05

LLM(Large Language Model)

오늘 학습을 통해 우리는

- LLM의 등장 배경에 대해 살펴봅니다.
- BERT 이후 주요 언어 모델들을 살펴보고, 모델들의 특징을 통해 다양한 학습법을 익힙니다.
- 거대 언어모델의 한계점을 파악하고, 개선 방향을 탐색합니다.

목차

01

거대 언어모델의 등장 배경

자연어 처리의 발전 과정

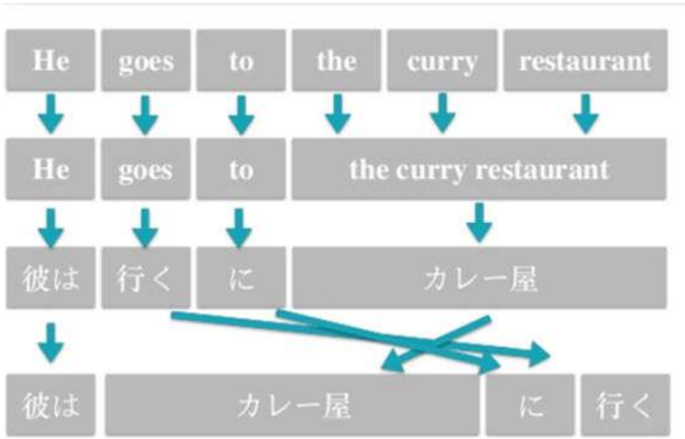
1. 규칙 기반 시스템 (Rule-based Systems)

- 초기 자연어 처리 방법
- 사람이 직접 규칙을 정의
- Ex)문법 규칙, 어휘 사전
- 한계: 언어의 다양성과 예외사항 처리 어려움

자연어 처리의 발전 과정

2. 통계 기반 방법 (Statistical Methods)

- 대량의 텍스트 데이터 활용
- 단어 빈도, 동시 출현 빈도 등의 통계적 정보 활용
- 예: n-gram 모델, 통계적 기계 번역

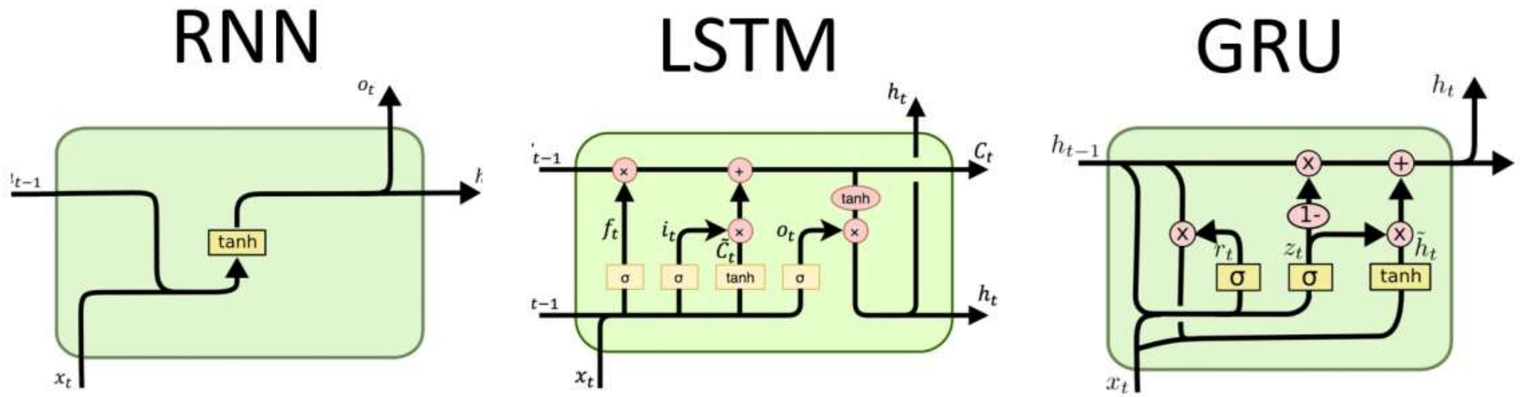


출처 : https://velog.io/@sjinu/Language-Modeling-N-gram, https://kantanmtblog.com/2019/04/02/a-short-introduction-to-the-statistical-machine-translation-model/

