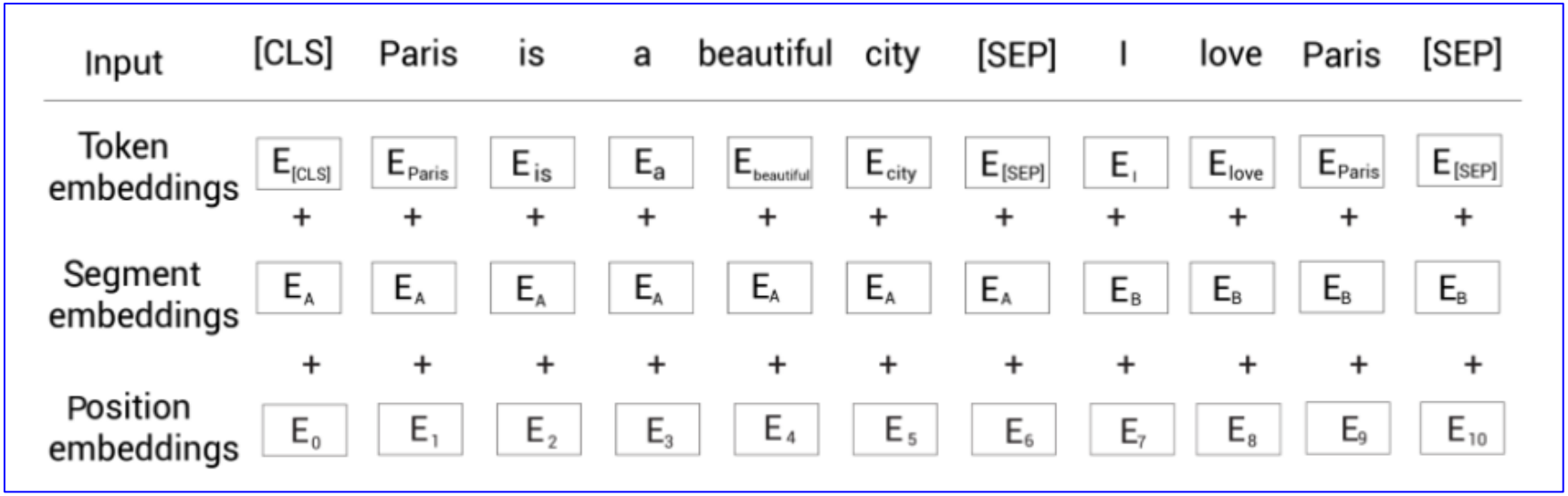
- BERT is essentially the transformer's encoder
- Transformer does not use any recurrence mechanism
- Processes all the words in parallel
- Need to provide information relating to word order

Applying Transformer's positional embedding technique!

Final Representation of Input Embedding

Sentence A: Paris is a beautiful city.

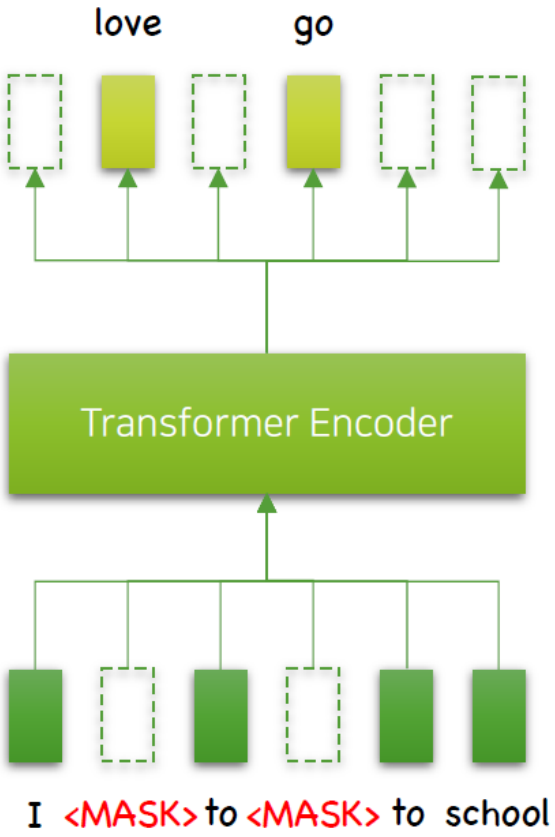
Sentence B: I love Paris



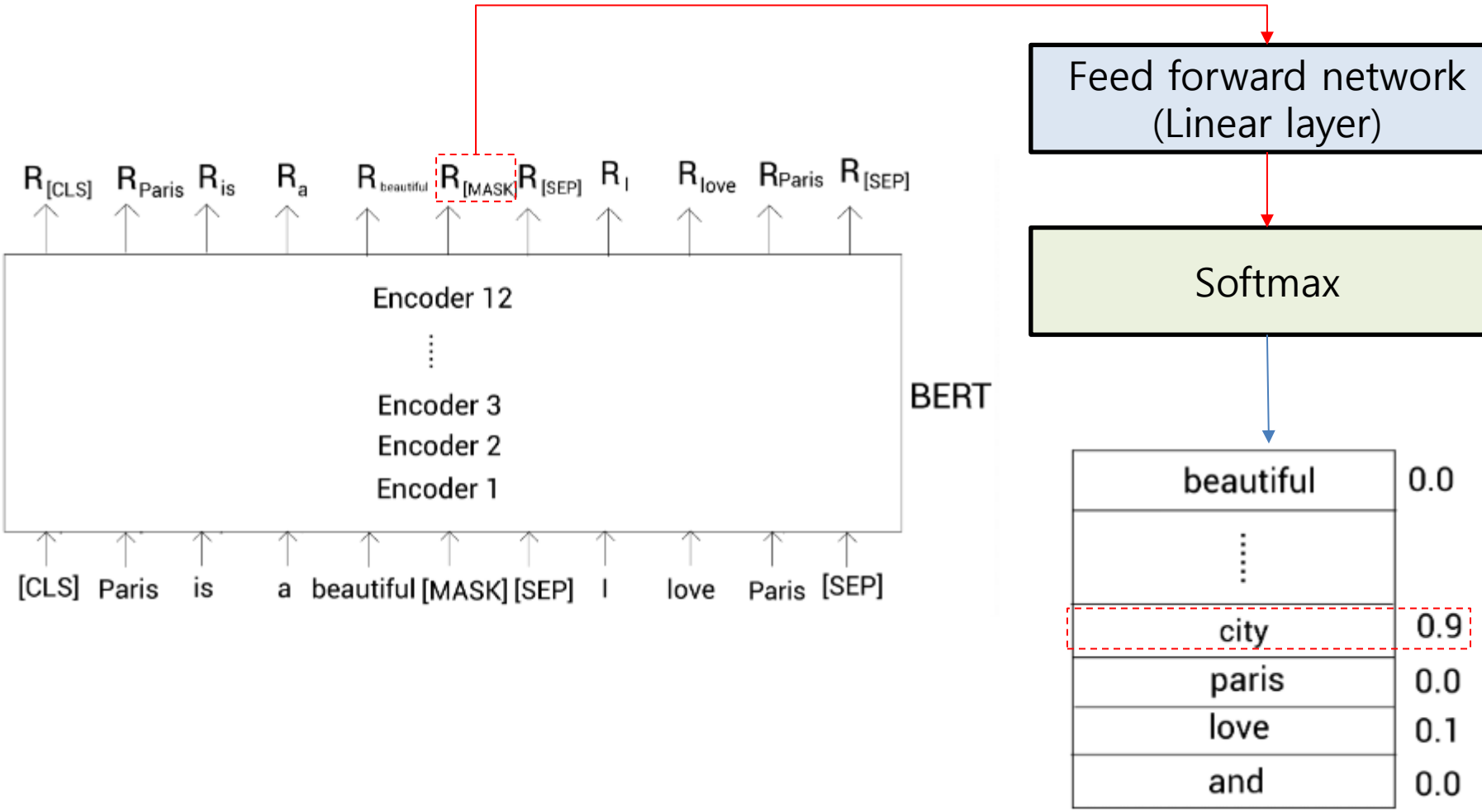
BERT pre-train: MLM

MLM (Masked Language Model)

- 일정 비율의 토큰을 숨기고, 원래 문장을 복원하도록 학습
- Next step 토큰을 예측하는 것이 아니라, 현재 step의 토큰 예측
  - Auto-regressive 속성 배제, Bi-directional 모델로 학습
- 학습(train)과 추론(inference)의 유사환경을 반영하기 위해,
  - 15%의 토큰만 추론 대상으로 선정
    - 이 중 80%를 <MASK>로 가림
    - 또한 10%를 랜덤 토큰으로 변환
    - 나머지 10%를 그대로 놔둠



BERT pre-train: MLM



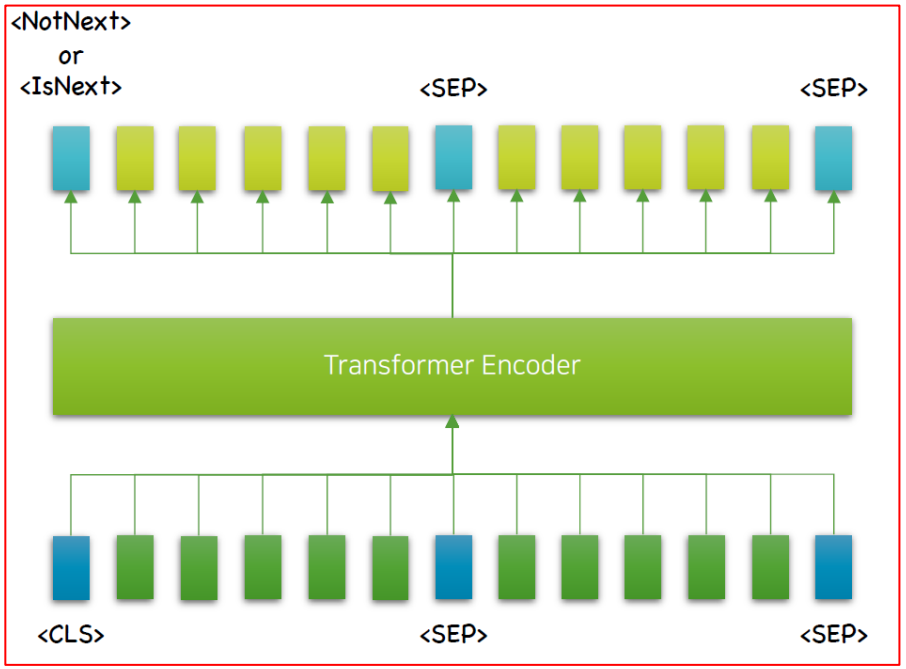
BERT pre-train: NSP

NSP (Next Sentence Prediction)