자연어 처리의 발전 과정

3. 딥러닝의 등장

- 인공 신경망을 활용한 자연어 처리
- Ex) Word2Vec, RNN, LSTM
- 문맥을 더 잘 이해하고 다양한 패턴 학습 가능

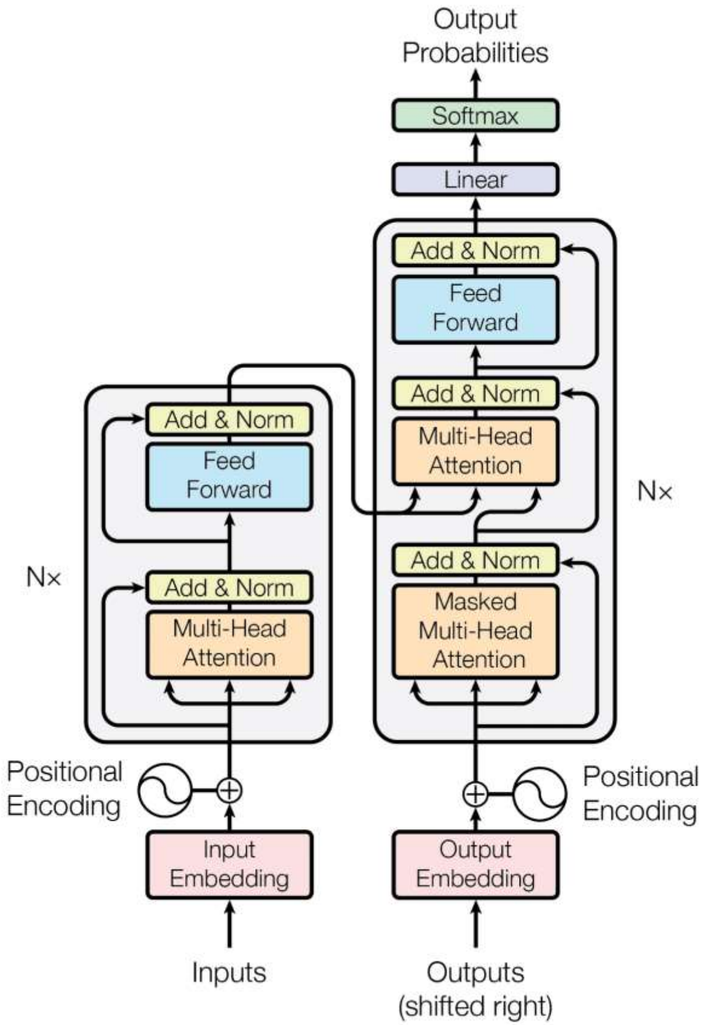


출처 : http://dprogrammer.org/rnn-lstm-gru

자연어 처리의 발전 과정

4. Transformer 아키텍처의 등장

- Attention 메커니즘 도입
- Ex) BERT, GPT, T5
- 병렬 처리 가능, 대규모 데이터 학습 효율화

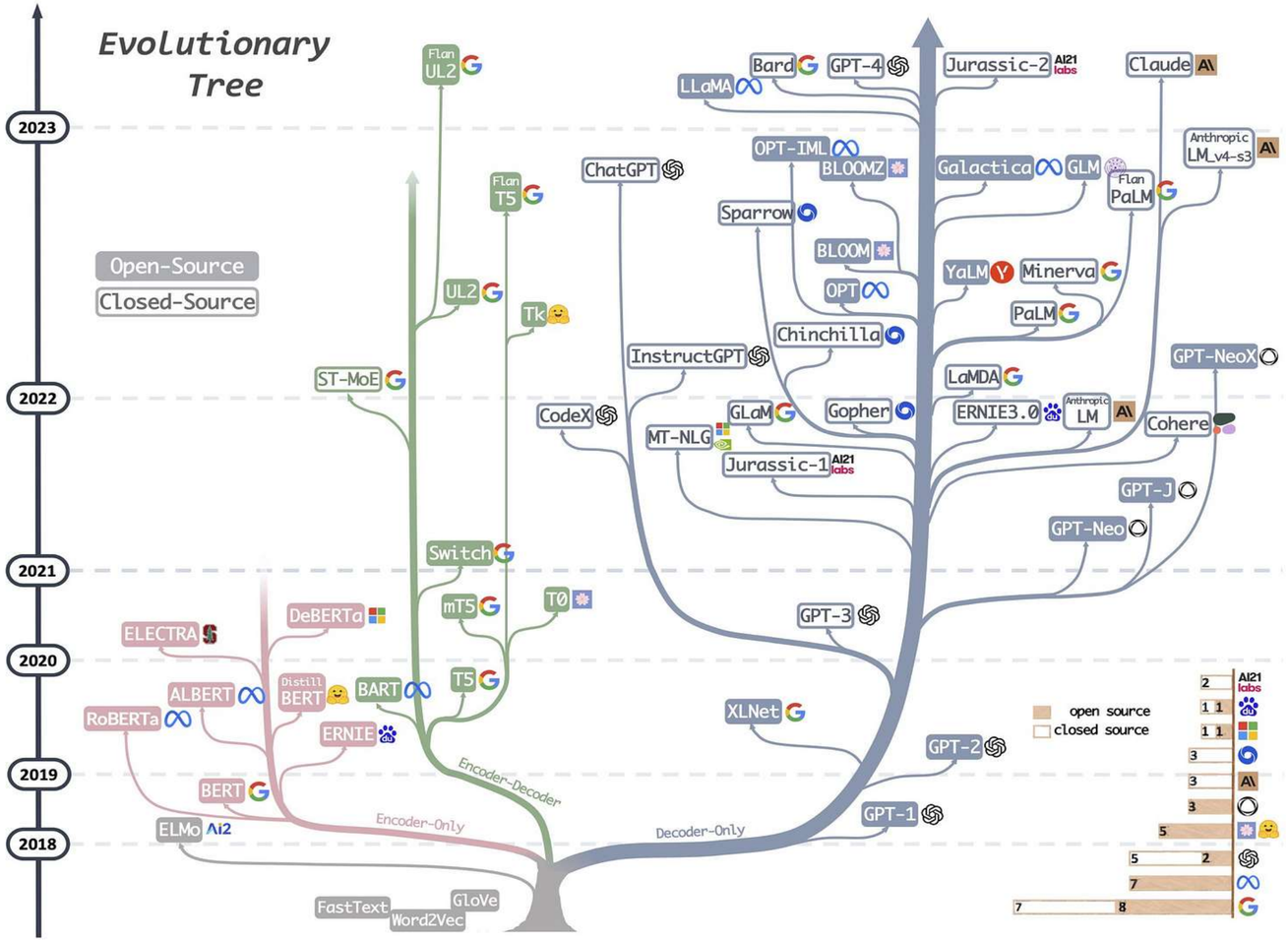


출처 : https://arxiv.org/abs/1706.03762

자연어 처리의 발전 과정

5. 거대 언어모델의 탄생

- 대규모 데이터와 컴퓨팅 파워의 결합
- Ex) GPT-4, LLaMA
- 일반적인 태스크에서 높은 성능, 다양한 응용 가능



출처 : https://github.com/Mooler0410/LLMsPracticalGuide

BERT의 혁신과 그 이후의 변화

- BERT (Bidirectional Encoder Representations from Transformers)의 혁신
  - 양방향 Transformer 인코더 사용
    - 문장의 모든 단어를 동시에 고려하여 문맥 이해
  - Pretraining을 통한 대규모 데이터 학습
    - Masked Language Model(MLM) 방식을 통한 단어와 문맥 학습
      - 일부 단어를 가리고 (masking) 나머지 단어로부터 해당 단어를 예측
    - Next Sentence Prediction(NSP) 학습을 통한 문장 간 연관성 학습
      - 두 문장을 붙여 개연성에 대한 이진 분류

BERT의 혁신과 그 이후의 변화

- BERT (Bidirectional Encoder Representations from Transformers)의 혁신
  - 전이 학습 (Transfer Learning)의 활용
    - 대규모 데이터셋에서 사전 학습 후, 특정 태스크에 미세 조정 (fine-tuning)
  - 다양한 NLP 태스크에서의 높은 성능
    - 단일 모델 아키텍처로 다양한 태스크 성능 향상

BERT의 혁신과 그 이후의 변화

- 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)

출처 : https://humboldt-wi.github.io/blog/research/information_systems_1920/uncertainty_identification_transformers/

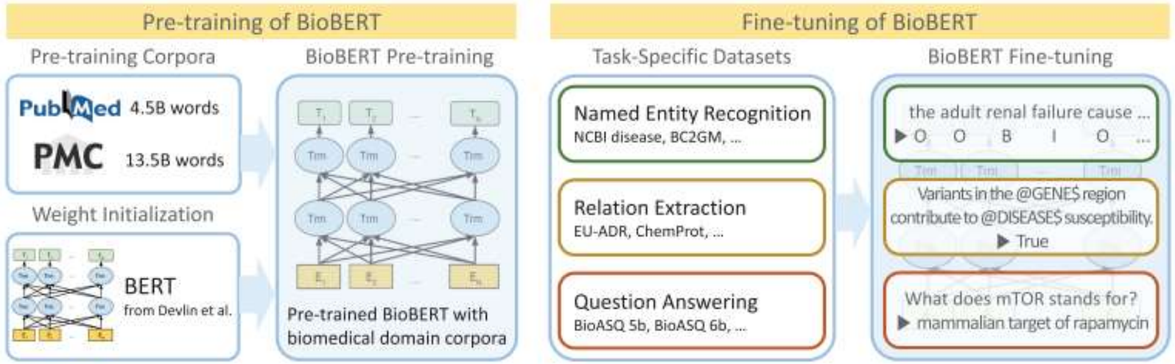
BERT의 혁신과 그 이후의 변화

- 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(1억2000만 문장 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]



출처 : https://sktelecom.github.io/project/kobert/, https://arxiv.org/abs/1901.08746

BERT의 혁신과 그 이후의 변화

- Zero-shot, Few-shot 학습의 활용
  - 제한된 데이터로도 높은 성능 달성 가능
- 거대 언어모델의 등장
  - BERT의 성공을 기반으로 더 큰 모델들의 연구 및 개발 (예: GPT-4, LLaMA2, ALPACA)

02

BERT 이후의 주요 언어모델들

Transformer-XL (Transformer eXtra Long): 장기 의존성 학습의 개선

- 장기 의존성(Long term dependency)문제
  - 전통적인 Transformer 모델은 고정된 길이의 문맥만을 고려
  - 긴 문서나 시퀀스에서 이전 정보를 잘 활용하지 못함

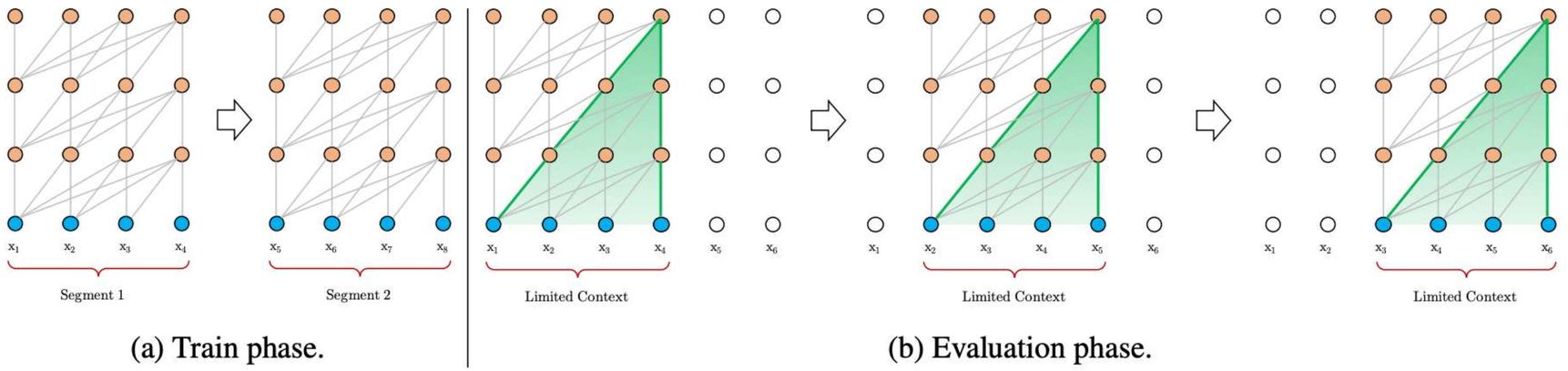


Figure 1: Illustration of the vanilla model with a segment length 4.