- Question Answering & Textual Entailment

  → Need to understanding the relation between sentences

- Learn two docs separated by SEP token
  - [CLS] 문장1 [SEP] 문장2 [SEP] ➔ Label: 1
  - [CLS] 문장1 [SEP] 문장k [SEP] ➔ Label: 0
  - Predict label using [CLS] token
- Proven to little performance enhancement
  - Removed in RoBERTa technology

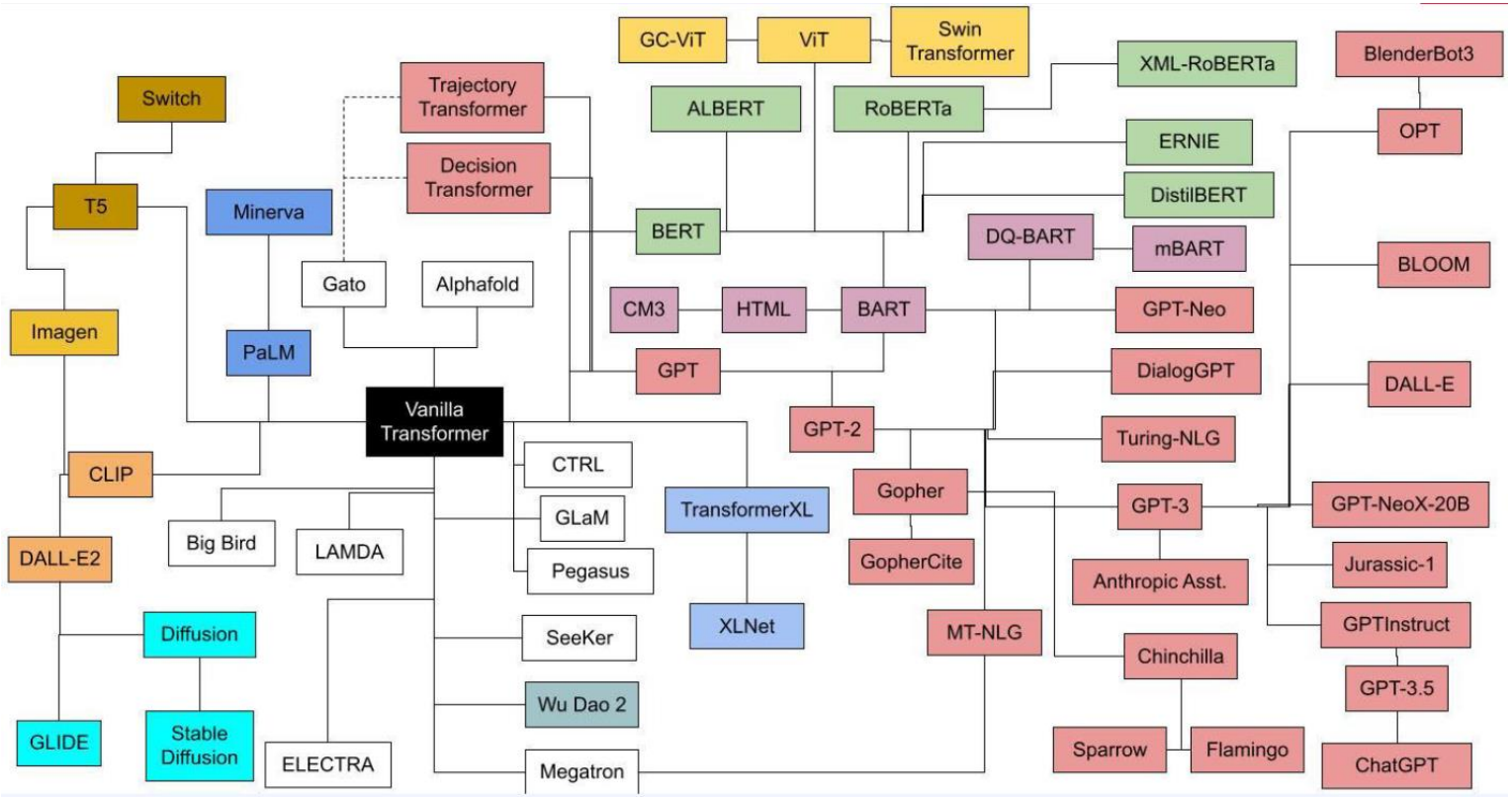


Auto-regressive Models

Family Map of Transformer

Transformer

- Encoder (BERT 등) Decoder (GPT 등) Encoder-Decoder (T5, BART 등)
- Multimodal Models (CLIP, GPT-V 등) Diffusion / Vision Transformers (Stable Diffusion 등)



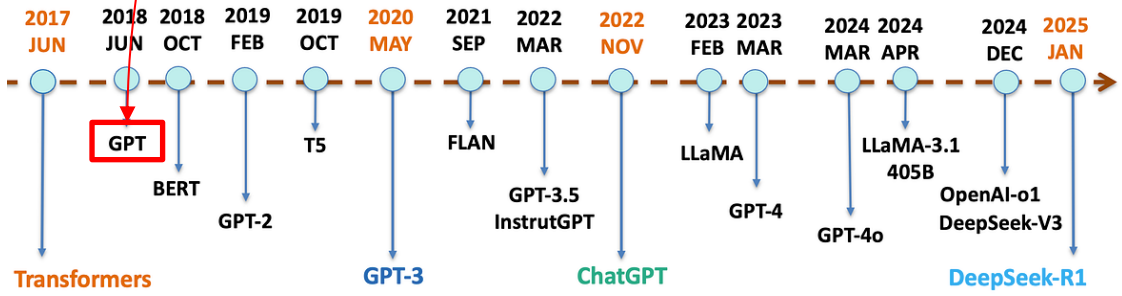
GPT Family

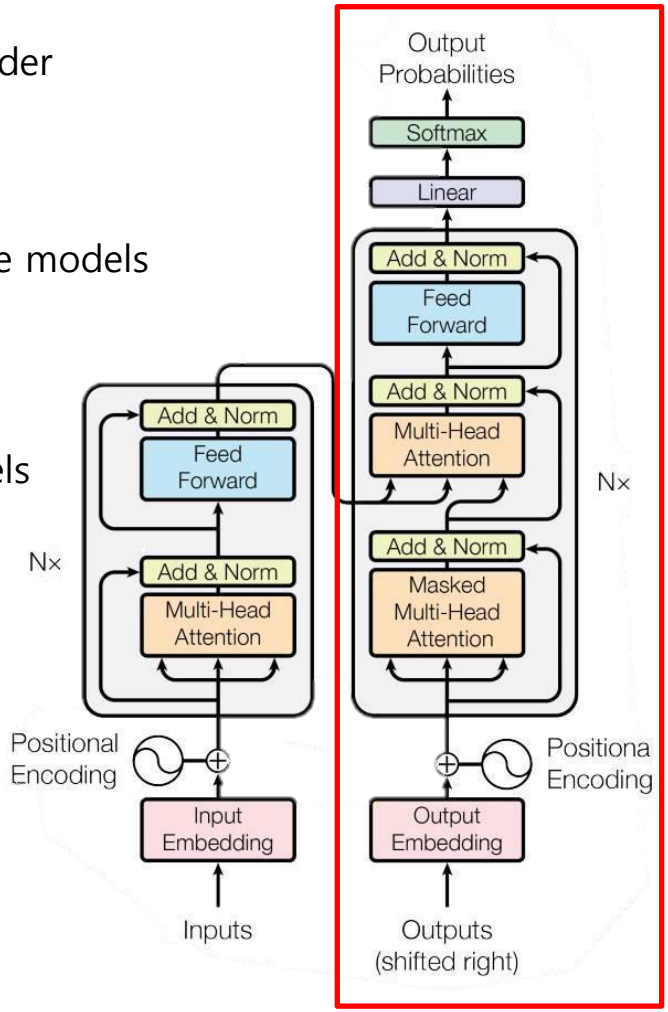
Question on BERT

- Incredible performance of Bi-directional Transformer Encoder
- However, limitation of applying auto-regressive tasks
  - Bi-directional Attention is not applicable to generative models

GPT (Generative Pre-trained Transformer)

- Only use Transformer's Decoder → build generative models
  - Only attention to previous information (token)
  - Only learn previous text





GPT1 - Overview

Main Characteristics

- Paper link: https://www.cs.ubc.ca/~amuham01/LING530/papers/radford2018improving.pdf

  [Radford et al., 2018]

- Motivation: Same as BERT
  - Model training with huge corpus
  - Downstream task: additional training for each domain dataset
- Learn BigData using Decoder → Downstream Task

Difference from BERT

- Instead of MLM, learn probability distribution
  - Maximize Likelihood

$$\mathcal{D} = \{x_i\}_{i=1}^{N},$$
$$\text{where } x_i = \{w_{i,1}, \cdots, w_{i,n}\}.$$

$$\mathcal{L}(\theta_{\text{PLM}}) = -\sum_{i=1}^{N}\sum_{t=1}^{n} \log P(w_{i,t}|w_{i,<t}; \theta_{\text{PLM}})$$

$$\hat{\theta}_{\text{PLM}} = \underset{\theta_{\text{PLM}} \in \Theta}{\text{argmin}}\ \mathcal{L}(\theta_{\text{PLM}})$$