출처 : https://arxiv.org/pdf/1901.02860.pdf

Transformer-XL의 핵심 아이디어 - Segment-level Recurrence

- 기본 Transformer 모델은 고정된 길이의 입력을 처리

- 그러나 Transformer-XL은 이전 시퀀스의 정보(State)를 현재 시퀀스에 전달하는 순환 메커니즘을 도입하여 문제를 해결
  - 모델은 더 긴 시퀀스의 정보를 활용할 수 있게 되어, 장기적인 의존성을 더 잘 학습
  - 또한, State를 재사용하므로 효율적이고 빠르게 학습 가능

Transformer-XL의 핵심 아이디어 - Segment-level Recurrence

- 학습이 진행되는 동안, 각 Segment의 연산 결과들을 다음 Segment가 이용할 수 있도록 저장(fixed/cached)

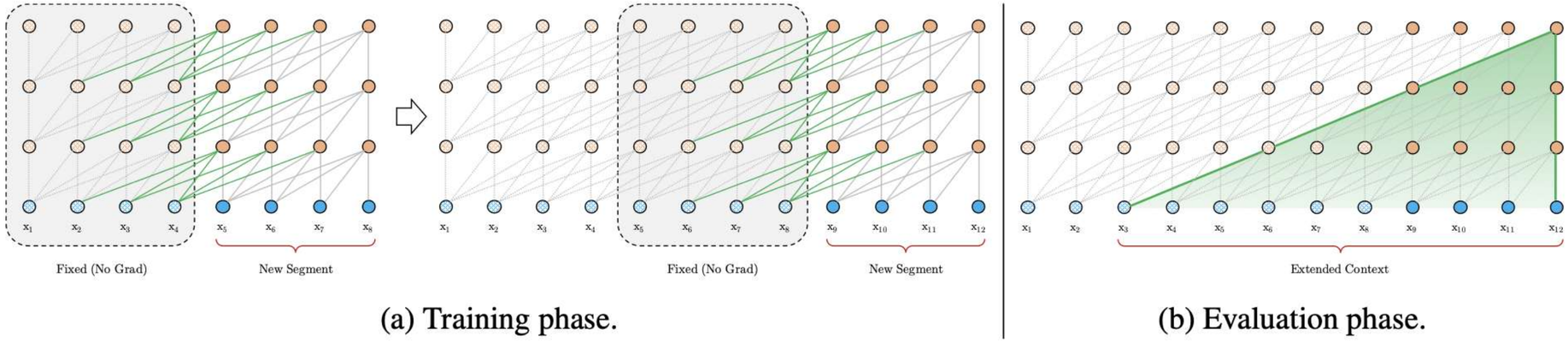


Figure 2: Illustration of the Transformer-XL model with a segment length 4.

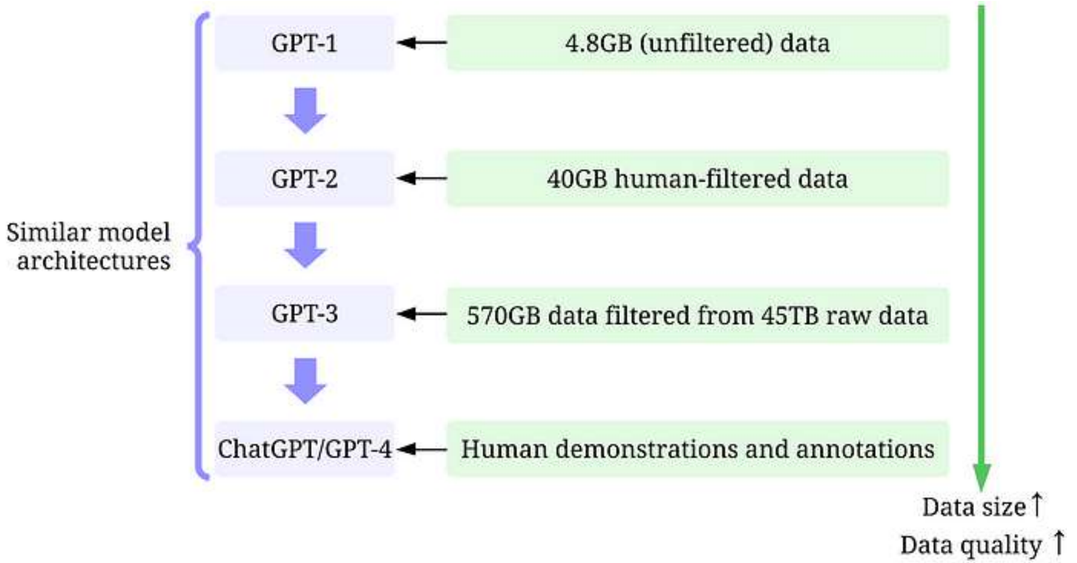
출처 : https://arxiv.org/pdf/1901.02860.pdf

Transformer-XL의 장점

- 장기 의존성 학습 개선
  - 긴 문맥 정보를 더 잘 활용하여 성능 향상
- 더 빠른 학습 속도
  - 이전 State의 재활용으로 효율적인 학습 가능
- 다양한 태스크에서의 높은 성능
  - 언어 모델링, 기계 번역 등에서 기존 모델보다 높은 성능

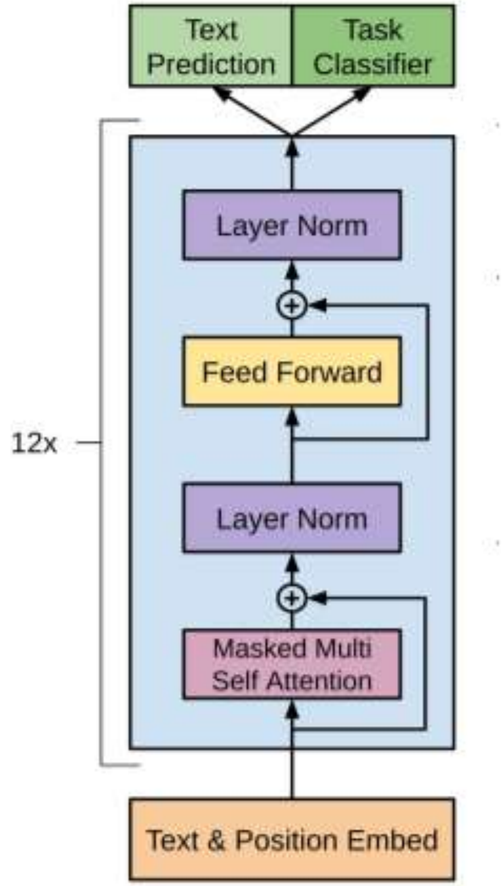
GPT: Generative Pre-trained Transformer

- 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



GPT-1: Generative Pre-trained Transformer 1

- 준지도학습을 통한 모델 훈련
  - 비지도학습을 통한 Pretraining
  - 지도학습을 통한 Fine-tuning
- 비지도학습은 Language Modelling을 통해 진행
  - MLM, NSP 등과 비슷한 방식

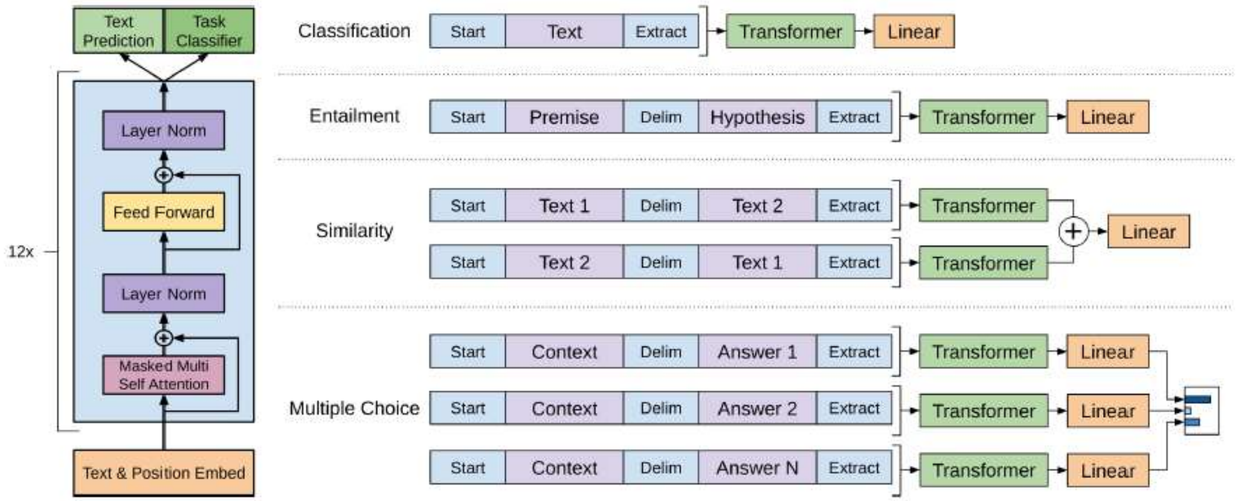


출처 : https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf

GPT-1: Generative Pre-trained Transformer 1

- 지도학습은
  - Task-specific input transformations
  - 최소한으로 모델 구조를 변형하여

    지도학습 수행
    - 입출력 구조 변형
    - Classification, Entailment
    - 혹은 데이터의 토큰을 추가
    - Similarity, Multiple choice



출처 : https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf

GPT-2: Generative Pre-trained Transformer 2

- 학습 방식
  - 준지도학습: 대규모 텍스트 데이터를 사용하여 언어 모델링
- 특징
  - 구조와 학습 방식이 GPT-1과 동일
  - 다양한 NLP 태스크에서 Zero-shot, Few-shot 학습 가능
  - 고도의 문장 생성 능력
- 컨트로버시
  - 초기에는 모델의 크기와 생성 능력 때문에 공개를 주저함

GPT-3: Generative Pre-trained Transformer 3

- 모델 크기
  - 1750억 개의 파라미터를 가진 거대한 모델
- 학습 방식
  - GPT-2와 유사한 준지도학습, 하지만 더 큰 데이터와 모델로 학습
- 특징
  - 매우 다양한 태스크에서 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

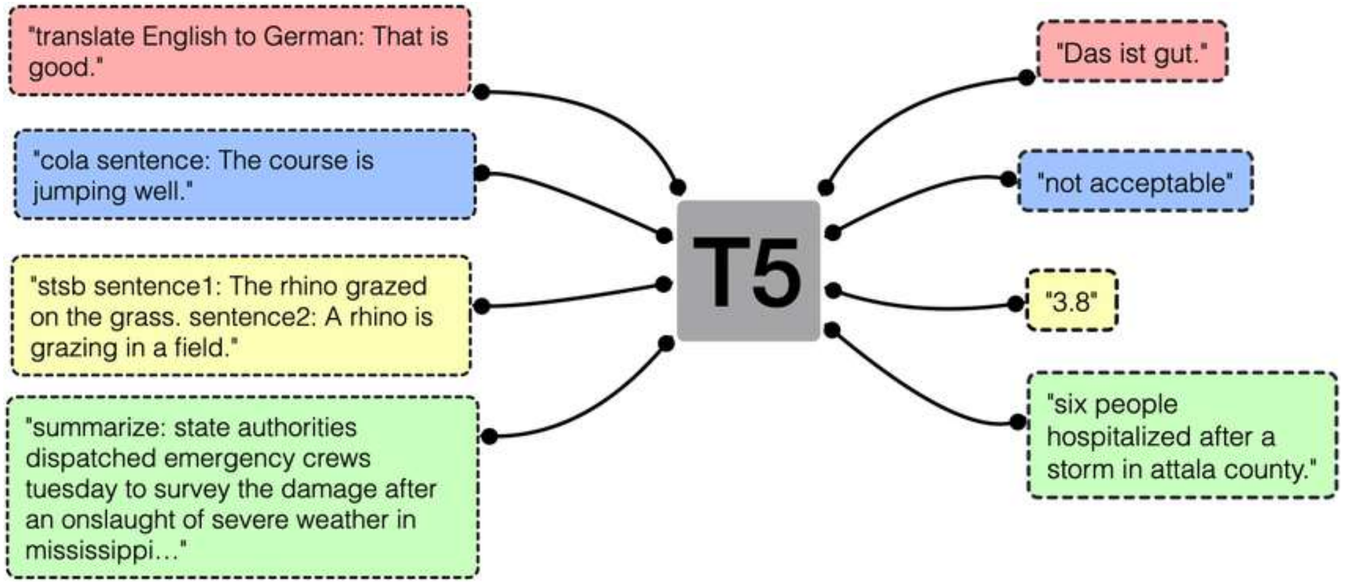
GPT-3: Generative Pre-trained Transformer 3

- 모델 크기, 학습 방식 등 비공개
  - NVIDIA H100 GPU 10,000여 대 이상 사용
  - Fine-tuning 기간이 6개월에 이를 것으로 추측
- 특징
  - Reinforcement Learning from Human Feedback (RLHF)
    - 모델 출력에 인간이 개입하면 강화학습 방식을 통해 파라미터가 업데이트됨
  - 멀티모달: 이미지, 음성 등 다른 데이터도 다룰 수 있음
  - API 및 상용화
    - GPT-4 기반의 API가 제공

T5 (Text-to-Text Transfer Transformer): 텍스트를 통한 모든 것

1. 기본 아이디어

- 모든 NLP 태스크를 "텍스트를 입력받아 텍스트를 출력하는" 문제로 변환
- 예: "번역: Hello, World!" → "안녕, 세상!"

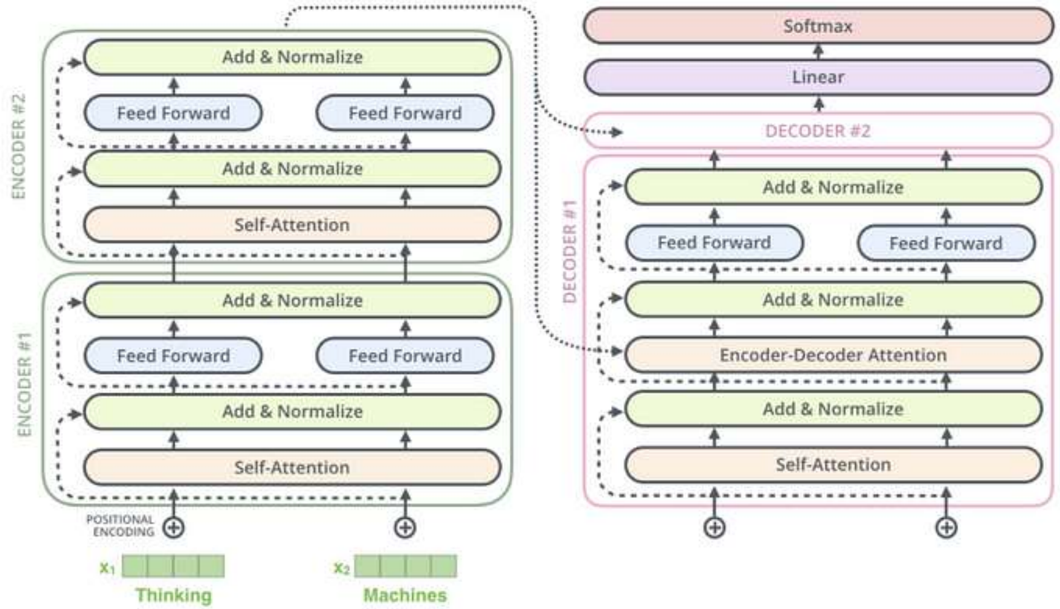


출처 : https://www.researchgate.net/figure/T5-as-a-unified-framework-for-down-stream-NLP-tasks-The-diagram-shows-each-down-stream_fig3_347534633

T5 (Text-to-Text Transfer Transformer): 텍스트를 통한 모든 것

2. 모델 구조

- Transformer 기반의 인코더-디코더 구조
- BERT나 GPT와는 달리, 인코더와 디코더 모두 사용

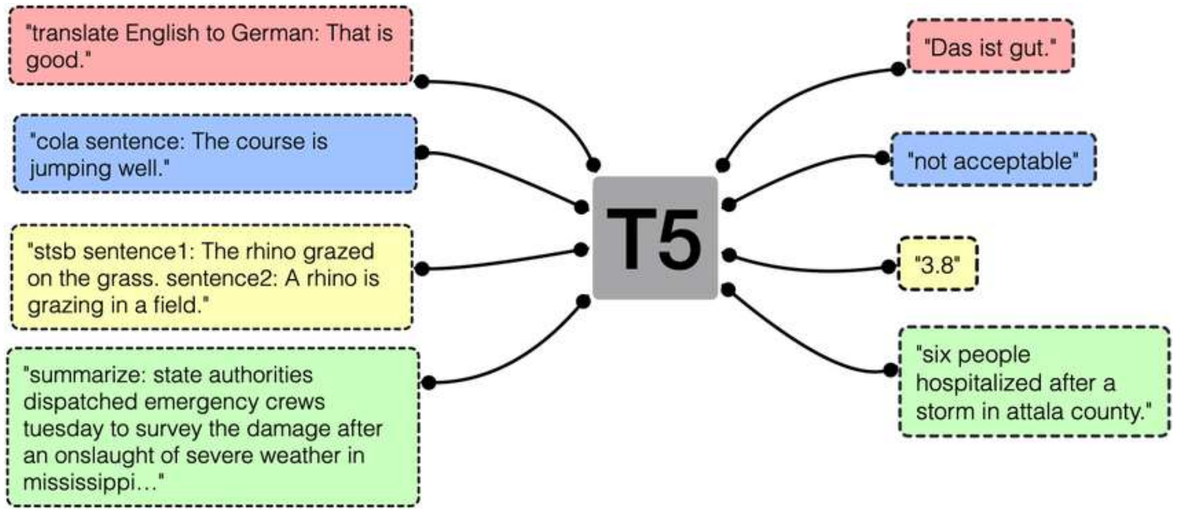


출처 : http://jalammar.github.io/illustrated-transformer/

T5 (Text-to-Text Transfer Transformer): 텍스트를 통한 모든 것

3. 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
    - translate, cola, stsb 등