GPT1 - Training

Auto-regressive

- Output: depending on the number of vocabulary

$$h_0 = UW_e + W_p$$
$$h_l = \texttt{transformer\_block}(h_{l-1}) \forall i \in [1, n]$$
$$P(u) = \texttt{softmax}(h_n W_e^T)$$

n: number of layers

$U = (u_{i-k}, \ldots, u_{i-1})$: context vector of tokens

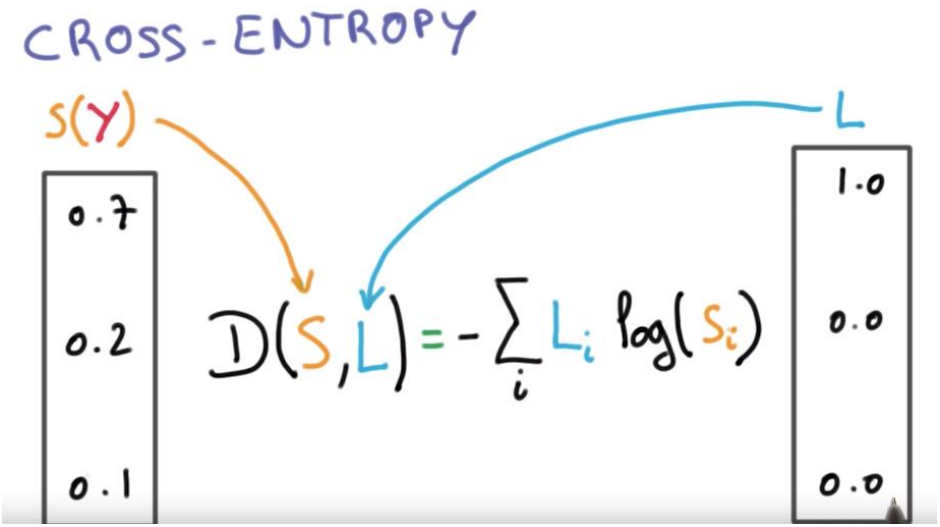
W_e: token embedding matrix

W_p: position embedding matrix

- Output dimension: $h_n W_e^T$

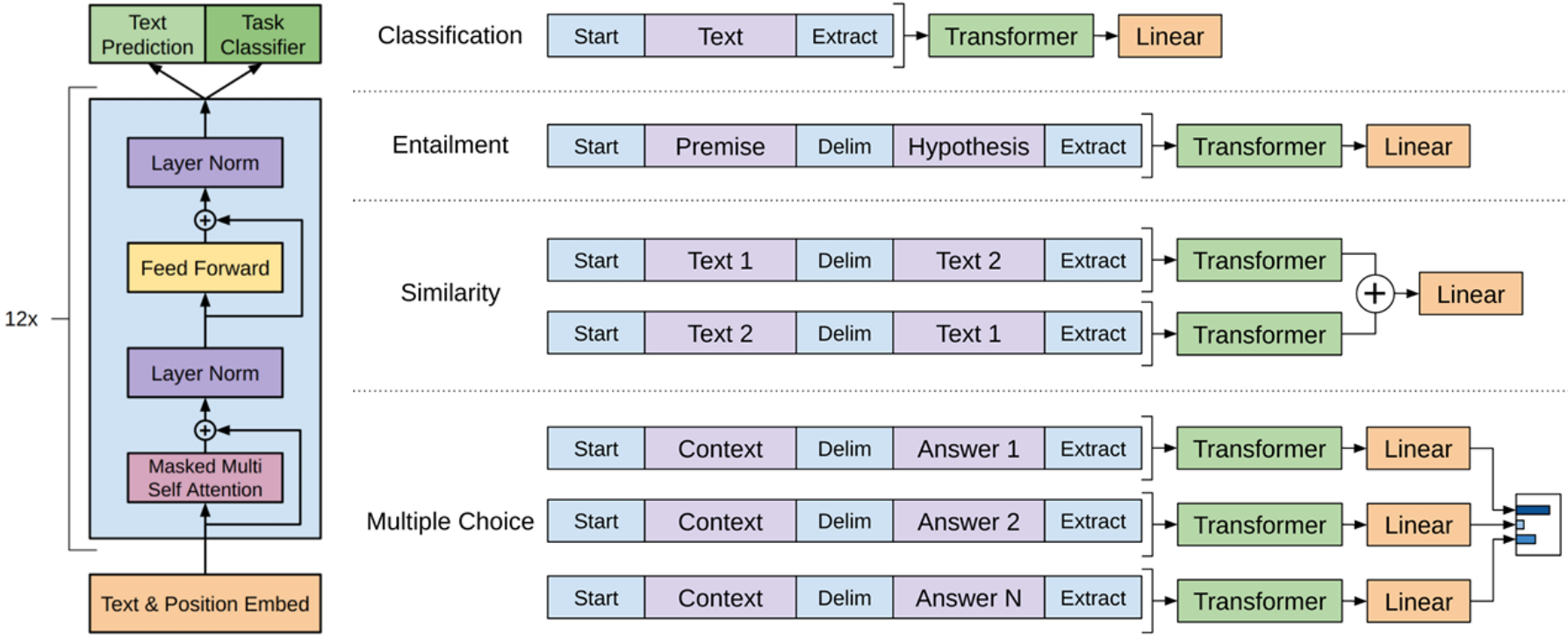
- Loss: Cross Entropy

$$L(\hat{y}, y) = -\sum_{k}^{K} y^{(k)} \log \hat{y}^{(k)}$$



이미지 출처: https://medium.com/data-science-bootcamp/understand-cross-entropy-loss-in-minutes-9fb263caee9a

GPT-1 Fine-tuning



GPT-2

Same to Transformer decoder, Nothing special!

- Paper link: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf [Radford et al., 2018]

GPT2 Characteristics

- Just increase the size of model

	GPT1	GPT2
# Parameters	1억 17백만(117M)	15억(1.5B)
# Layer	12	48
Batch Size	64	512
Etc.		Change the location of Layer Normalization (Maybe after experiments)

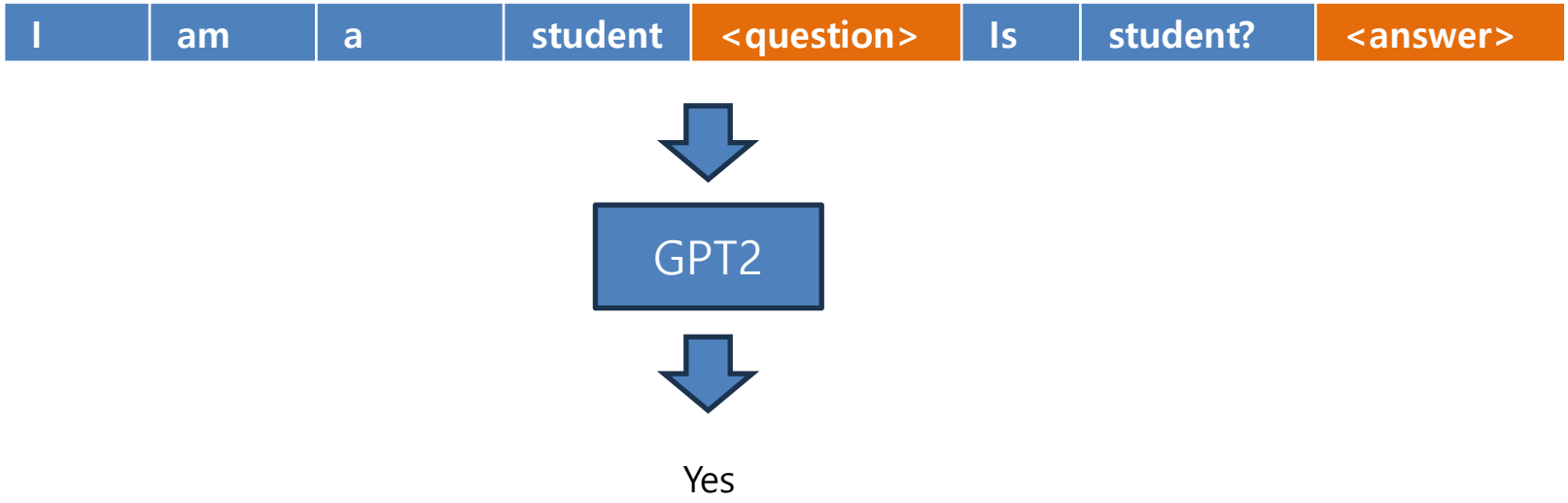
- Removing Fine-tuning in GPT-1

→ In the actual fields, fine-tuning is added because of low performance of Zero-shot

GPT2 – removing fine-tuning

How to remove fine-tuning in GPT2

- Special Token을 활용한 학습



Auto-regressive: Just use given data as label

GPT2 – Model Size & Performance

GPT2 model types

- Small
- Medium
- Large (GPT2)

Parameters	Layers	d_{model}
117M	12	768
345M	24	1024
762M	36	1280
1542M	48	1600

Table 2. Architecture hyperparameters for the 4 model sizes.

- LAMBADA: 긴 문장을 얼마나 정확하게 재생한 하는지 테스트
- CBT(Children's Book Test): 교재에 바진 개체명이나 명사를 얼마나 정확히 예측하는지 테스트
- WikiText2: 특정 개념에 대한 설명을 얼마나 잘 하는지 테스트

:

BPB = Bits Per Byte
BPC = Bits Per Character
모델이 평균적으로 한 글자를 표현하는 데 필요한 정보량, 낮을수록 좋은 모델

Zero-shot Performance

Language Models are Unsupervised Multitask Learners

	LAMBADA (PPL)	LAMBADA (ACC)	CBT-CN (ACC)	CBT-NE (ACC)	WikiText2 (PPL)	PTB (PPL)	enwik8 (BPB)	text8 (BPC)	WikiText103 (PPL)	1BW (PPL)
SOTA	99.8	59.23	85.7	82.3	39.14	46.54	0.99	1.08	18.3	**21.8**
117M	**35.13**	45.99	**87.65**	**83.4**	**29.41**	65.85	1.16	1.17	37.50	75.20
345M	**15.60**	55.48	**92.35**	**87.1**	**22.76**	47.33	1.01	**1.06**	26.37	55.72
762M	**10.87**	**60.12**	**93.45**	**88.0**	**19.93**	**40.31**	**0.97**	**1.02**	22.05	44.575
1542M	**8.63**	**63.24**	**93.30**	**89.05**	**18.34**	**35.76**	**0.93**	**0.98**	**17.48**	42.16

GPT2 모델 공개

OpenAI did not open GPT2 model: Malicious usage of GPT2

- At the time of GPT2 opening,
  - Q&A, Reading, Summary, Translation → Top performance!
- Zero-shot is possible without additional training

GPT2 model is open to public

- Huggingface: https://huggingface.co/models?sort=trending&search=gpt2

GPT3 ...