출처 : https://www.researchgate.net/figure/T5-as-a-unified-framework-for-down-stream-NLP-tasks-The-diagram-shows-each-down-stream_fig3_347534633

T5 (Text-to-Text Transfer Transformer): 텍스트를 통한 모든 것

4. 태스크 독립성

- 동일한 모델 아키텍처와 학습 방식으로 다양한 NLP 태스크 수행
- 태스크 특화 파라미터가 없음

5. 성능

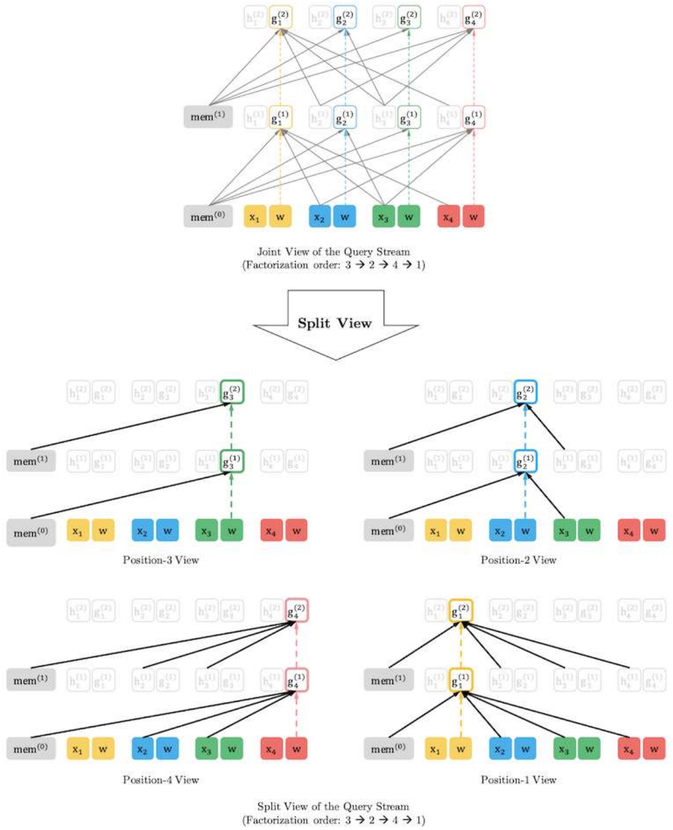
- 다양한 벤치마크 태스크에서 최고 성능 달성
- GLUE, SuperGLUE, SQuAD 등의 태스크에서 높은 성능

XLNet: 순열 기반 학습과 Transformer-XL의 결합

1. 기본 아이디어

- BERT의 Masked Language Model (MLM) 방식의 한계
  - MLM은 실제 Downstream 태스크에서 사용되는 방식과 일치하지 않음

- 이를 극복하기 위해 순열 기반의 언어 모델링(PLM, Permutation Language Modeling) 방식 도입



출처 : https://arxiv.org/abs/1906.08237

XLNet: 순열 기반 학습과 Transformer-XL의 결합

2. Permutation Language Modeling의 핵심 아이디어:

- 단어 간 순열 고려:
  - 주어진 문장의 토큰들에 대해 가능한 모든 순열(permutations)을 고려
  - 문장 내 n 개의 단어가 있다면 총 n! 가지의 순열이 생성
  - 예를 들어, "A B C"라는 문장이 있다면, "A B C", "A C B", "B A C", "B C A", "C A B", "C B A"와 같은 모든 순열이 생성됨

XLNet: 순열 기반 학습과 Transformer-XL의 결합

2. Permutation Language Modeling의 핵심 아이디어:

- 자기 회귀적 예측:
  - 각 순열에 대해, 토큰을 하나씩 차례대로 예측
  - 이때, 이전에 예측한 토큰들의 정보를 활용하여 다음 토큰을 예측함
  - 예를 들어, "A C B" 순열에서 "A"를 먼저 예측하고, 그 다음 "C"를 예측할 때 "A"의 정보를 활용하며, 마지막으로 "B"를 예측할 때 "A"와 "C"의 정보를 활용
  - 이러한 방식을 앞선 모든 순열에 대해 적용

XLNet: 순열 기반 학습과 Transformer-XL의 결합

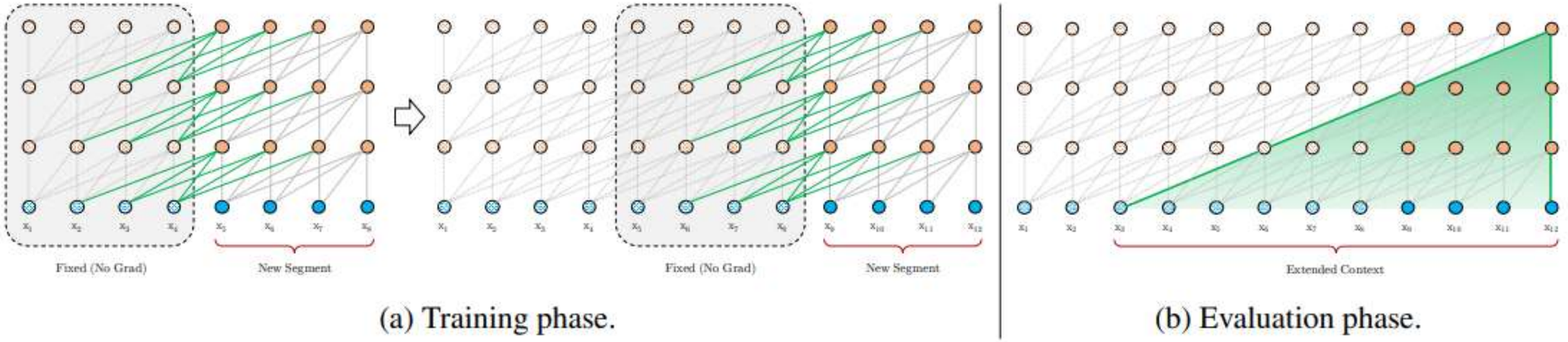
2. Permutation Language Modeling의 핵심 아이디어:

- 마스킹 없음:
  - PLM 방식은 BERT의 MLM과는 달리 토큰을 마스킹하지 않음
    - 즉 일부 데이터만 활용하는 것이 아니고, 모든 데이터를 학습에 활용
  - 대신, 각 순열에서의 토큰의 순서를 바꿔가며 모델을 학습시킴

XLNet: 순열 기반 학습과 Transformer-XL의 결합

3. Transformer-XL의 결합

- Transformer-XL의 장기 의존성 학습 능력을 활용
- Segment-level Recurrence와 Relative Positional Encoding 기법 적용



출처 : https://arxiv.org/abs/1901.02860

XLNet: 순열 기반 학습과 Transformer-XL의 결합

4. 성능

- 다양한 벤치마크 태스크에서 높은 성능을 보임
- 특히, 긴 문맥 정보가 필요한 태스크에서 뛰어난 성능

SQuAD2.0	EM	F1	SQuAD1.1	EM	F1
Dev set results (single model)					
BERT [10]	78.98	81.77	BERT† [10]	84.1	90.9
RoBERTa [21]	86.5	89.4	RoBERTa [21]	88.9	94.6
XLNet	**87.9**	**90.6**	XLNet	**89.7**	**95.1**
Test set results on leaderboard (single model, as of Dec 14, 2019)					
BERT [10]	80.005	83.061	BERT [10]	85.083	91.835
RoBERTa [21]	86.820	89.795	BERT* [10]	87.433	93.294
XLNet	**87.926**	**90.689**	XLNet	**89.898‡**	**95.080‡**

ELECTRA: 효율적인 학습을 위한 새로운 접근

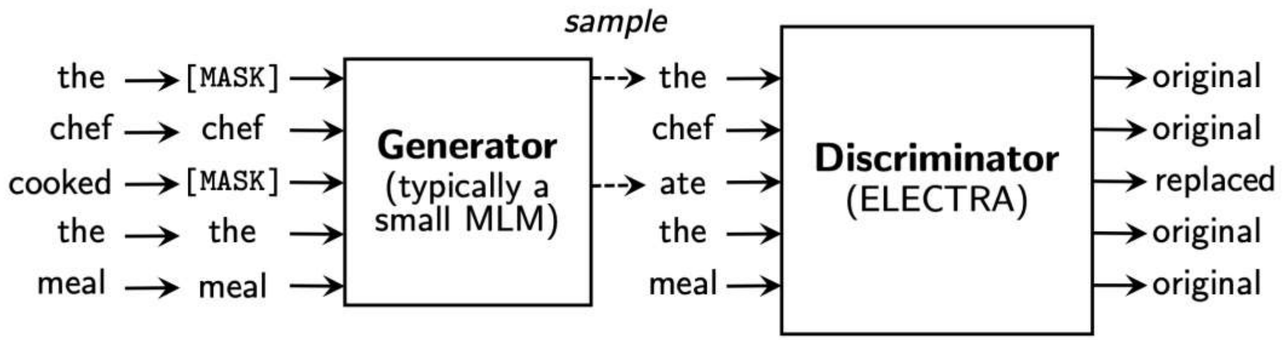
- ELECTRA(Efficiently Learning an Encoder that Classifies Token Replacements Accurately)

1. 기본 아이디어
   - BERT의 Masked Language Model (MLM) 방식의 한계
     - 데이터가 입력되면, 전체 토큰 중 15%에 대해서만 loss가 발생
     - 즉 85%의 토큰은 비용 측면에서 낭비
   - 극복하기 위해 RTD(Replaced Token Detection) 기법 도입

ELECTRA: 효율적인 학습을 위한 새로운 접근

2. RTD(Replaced Token Detection)

- 생성기(Generator)와 판별기(Discriminator) 사용
  - 생성기는 입력 텍스트의 일부 토큰을 다른 토큰으로 대체
  - 판별기는 입력 텍스트의 각 토큰이 원래 텍스트의 토큰인지, 아니면 생성기에 의해 대체된 토큰인지를 판별
  - 전반적인 흐름은 GAN(Generative Adversarial Network)와 유사

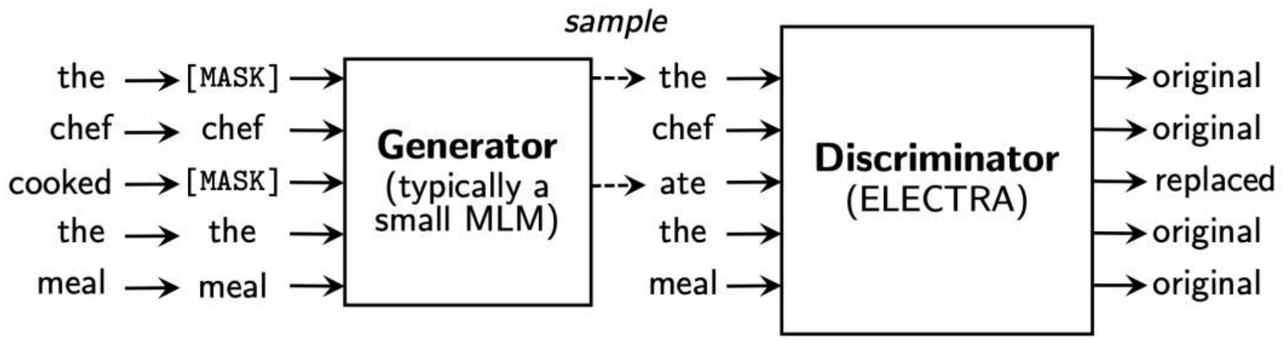


출처 : https://openreview.net/attachment?id=r1xMH1BtvB&name=original_pdf

ELECTRA: 효율적인 학습을 위한 새로운 접근

2. RTD(Replaced Token Detection)

- 대체된 토큰의 활용
  - MLM 방식에서는 일부(15%) 토큰을 마스킹하고 그 마스킹된 토큰을 예측하는 방식을 사용
  - 반면, RTD 방식에서는 생성기가 대체한 토큰을 판별기가 식별하는 방식을 사용합니다
  - 모델은 잘못된 예측에 대한 정보도 학습하게 되어, 학습 효율이 향상됩니다.

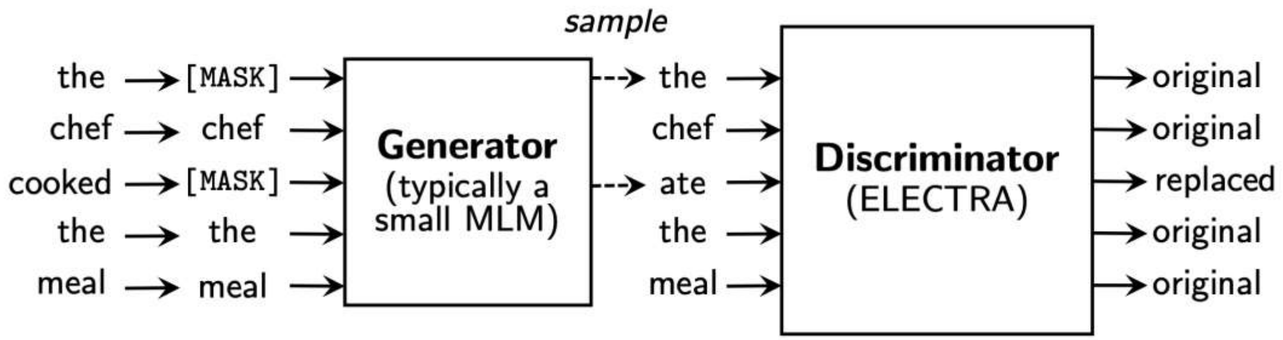


출처 : https://openreview.net/attachment?id=r1xMH1BtvB&name=original_pdf

ELECTRA: 효율적인 학습을 위한 새로운 접근

2. RTD(Replaced Token Detection)

- 효율적인 학습(Efficiently Learning)
  - RTD 방식은 전체 토큰에 대해 학습을 진행하기 때문에, MLM (Masked Language Modeling) 방식에 비해 더 많은 토큰을 학습에 활용할 수 있음
  - 동일한 계산 리소스를 사용하면서도 더 높은 성능을 달성



출처 : https://openreview.net/attachment?id=r1xMH1BtvB&name=original_pdf

ELECTRA: 효율적인 학습을 위한 새로운 접근

3. 성능

- 다양한 벤치마크 태스크에서 BERT와 유사한 또는 더 나은 성능을 보임
- 특히, 작은 모델 크기나 데이터 제한 상황에서 높은 효율성

Model	Train FLOPs	Params	CoLA	SST	MRPC	STS	QQP	MNLI	QNLI	RTE	Avg.
BERT	1.9e20 (0.27x)	335M	60.6	93.2	88.0	90.0	91.3	86.6	92.3	70.4	84.0
XLNet	9.6e20 (1.3x)	360M	63.6	95.6	89.2	91.8	91.8	89.8	93.9	83.8	87.4
RoBERTa-100K	6.4e20 (0.90x)	356M	66.1	95.6	**91.4**	92.2	92.0	89.3	94.0	82.7	87.9
RoBERTa	3.2e21 (4.5x)	356M	68.0	**96.4**	90.9	**92.4**	92.2	90.2	**94.7**	86.6	88.9
BERT (ours)	7.1e20 (1x)	335M	67.0	95.9	89.1	91.2	91.5	89.6	93.5	79.5	87.2
ELECTRA	7.1e20 (1x)	335M	**69.3**	96.0	90.6	92.1	**92.4**	**90.5**	94.5	**86.8**	**89.0**
Test set results for models with standard single-task finetuning (no ensembling, task-specific tricks, etc.)											
BERT	1.9e20 (0.27x)	335M	60.5	94.9	89.3	86.5	89.3	86.7	92.7	70.1	83.8
SpanBERT	7.1e20 (1x)	335M	64.3	94.8	**90.9**	89.9	89.5	87.7	94.3	79.0	86.3
ELECTRA	7.1e20 (1x)	335M	**68.2**	**96.9**	89.6	**91.0**	**90.1**	**90.1**	**95.4**	**83.6**	**88.1**