GPT3, what's different?

- Almost same as existing GPT based on Transformer
- Attention → Sparse Attention
- Increase the number of tokens
- Increase the number of parameters: 15억 (1.5B) → 1,750억 (175 B)

Characteristics

- More focus on zero-shot learning
- Advanced Few-shot setting
  - Few-show: give some example → predict answer
  - Show the possibility of ChatGPT
- GPT3 is NOT open to public

GPT-2가 공개 사유

- 위험성 우려보다 연구적 가치가 더 크다고 판단
- community 발전에 기여

GPT-3가 비공개 사유

- 악용 위험성 증가:
  - 스팸/피싱 글 자동 생성
  - 가짜 뉴스 생성
  - 사기 문장 자동화
- 모델 크기가 너무 커서 일반 컴퓨터에서 학습/서빙 불가

GPT3 Dataset

Applying BBPE

- Processing all languages
- White space is added between different alphabet (e.g.: Korean, English)

Use 'Common Crawl' dataset

- https://commoncrawl.org/
- After preprocessing, reduce the dataset size: 45TB → 570GB

Disadvantage of crawling dataset: More learning with better quality of sentences

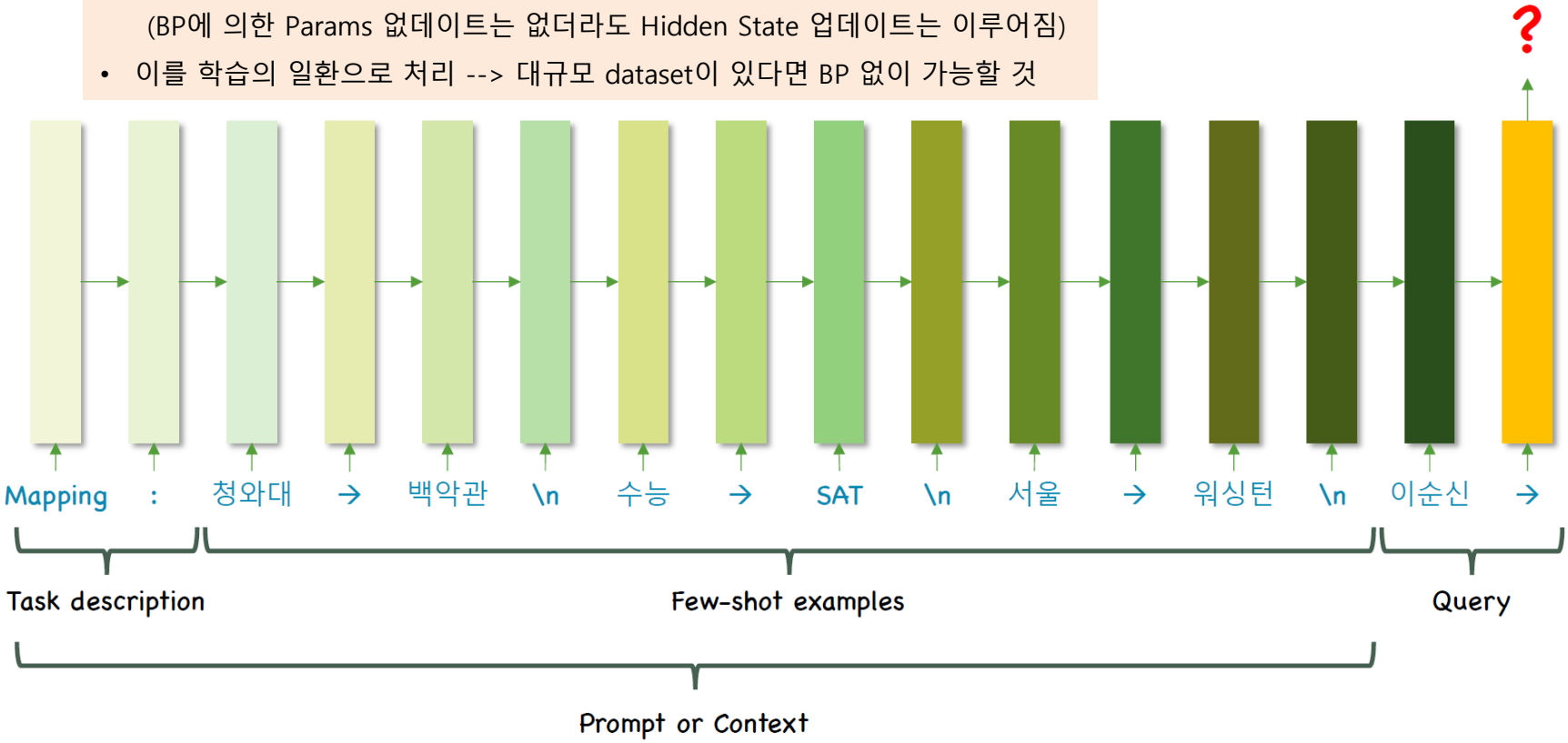
Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

GPT3 w/o fine-tuning?

In Context Learning

- Learning without Back-propagation

- 이전에 주어진 각 단어 입력의 representation에는 이전 정보를 담고 있을 것임 (BP에 의한 Params 없데이트는 없더라도 Hidden State 업데이트는 이루어짐)
- 이를 학습의 일환으로 처리 --> 대규모 dataset이 있다면 BP 없이 가능할 것



자료 출처: 패스트 캠퍼스, '김기현의 BERT, GPT를 활용한 자연어처리' 강의 자료 발췌

GPT3 In-context Learning

In-context Learning?

$$P(output \mid input, prompt)$$

If model learn language? (GPT3 back-born is well trained)

- Can complete sentence using given context
  - Only use prompt and given input
- Theory of "next word prediction is possible"

If train dataset is big & model size is large,

➔ No Downstream-task ➔ No Fine-tuning

GPT3: Few-show, One-shot, Zero-shot

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
Translate English to French:        ← task description
sea otter => loutre de mer          ← examples
peppermint => menthe poivrée        ←
plush girafe => girafe peluche      ←
cheese =>                           ← prompt
```

One-shot

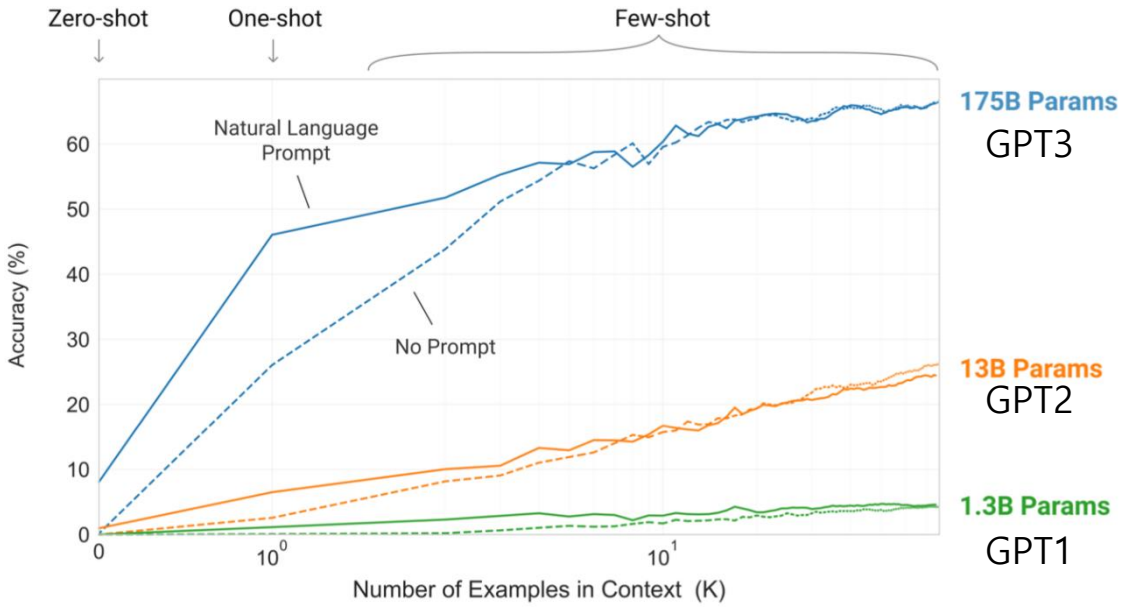
In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

```
Translate English to French:        ← task description
sea otter => loutre de mer          ← example
cheese =>                           ← prompt
```

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
Translate English to French:        ← task description
cheese =>                           ← prompt
```



As model size larger,

the effect of in-context learning is increased

Also, zero-shot performance is increased

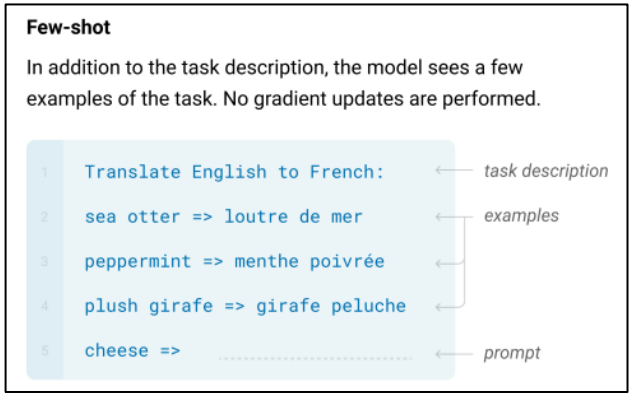
Disadvantage: In-context Learning

Prompt configuration

- In-context Learning → No Fine-tuning needed!
- But, user must make 'Prompt' by himself/herself → depending on Few-shots → different output

  - Translate English to Korean:
    - I want to eat. ==> 나는 배고프다.
  - Translate English from Korean sentence:
    - I want to eat. ==> 나는 배고프다.

Same results are expected,
but different results can be occurred.