출처 : https://openreview.net/attachment?id=r1xMH1BtvB&name=original_pdf

03

거대 언어모델 학습 방식

전이 학습 (Transfer Learning)

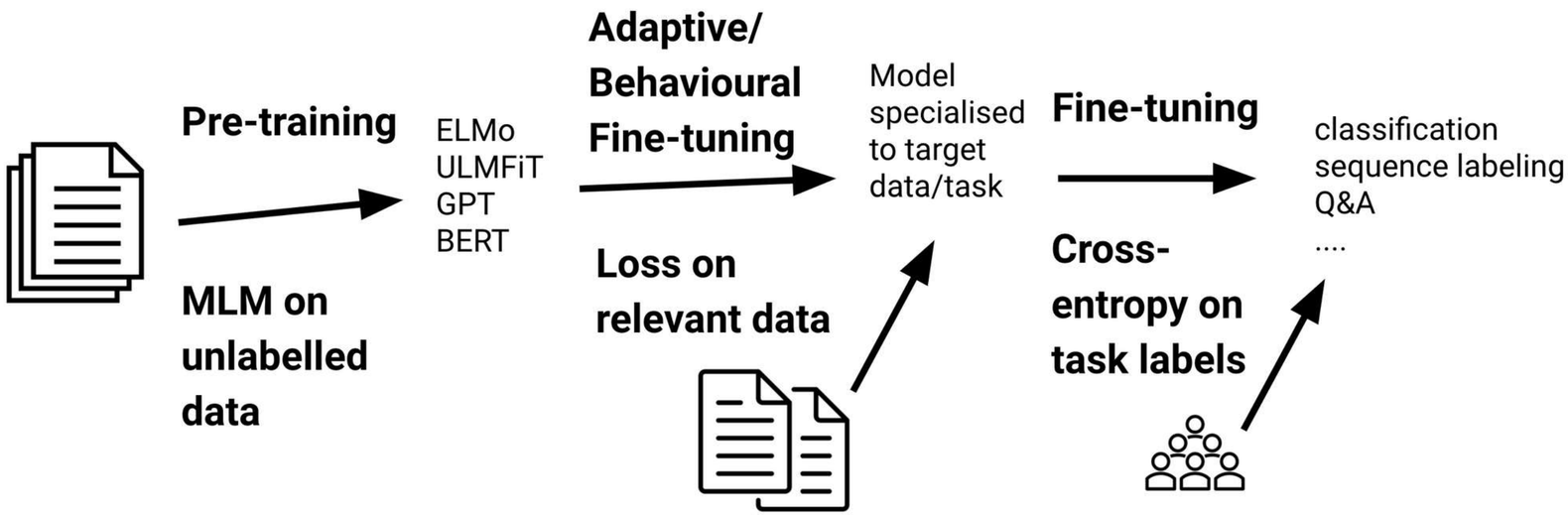
- 전이 학습의 정의
  - 이미 대규모 데이터셋에서 학습된 모델을 특정 태스크에 맞게 미세 조정하여 사용하는 방법
  - 대부분의 거대 언어모델은 대규모의 일반 텍스트 데이터에서 사전 학습
    - 갈 수록 비지도학습의 방식을 채택
    - 언어모델 뿐만 아니라 CV 분야에서도 자주 사용
  - 이후, 특정 태스크에 대해 미세 조정(Fine-tuning)을 통해 성능 향상
    - 이 과정은 지도학습을 사용하는 경우가 많음

전이 학습 (Transfer Learning)

- 전이 학습의 장점
  - 데이터 효율성: 작은 양의 태스크 특화 데이터로도 높은 성능 달성 가능
  - 학습 속도: 사전 학습된 가중치를 활용하므로 빠른 학습 가능
  - 일반화 능력: 다양한 데이터에서 학습된 지식을 활용하여 새로운 태스크에 빠르게 적응
  - 환경 측면: 초대형 언어모델의 학습에 소요되는 시간과 전력을 고려

미세 조정(Fine-tuning) 방법론

- 미세 조정의 정의
  - 사전에 학습된 모델의 가중치를 초기값으로 사용하고, 특정 태스크의 데이터로 추가 학습을 수행하는 방법



출처 : https://www.ruder.io/recent-advances-lm-fine-tuning/

미세 조정(Fine-tuning) 방법론

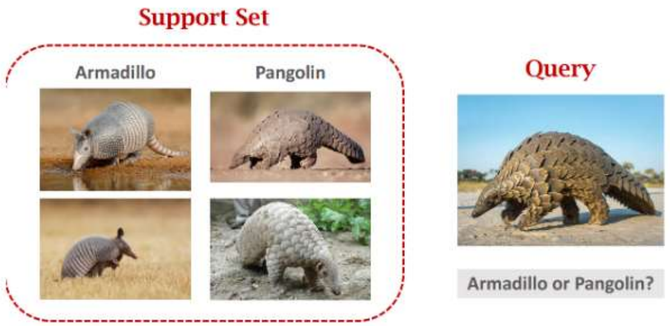
- 기본 절차
  - 모델 초기화: 사전 학습된 모델의 가중치를 불러옴
  - 데이터 준비: 특정 태스크에 대한 학습 및 검증 데이터 준비
  - 학습 설정: 학습률, 배치 크기, 에폭 수 등의 하이퍼파라미터 설정
  - 학습 수행: 태스크 데이터로 모델을 미세 조정
  - 평가 및 테스트: 검증 데이터와 테스트 데이터로 모델 성능 평가

미세 조정(Fine-tuning) 방법론

- 주의사항
  - 학습률: 사전 학습된 가중치를 고려하여, 너무 큰 학습률은 피해야 함
  - 과적합
    - 미세 조정 시 사용되는 데이터가 적을 경우, 과적합에 주의해야 함
    - 이를 위해 조기 종료, 드롭아웃 등의 기법 활용
  - 데이터 불균형: 태스크 데이터의 클래스 불균형이 있을 경우 데이터 증강, 적절한 샘플링이나 손실 함수 조절 필요

Zero-shot, Few-shot, Many-shot 학습

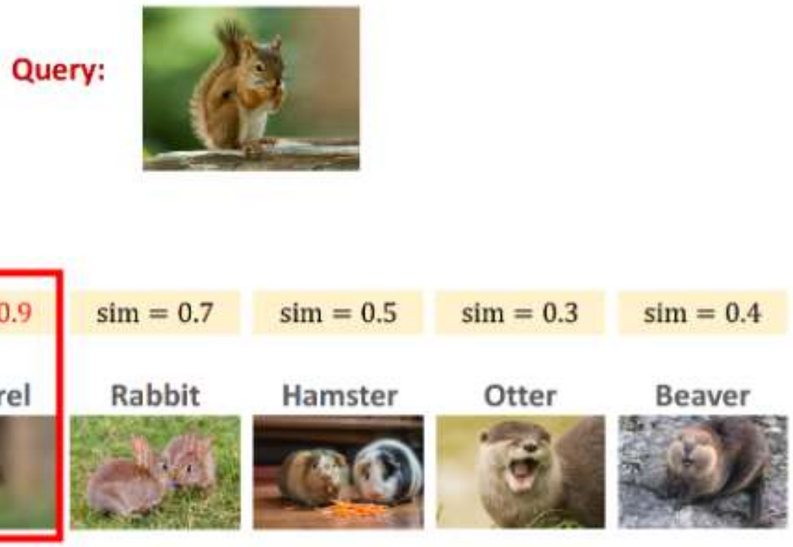
- Many-shot 학습
  - 정의: 대량의 학습 데이터를 사용하여 모델을 학습하는 전통적인 방법
  - 특징:
    - 데이터가 풍부할 때 가장 효과적
    - 대부분의 딥러닝 모델들이 이 방법을 사용
    - 사례: 대부분의 이미지 분류, 자연어 처리 모델들은 Many-shot 학습 방법을 사용



출처 : https://www.analyticsvidhya.com/blog/2022/12/know-about-zero-shot-one-shot-and-few-shot-learning/