- Majority Label Bias: Bias to focus on Major classes
- Recency Bias: Bias to select recent results
- Common Token Bias: Bias to select most frequent classes

Prompt Engineering

■ **If you want 'Few-show Learning'**

➔ Be aware of disadvantage of In-context Learning

- Weak point in Prompt (Task Description) change
- Need to human intervention to find "GOOD Prompt"
  - In other words, In-context learning is still low performance compared to Fine-tuning
- Research type of finding "GOOD prompt" without human intervention

  → So called, "Prompt Engineering"
  - GPT3 Prompt Engineering

GPT3 Limitation

- **Not exactly understand real world**
- **Not excellent in all areas**
  - Hi performance in General knowledge
- **Since not Bi-directional, NLU performance is little bit low**
- **Hallucination**
- **Environment problem**
- **Only remember previous information**
- **Still very low performance compared to human**

ChatGPT

ChatGPT

GPT3.5 == InstructGPT == ChatGPT

- 2022년 11월: ChatGPT (GPT3.5) launched ← InstructGPT: ChatGPT' predecessor
- Change the learning approach of GPT3

  → ChatGPT (GPT3' Fine-tuned version: learn chatting)
- Blog: https://openai.com/blog/chatgpt

Explosive attention from public ➔ Is it real? Is it possible? really?

Time it took to reach 100 million users worldwide:
- Telephone: 75 years
- Mobile phone: 16 years
- World Wide Web: 7 years
- iTunes: 6.5 years
- Twitter: 5 years
- Facebook: 4.5 years
- WhatsApp: 3.5 years
- Instagram: 2.5 years
- Apple App Store: 2 years
- ChatGPT: 2 months

Data source:
https://www.facebook.com/groups/ChatbotDevKR/permalink/1646183582466431/?sfnsn=mo&ref=share&mibextid=LROouL

InstructGPT

Problem of Large LM

- Untruthful, Toxic → harmful information to users

InstructGPT

- Known as the predecessor of ChatGPT
- Via fine-tuning, aligning model to human intentions
- Paper link: https://cdn.openai.com/papers/Training_language_models_to_follow_instructions_with_human_feedback.pdf
- Reinforce Learning from Human Feedback (RLHF)
  - Reinforce learning based on human feedback
- Dataset
  - Prompt data submitted to OpenAI API
  - Mainly, collect prompt data in OpenAI playground (https://platform.openai.com/playground)

InstructGPT

Step 1

Collect demonstration data, and train a supervised policy.

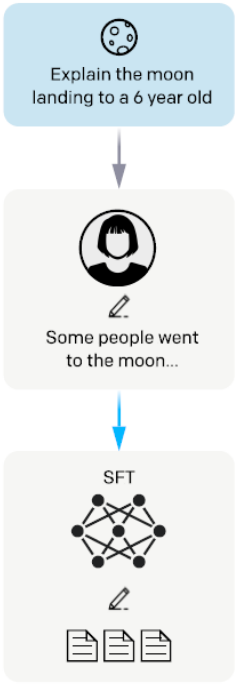
A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

Labeler 40명 고용

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

GPT3 백본 네트워크 Fine-tuning



Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

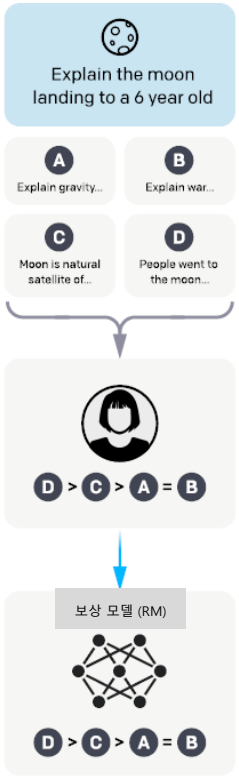
하나의 prompt 대한 여러 개 답변 출력

A labeler ranks the outputs from best to worst.

Labeler는 좋은 답변 순위를 결정

This data is used to train our reward model.

강화학습 보상 모델을 학습



Step 3

보상(Reward) 최대화 되도록 PPO 학습

Optimize a policy against the reward model using reinforcement learning.

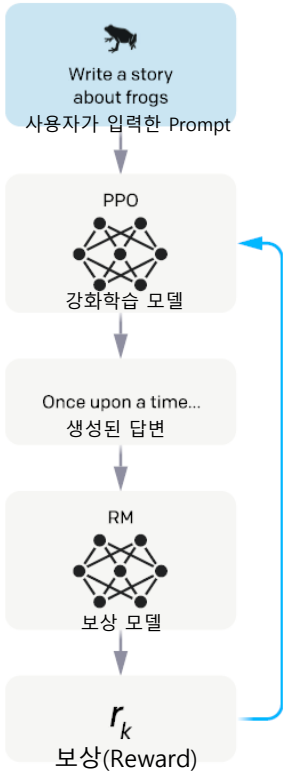
A new prompt is sampled from the dataset.

The policy generates an output.

PPO: Proximal Policy Optimization
(강화학습 알고리즘 중 하나)

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



InstructGPT vs. ChatGPT

Step 1, Step 2는 동일

InstructGPT

ChatGPT

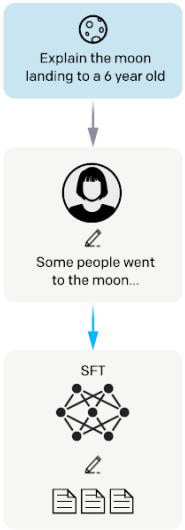
Step 1

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.



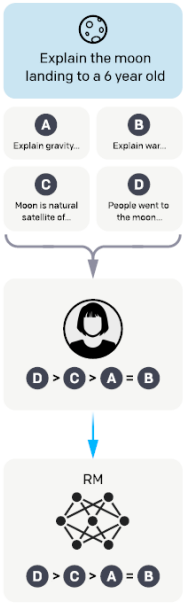
Step 2

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

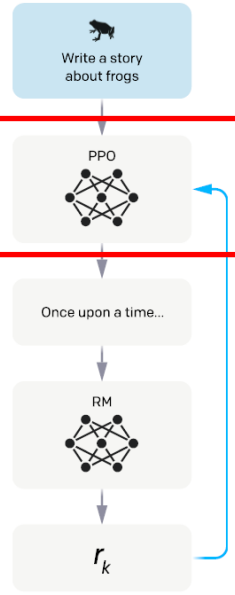
Optimize a policy against the reward model using reinforcement learning.

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.



Step 3

Optimize a policy against the reward model using the PPO reinforcement learning algorithm.

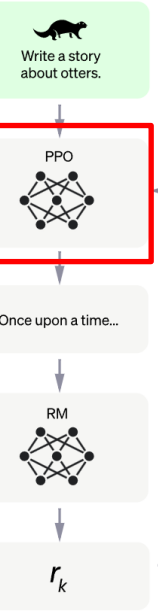
A new prompt is sampled from the dataset.

The PPO model is initialized from the supervised policy.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



GPT4

GPT4

GPT4 공개

- 2023. 3. 15. open (OpenAI's speed of upgrading models...)
- Provide Technical Report
    - Only provide concept & performance evaluations
- No detailed model explanation
- Technical report: https://arxiv.org/abs/2303.08774

Characteristics

- Multi-modal
    - Input: text + image
    - output: text

GPT4 – 성능 향상

Achieve human capability in various benchmarks

- GPT-3.5 (ChatGPT): low 10% of various test
- GPT-4: top 10% of various test (ex: lawyer test etc.)

Characteristics

- Not open to public → OpenAI ?? ClosedAI??
  - Transformer pre-trained model + RLHF tuningin
  - Model size: about 100 Trillion… (Not sure yet)
- Complete to combine MS Bing + GPT4
- Longer input sequence
- Reduce Hallucination
- Better performance in various languages

GPT4 – 성능 향상

Exam	GPT-4	GPT-4 (no vision)	GPT-3.5
Uniform Bar Exam (MBE+MEE+MPT)	298 / 400 (~90th)	298 / 400 (~90th)	213 / 400 (~10th)
LSAT	163 (~88th)	161 (~83rd)	149 (~40th)
SAT Evidence-Based Reading & Writing	710 / 800 (~93rd)	710 / 800 (~93rd)	670 / 800 (~87th)
SAT Math	700 / 800 (~89th)	690 / 800 (~89th)	590 / 800 (~70th)
Graduate Record Examination (GRE) Quantitative	163 / 170 (~80th)	157 / 170 (~62nd)	147 / 170 (~25th)
Graduate Record Examination (GRE) Verbal	169 / 170 (~99th)	165 / 170 (~96th)	154 / 170 (~63rd)
Graduate Record Examination (GRE) Writing	4 / 6 (~54th)	4 / 6 (~54th)	4 / 6 (~54th)
USABO Semifinal Exam 2020	87 / 150 (99th - 100th)	87 / 150 (99th - 100th)	43 / 150 (31st - 33rd)
USNCO Local Section Exam 2022	36 / 60	38 / 60	24 / 60
Medical Knowledge Self-Assessment Program	75 %	75 %	53 %
Codeforces Rating	392 (below 5th)	392 (below 5th)	260 (below 5th)
AP Art History	5 (86th - 100th)	5 (86th - 100th)	5 (86th - 100th)
AP Biology	5 (85th - 100th)	5 (85th - 100th)	4 (62nd - 85th)
AP Calculus BC	4 (43rd - 59th)	4 (43rd - 59th)	1 (0th - 7th)
AP Chemistry	4 (71st - 88th)	4 (71st - 88th)	2 (22nd - 46th)
AP English Language and Composition	2 (14th - 44th)	2 (14th - 44th)	2 (14th - 44th)
AP English Literature and Composition	2 (8th - 22nd)	2 (8th - 22nd)	2 (8th - 22nd)
AP Environmental Science	5 (91st - 100th)	5 (91st - 100th)	5 (91st - 100th)
AP Macroeconomics	5 (84th - 100th)	5 (84th - 100th)	2 (33rd - 48th)
AP Microeconomics	5 (82nd - 100th)	4 (60th - 82nd)	4 (60th - 82nd)
AP Physics 2	4 (66th - 84th)	4 (66th - 84th)	3 (30th - 66th)
AP Psychology	5 (83rd - 100th)	5 (83rd - 100th)	5 (83rd - 100th)
AP Statistics	5 (85th - 100th)	5 (85th - 100th)	3 (40th - 63rd)
AP US Government	5 (88th - 100th)	5 (88th - 100th)	4 (77th - 88th)
AP US History	5 (89th - 100th)	4 (74th - 89th)	4 (74th - 89th)
AP World History	4 (65th - 87th)	4 (65th - 87th)	4 (65th - 87th)
AMC 10^3	30 / 150 (6th - 12th)	36 / 150 (10th - 19th)	36 / 150 (10th - 19th)
AMC 12^3	60 / 150 (45th - 66th)	48 / 150 (19th - 40th)	30 / 150 (4th - 8th)
Introductory Sommelier (theory knowledge)	92 %	92 %	80 %
Certified Sommelier (theory knowledge)	86 %	86 %	58 %
Advanced Sommelier (theory knowledge)	77 %	77 %	46 %
Leetcode (easy)	31 / 41	31 / 41	12 / 41
Leetcode (medium)	21 / 80	21 / 80	8 / 80
Leetcode (hard)	3 / 45	3 / 45	0 / 45

Table 1. GPT performance on academic and professional exams. In each case, we simulate the conditions and scoring of the real exam. We report GPT-4's final score graded according to exam-specific rubrics, as well as the percentile of test-takers achieving GPT-4's score.

	GPT-4 Evaluated few-shot	GPT-3.5 Evaluated few-shot	LM SOTA Best external LM evaluated few-shot	SOTA Best external model (incl. benchmark-specific tuning)
MMLU [49] Multiple-choice questions in 57 subjects (professional & academic)	**86.4%** 5-shot	70.0% 5-shot	70.7% 5-shot U-PaLM [50]	75.2% 5-shot Flan-PaLM [51]
HellaSwag [52] Commonsense reasoning around everyday events	**95.3%** 10-shot	85.5% 10-shot	84.2% LLaMA (validation set) [28]	85.6 ALUM [53]
AI2 Reasoning Challenge (ARC) [54] Grade-school multiple choice science questions. Challenge-set.	**96.3%** 25-shot	85.2% 25-shot	85.2% 8-shot PaLM [55]	86.5% ST-MOE [18]
WinoGrande [56] Commonsense reasoning around pronoun resolution	**87.5%** 5-shot	81.6% 5-shot	85.1% 5-shot PaLM [3]	85.1% 5-shot PaLM [3]
HumanEval [43] Python coding tasks	**67.0%** 0-shot	48.1% 0-shot	26.2% 0-shot PaLM [3]	65.8% CodeT + GPT-3.5 [57]
DROP [58] (F1 score) Reading comprehension & arithmetic.	80.9 3-shot	64.1 3-shot	70.8 1-shot PaLM [3]	**88.4** QDGAT [59]
GSM-8K [60] Grade-school mathematics questions	**92.0%*** 5-shot chain-of-thought	57.1% 5-shot	58.8% 8-shot Minerva [61]	87.3% Chinchilla + SFT+ORM-RL, ORM reranking [62]

Table 2. Performance of GPT-4 on academic benchmarks. We compare GPT-4 alongside the best SOTA (with benchmark-specific training) and the best SOTA for an LM evaluated few-shot. GPT-4 outperforms existing LMs on all benchmarks, and beats SOTA with benchmark-specific training on all datasets except DROP. For each task we report GPT-4's performance along with the few-shot method used to evaluate. For GSM-8K, we included part of the training set in the GPT-4 pre-training mix (see Appendix E), and we use chain-of-thought prompting [11] when evaluating. For multiple-choice questions, we present all answers (ABCD) to the model and ask it to choose the letter of the answer, similarly to how a human would solve such a problem.

GPT4 – Visual Input

Example of GPT-4 visual input:

User — What is funny about this image? Describe it panel by panel.

Source: https://www.reddit.com/r/hmmm/comments/ubab5v/hmmm/

GPT-4 — The image shows a package for a "Lightning Cable" adapter with three panels.

Panel 1: A smartphone with a VGA connector (a large, blue, 15-pin connector typically used for computer monitors) plugged into its charging port.

Panel 2: The package for the "Lightning Cable" adapter with a picture of a VGA connector on it.

Panel 3: A close-up of the VGA connector with a small Lightning connector (used for charging iPhones and other Apple devices) at the end.

The humor in this image comes from the absurdity of plugging a large, outdated VGA connector into a small, modern smartphone charging port.

Table 3. Example prompt demonstrating GPT-4's visual input capability. The prompt consists of a question about an image with multiple panels which GPT-4 is able to answer.

GPT-4 visual input example, Extreme Ironing:

User — What is unusual about this image?

Source: https://www.barnorama.com/wp-content/uploads/2016/12/03-Confusing-Pictures.jpg

GPT-4 — The unusual thing about this image is that a man is ironing clothes on an ironing board attached to the roof of a moving taxi.

Table 16. Example prompt demonstrating GPT-4's visual input capability. The prompt requires image understanding.

GPT4 – Visual Input

GPT-4 visual input example, Pixel to Paper Summaries:

User Below is part of the InstuctGPT paper. Could you read and summarize it to me?


Training language models to follow instructions with human feedback

Long Ouyang* Jeff Wu* Xu Jiang* Diogo Almeida* Carroll L. Wainwright* Pamela Mishkin* Chong Zhang Sandhini Agarwal Katarina Slama Alex Ray John Schulman Jacob Hilton Fraser Kelton Luke Miller Maddie Simens Amanda Askell† Peter Welinder Paul Christiano*† Jan Leike* Ryan Lowe*

OpenAI

Abstract

1 Introduction


Figure 1: Human evaluations of various models on our API prompt distribution, evaluated by how often outputs from each model were preferred to those from the 175B SFT model. Our InstructGPT models (PPO-ptx) as well as its variant trained without pretraining mix (PPO) significantly outperform the GPT-3 baselines (GPT, GPT prompted); outputs from our 1.3B PPO-ptx model are preferred to those from the 175B GPT-3. Error bars throughout the paper are 95% confidence intervals.

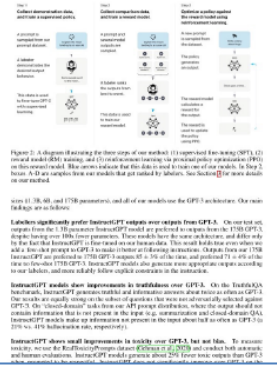


Figure 2: A diagram illustrating the three steps of our method: (1) supervised fine-tuning (SFT), (2) reward model (RM) training, and (3) reinforcement learning via proximal policy optimization (PPO) on this reward model. Blue arrows indicate that this data is used to train one of our models. In Step 2, boxes A-D are samples from our models that get ranked by labelers. See Section 3 for more details on our method.

GPT-4 The InstructGPT paper focuses on training large language mod
with human feedback. The authors note that making languag
inherently make them better at following a user's intent. Large mo
that are untruthful, toxic, or simply unhelpful.

To address this issue, the authors fine-tune language mode
tasks using human feedback. They start with a set of labeler-writt
then collect a dataset of labeler demonstrations of the desire
fine-tune GPT-3 using supervised learning and then use reinforcen
feedback to further fine-tune the model. The resulting model, c
improvements in truthfulness and reductions in toxic output
minimal performance regressions on public NLP datasets.

The authors conclude that fine-tuning with human feedbac
tion for aligning language models with human intent.

User Could you explain the process described in Figure 2?

GPT-4 Figure 2 in the InstructGPT paper illustrates the three steps of their method:

1. Supervised fine-tuning (SFT): They collect demonstration data, which consists of labeler-written prompts and desired output behavior. This data is used to fine-tune GPT-3 with supervised learning.

2. Reward model (RM) training: They collect comparison data, where a labeler ranks the outputs from several model samples (from best to worst). This data is used to train the reward model.

3. Reinforcement learning using Proximal Policy Optimization (PPO): They optimize a policy against the reward model using reinforcement learning. For each new prompt sampled from the dataset, the policy generates an output. The reward model calculates a reward for the output, and the reward is used to update the policy using the PPO algorithm.

These three steps are iteratively performed to train the InstructGPT model, which aims to generate outputs that align better with human preferences and follow instructions more effectively.

GPT4 – Visual Input

GPT-4 visual input example, École Polytechnique Exam Question:

User Answer question I.1.a. Think step-by-step.

I. Principe de la détection de rayonnement avec un bolomètre

Comme illustré sur la figure 1 un bolomètre est constitué d'un absorbeur qui reçoit le rayonnement que l'on désire détecter. Sa température T, supposée uniforme, est mesurée à l'aide d'un thermomètre incorporé, constitué d'un matériau conducteur dont la résistance $R(T)$ varie avec la température T ; cette variation est caractérisée par le coefficient $\alpha = \frac{1}{R}\frac{dR}{dT}$. L'ensemble possède la capacité thermique C_{th}.

Un barreau, conducteur thermique, homogène, de longueur L, de section S et de conductivité thermique λ et sans échanges thermiques latéraux, relie le bolomètre à un thermostat de température T_b fixe.

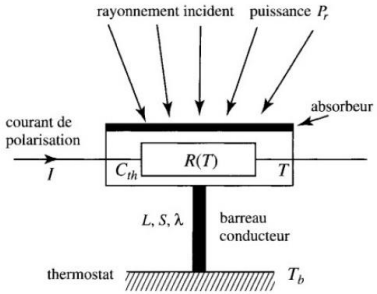


Figure 1 - Schéma d'un bolomètre

I.1. Dans cette première partie, on étudie le comportement du bolomètre en régime stationnaire, à la température de fonctionnement $T = T_0$.

I.1.a) Exprimer la température $T(x)$ en tout point du barreau conducteur repéré par son abscisse x ; on prendra $T(0) = T_0$ et $T(L) = T_b$.

GPT-4 To answer question I.1.a, we need to express the temperature $T(x)$ in every point of the conductive bar, which is characterized by the abscissa x. We are given that $T(0) = T0$ and $T(L) = Tb$.

For a stationary regime, the heat equation for the one-dimensional heat conduction in the bar is:
$d^2T/dx^2 = 0$

Integrating once, we get:
$dT/dx = C_1$

Integrating a second time:
$T(x) = C_1x + C_2$

To determine the constants C_1 and C_2, we use the boundary conditions:
$T(0) = T0$, which gives $C_2 = T0$
$T(L) = Tb$, which gives $C_1 = (Tb - T0)/L$

Finally, the expression for $T(x)$ is:
$T(x) = (Tb - T0) * (x/L) + T0$
This represents a linear temperature profile along the conductive bar, with a slope of $(Tb - T0)/L$.

GPT4 – Multi-language

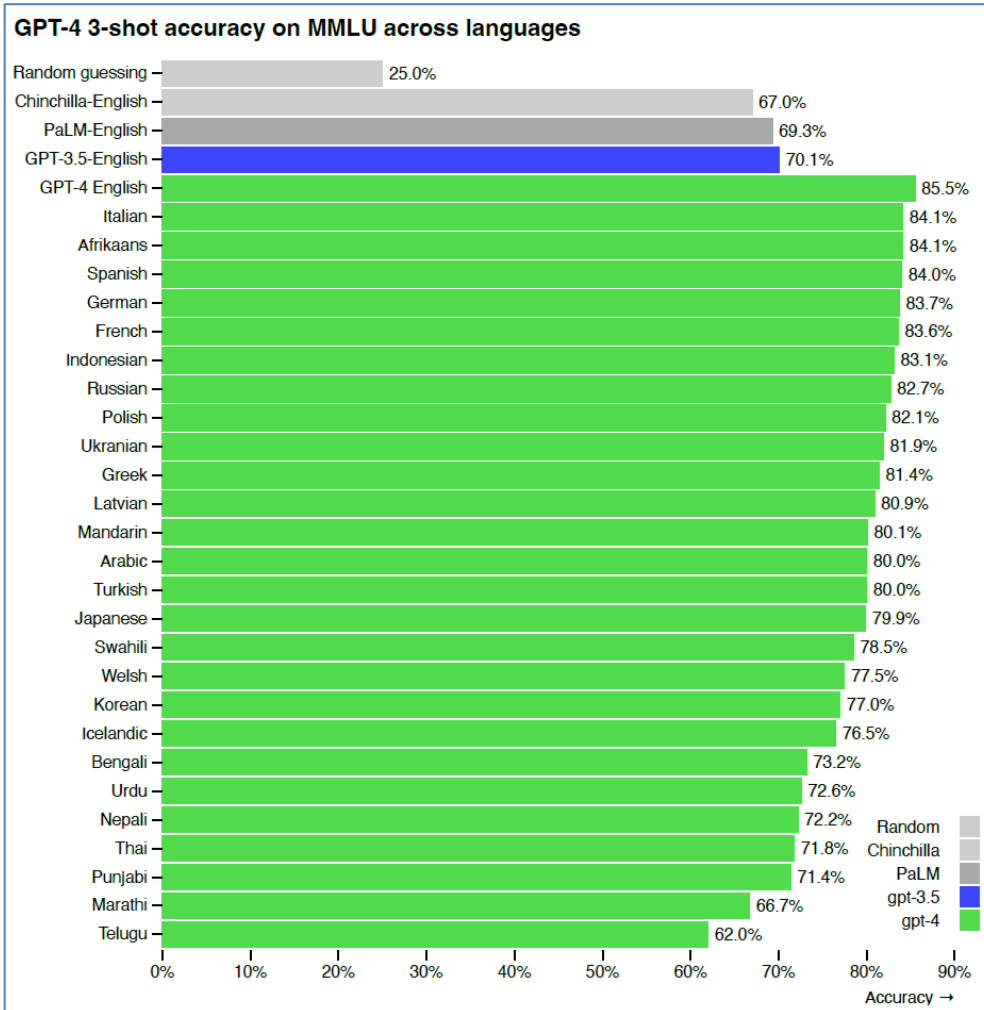


Figure 5. Performance of GPT-4 in a variety of languages compared to prior models in English on MMLU. GPT-4 outperforms the English-language performance of existing language models [2, 3] for the vast majority of languages tested, including low-resource languages such as Latvian, Welsh, and Swahili.

MMLU (Massive Multitask Language Understanding) 인공지능 모델이 획득한 지식을 측정하는 벤치마크이다. 약 57개의 주제(STEM, the humanities, the social sciences 등)에 대해 다지선다 문제를 푸는 테스트

GPT4 - Limitations

- **No knowledge after Sept. 2019.**
- **Simply accept wrong information & white lying**
- **Cannot solve human's problems with high complexity**
  - Security issues in GPT4 generated source codes
- **Repeat mistakes**
  - Wrong doctor's prescription
  - Dangerous emergency treatment
  - Same homework report generation
- **Various output bias still exist → during RLHF, lose original information??**

ChatGPT vs. GPT4

Common

- Transformer Decoder
- RLHF learning

Difference

- Overall performance enhancement
- Longer input sequence (# tokens: 4K → 32K)
- Image input is possible
- Reducing hallucination

GPT-5 개요

- **2025 공개**
- **GPT4 대비 핵심 성능 개선**
  - 더 긴 컨텍스트 처리 (수십만~백만 토큰 수준 추정)
  - 추론(reasoning)능력 강화
  - 계산형 사고(Chain-of-Thought) 정확도 향상
  - 인간-수준(Human-Level) 의사결정에 더 근접
  - 안전성(Safety)·사실성(Factual accuracy)이 더욱 강화됨

GPT-5의 기술적 변화 (Technical Shift)

기존 Transformer + RLHF에서 확장된 방식

- 강화된 RLHF (multi-stage human alignment)
- Human + AI feedback 혼합 학습
- 모델이 자기 자신을 평가하고 수정하는 Self-Correction 루프

멀티모달 확장

- 텍스트 + 이미지 → 텍스트 + 이미지 + 오디오 + 비디오

실시간형(Real-Time) AI로의 이동

- GPT-5는 API 기반이 아니라
- "지능형 에이전트(autonomous AI agent)" 방향으로 진화

GPT 시대의 걱정...

Humanity

- 생산성 향상은 확실함
- 사람을 완벽히 대체 X, 상당부분 감소 가능 → 사람? 기계

이제는 빅 모델 시대...

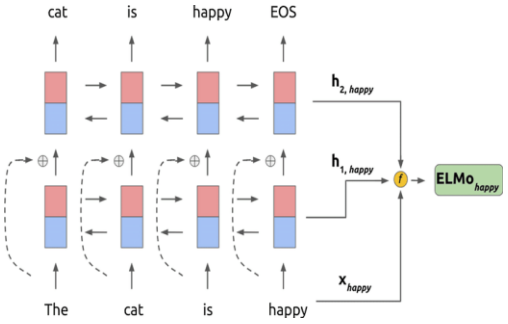
- 기술발전 속도는 더욱 빨라질 것: GPT5, 6, 7, ...
- 특정 Task에서 성능 향상하기 위한 노력이 의미가 있는가?
  - 모델을 키우고 데이터만 많다면 성능이 더 좋음
  - 많은 연구자들의 좌절 감정도 존재

Encoder-Decoder Models

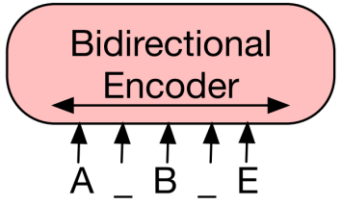
BART: Bi-directional & Auto-Regressive Transformer

BART: Transformer's encoder & decoder simultaneously

- **BART** Bi-directional & Auto-regressive Transformers
  - Paper link: https://arxiv.org/abs/1910.13461 [Lewis et al., 2019]
  - Github: https://github.com/facebookresearch/fairseq/blob/main/examples/bart/README.md

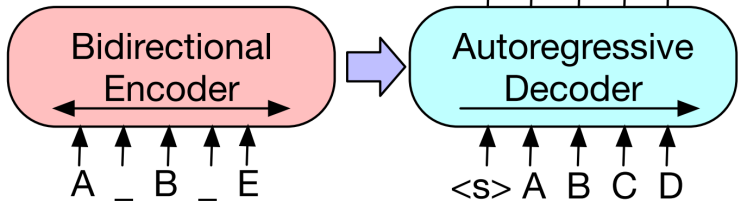


ELMO



BERT

Google

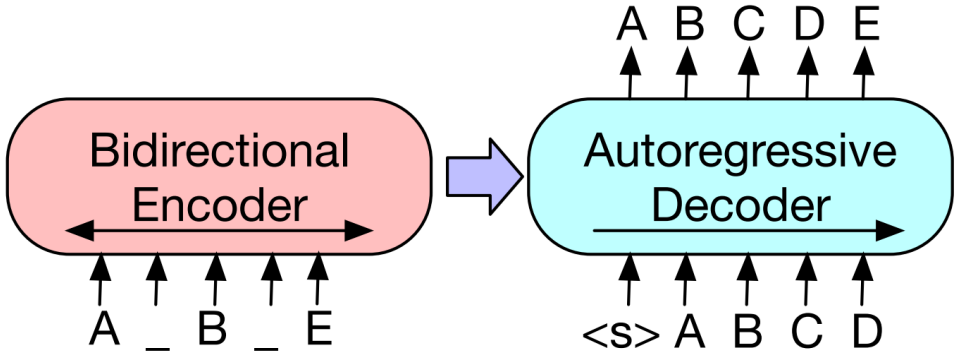


BART

facebook

BART - Pretraining

- **Main characteristics: Using Transformers Encoder & Decoder → Pre-training**
  - Input: encoder, output: decoder
  - Reconstruct original text
  - Applying autoregressive



- **BART (Autoregressive) feature**
  - Don't care the length of input & output
    - Can add various noise into encoder
    - Increase learning difficulty → maybe increase the accuracy of final prediction
    - Therefore, NLU performance is increased
  - Also, NLG is possible since BART has decoder (auto-regressive)

BART - Pretraining (Denoising)

How can make a model study harder?

- Adding noise into input → BART can denoise the noise → Reconstruct original text (denoising)

1. 토큰 마스킹 (Token Masking)

original	corrupted
Chelsea is my favorite football club	Chelsea is my favorite [MASK] club

2. 토큰 삭제 (Token Deletion)

original	corrupted
Chelsea is my favorite football club	Chelsea is my favorite club

3. 토큰 채우기 (Token Infilling, 일정길이 masking)

original	corrupted
I loved the book so much and I have read it so many times	I loved [MASK] and I have read it so many times

4. 문장 섞기 (Sentence Shuffling)

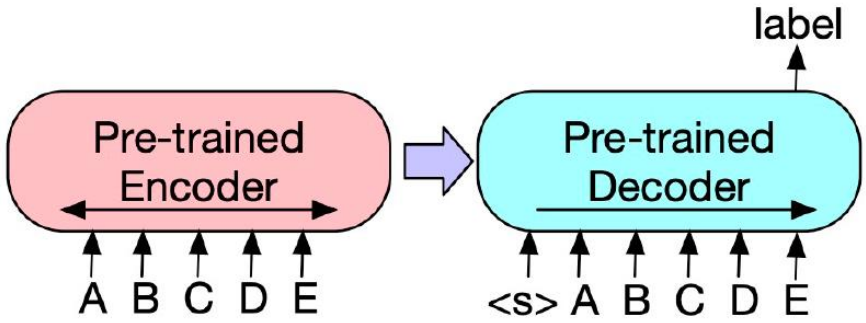
original	corrupted
I completed my assignment by evening. Then I started playing a game. I played the game until 10 PM. Then I went to sleep.	I played the game until 10 PM.Then I started playing a game.I completed my assignment by evening Then I went to sleep.

5. 문서 회전 (Document Rotations)

original	corrupted
I completed my assignment by evening. Then I started playing a game. I played the game until 10 PM. Then I went to sleep	Playing a game. I played the game until 10 PM. Then I went to sleep. I completed my assignment by evening Then I started

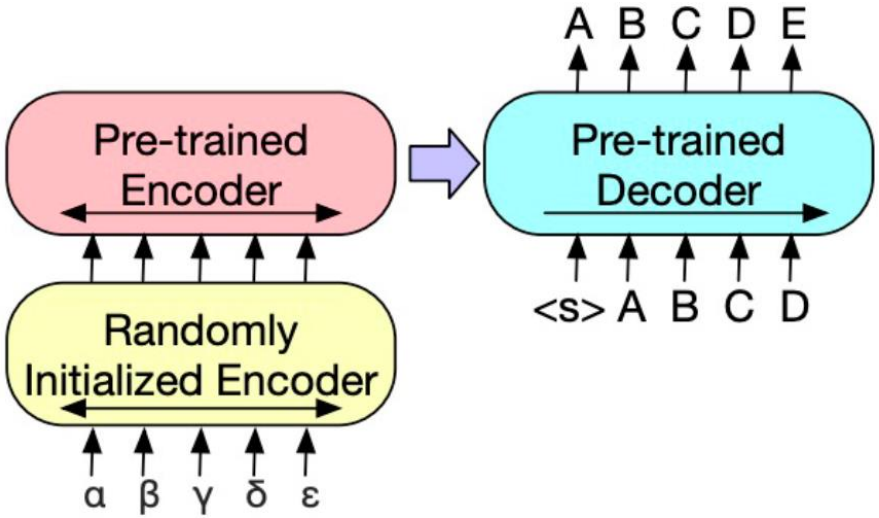
BART - Fine-tuning

■ **BART NLU:** (example) Text Classification



- The same input is fed into the encoder and decoder,
- The representation from the final output is used.

■ **BART NLG:** (example) Machine Translation



- Learn a small additional encoder that replaces the word embeddings in BART.
- The new encoder can use a disjoint vocabulary

BART - Performance

BART Performance

	Question Answering	Question Answering	Textual Entailment	Text Classification	Text Comparison	Question Answering	Text Comparison	Textual Entailment	Text Classification	Linguistic Acceptability
	SQuAD 1.1 EM/F1	**SQuAD 2.0** EM/F1	**MNLI** m/mm	**SST** Acc	**QQP** Acc	**QNLI** Acc	**STS-B** Acc	**RTE** Acc	**MRPC** Acc	**CoLA** Mcc
BERT	84.1/90.9	79.0/81.8	86.6/-	93.2	91.3	92.3	90.0	70.4	88.0	60.6
UniLM	-/-	80.5/83.4	87.0/85.9	94.5	-	92.7	-	70.9	-	61.1
XLNet	**89.0**/94.5	86.1/88.8	89.8/-	95.6	91.8	93.9	91.8	83.8	89.2	63.6
RoBERTa	88.9/**94.6**	**86.5/89.4**	**90.2/90.2**	96.4	92.2	94.7	**92.4**	86.6	**90.9**	**68.0**
BART	88.8/**94.6**	86.1/89.2	89.9/90.1	**96.6**	**92.5**	**94.9**	91.2	**87.0**	90.4	62.8

Generation Task (대화 응답)

	ConvAI2	
	Valid F1	Valid PPL
Seq2Seq + Attention	16.02	35.07
Best System	19.09	17.51
BART	**20.72**	**11.85**

Table 4: BART outperforms previous work on conversational response generation. Perplexities are renormalized based on official tokenizer for ConvAI2.

Generation Task (요약 응답)

	ELI5		
	R1	R2	RL
Best Extractive	23.5	3.1	17.5
Language Model	27.8	4.7	23.1
Seq2Seq	28.3	5.1	22.8
Seq2Seq Multitask	28.9	5.4	23.1
BART	**30.6**	**6.2**	**24.3**

Table 5: BART achieves state-of-the-art results on the challenging ELI5 abstractive question answering dataset. Comparison models are from Fan et al. (2019).

Encoder - Decoder Models
T5

T5 - 작동 예제

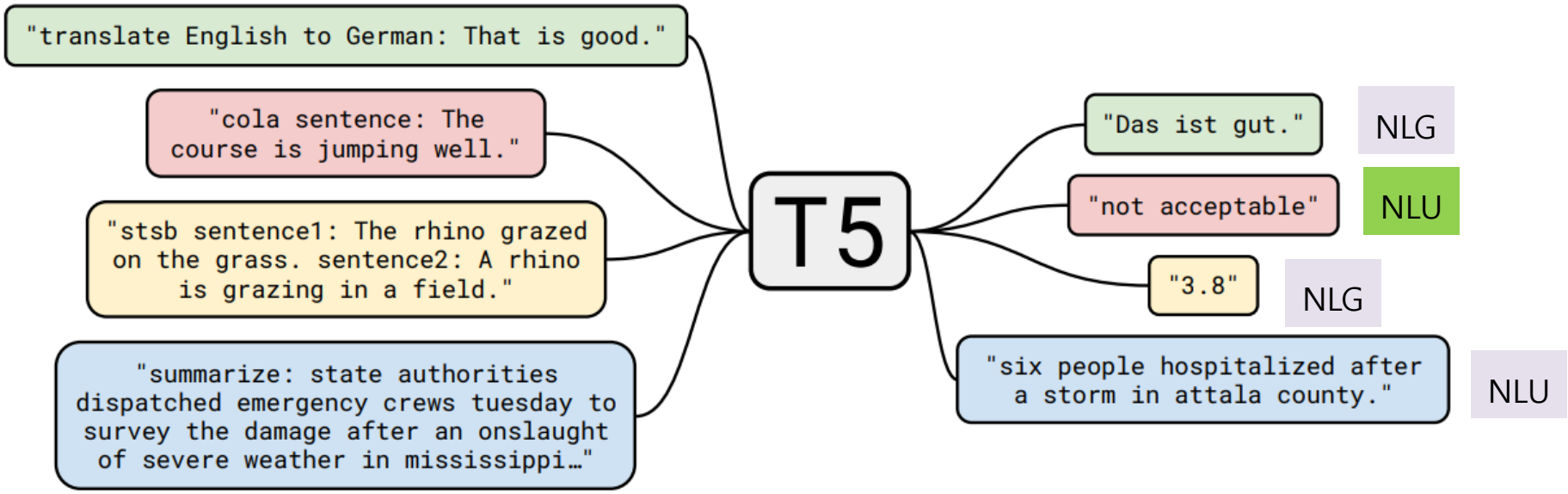
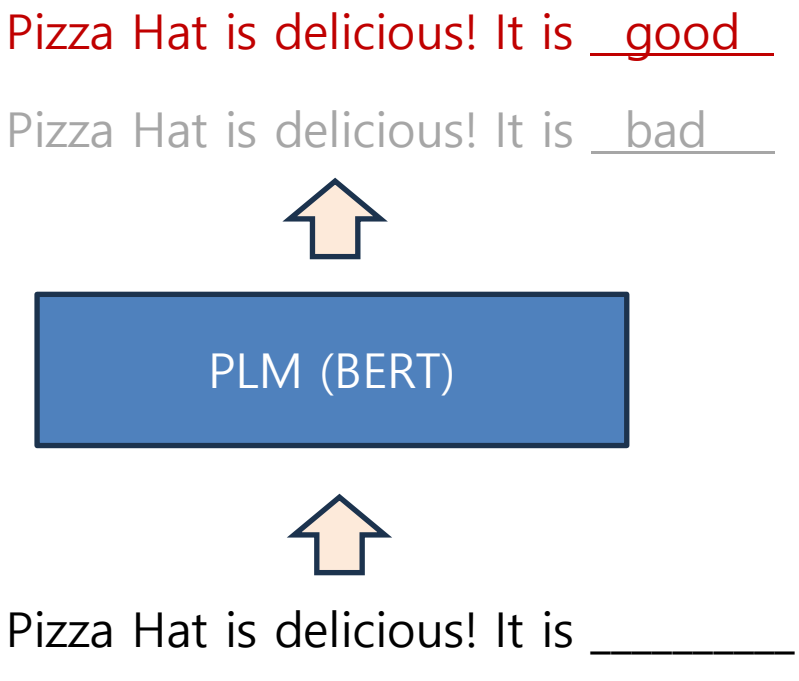


Figure 1: A diagram of our text-to-text framework. Every task we consider—including translation, question answering, and classification—is cast as feeding our model text as input and training it to generate some target text. This allows us to use the same model, loss function, hyperparameters, etc. across our diverse set of tasks. It also provides a standard testbed for the methods included in our empirical survey. "T5" refers to our model, which we dub the "**T**ext-**t**o-**T**ext **T**ransfer **T**ransformer".

T5 - even NLU possible?

- **T5: Without fine-tuning → able to apply NLU**



Training to learn right answer!

Pros

Without additional Layer,

Tine-tuning is converted to NLG

Cons

Depending sentences,

it is important to make prompt

(Issue: how to make good prompt?)

수고하셨습니다 ..^^..