Zero-shot, Few-shot, Many-shot 학습

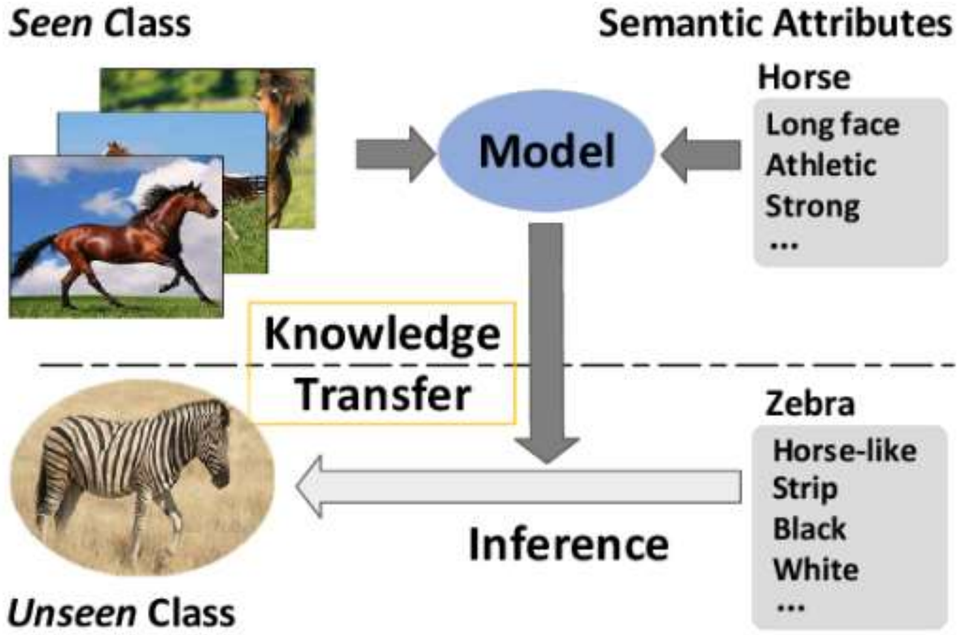
- Few-shot 학습
  - 정의: 제한된 양의 학습 데이터 (예: 몇 개의 샘플)만을 사용하여 모델을 학습하는 방법
  - 특징:
    - 데이터가 제한적인 실제 환경에서 유용
    - 사전 학습된 모델과의 조합이 흔히 사용됨
    - 대부분의 Fine-tuning은 Few-shot 학습에 해당함



출처 : https://www.analyticsvidhya.com/blog/2022/12/know-about-zero-shot-one-shot-and-few-shot-learning/

Zero-shot, Few-shot, Many-shot 학습

- Zero-shot 학습
  - 정의: 모델이 학습 과정에서 본 적 없는 카테고리나 태스크에 대해 예측을 수행하는 학습 방법
  - 특징:
    - 일반화 능력이 중요
    - NLP 분야에서는 상대적으로 많이 이루어짐



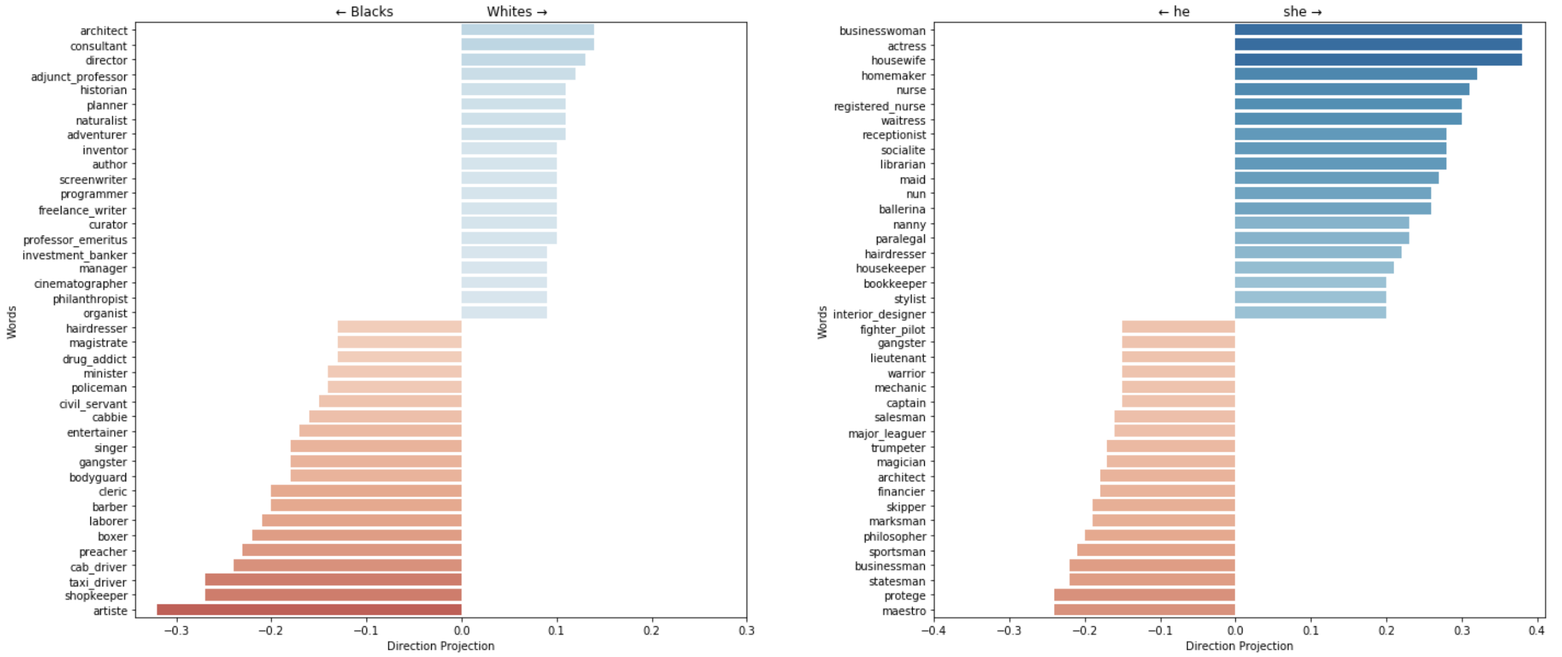
출처 : https://www.analyticsvidhya.com/blog/2022/12/know-about-zero-shot-one-shot-and-few-shot-learning/

04

거대 언어모델의 한계와 도전

데이터 편향성 문제

- 편향성의 원인
  - 거대 언어모델은 대규모의 텍스트 데이터에서 학습
  - 이 데이터에는 사회, 문화, 인간의 선입견 등 다양한 편향이 내재되어 있음



출처 : https://medium.com/institute-for-applied-computational-science/bias-in-nlp-embeddings-b1dabb8bbe20

데이터 편향성 문제

- 편향성의 결과
  - 모델은 학습 데이터의 편향을 그대로 반영
  - 예측, 분류, 생성 등의 작업에서 편향된 결과를 출력할 수 있음

Bias	#	Targets	Attributes	German BERT	German T5	German GPT-2
Conceptual	1	Flowers vs. Insects	Pleasant vs. Unpleasant	-0.22	0.61	0.25
	2	Instruments vs. Weapons	Pleasant vs. Unpleasant	0.58	0.11	0.15
	9	Mental vs. Physical Disease	Temporary vs. Permanent	0.16	0.5	0.54
Racial	3	Native vs. Foreign Names	Pleasant vs. Unpleasant	0.48	0.44	0.64
	4	Native vs. Foreign Names (v2)	Pleasant vs. Unpleasant	0.48	0.44	0.64
	5	Native vs. Foreign Names (v2)	Pleasant vs. Unpleasant (v2)	0.67	-0.38	0.74
Gender	6	Male vs. Female Names	Career vs. Family	0.61	-0.56	0.79
	7	Math vs. Arts	Male vs. Female Terms	0.4	0.73	0.14
	8	Science vs. Arts	Male vs. Female Terms	-0.24	0.22	-0.28

출처 : https://arxiv.org/abs/2209.10335

데이터 편향성 문제

- 도전과 문제점
  - 불평등 초래: 모델의 편향된 결과가 실제 세계의 의사 결정에 사용될 경우, 불평등이나 차별을 초래할 수 있음
  - 신뢰성 저하: 사용자는 모델의 예측이나 추천에 대한 신뢰성을 잃을 수 있음
  - 무엇보다, 아직까지 모델의 설명 가능성이 떨어지므로, 판단의 근거를 파악할 수 없는 것이 가장 큰 문제

데이터 편향성 문제

- 해결 방안
  - 데이터 다양성: 학습 데이터의 다양성을 높여 편향을 최소화
  - 편향 감지 도구: 모델의 출력을 분석하여 편향을 감지하는 도구 활용
  - 투명성 및 해석 가능성: 모델의 예측 기준 및 원인을 명확히 하는 연구 활발
  - 사용자 피드백: 사용자로부터의 피드백을 통해 모델의 편향을 교정

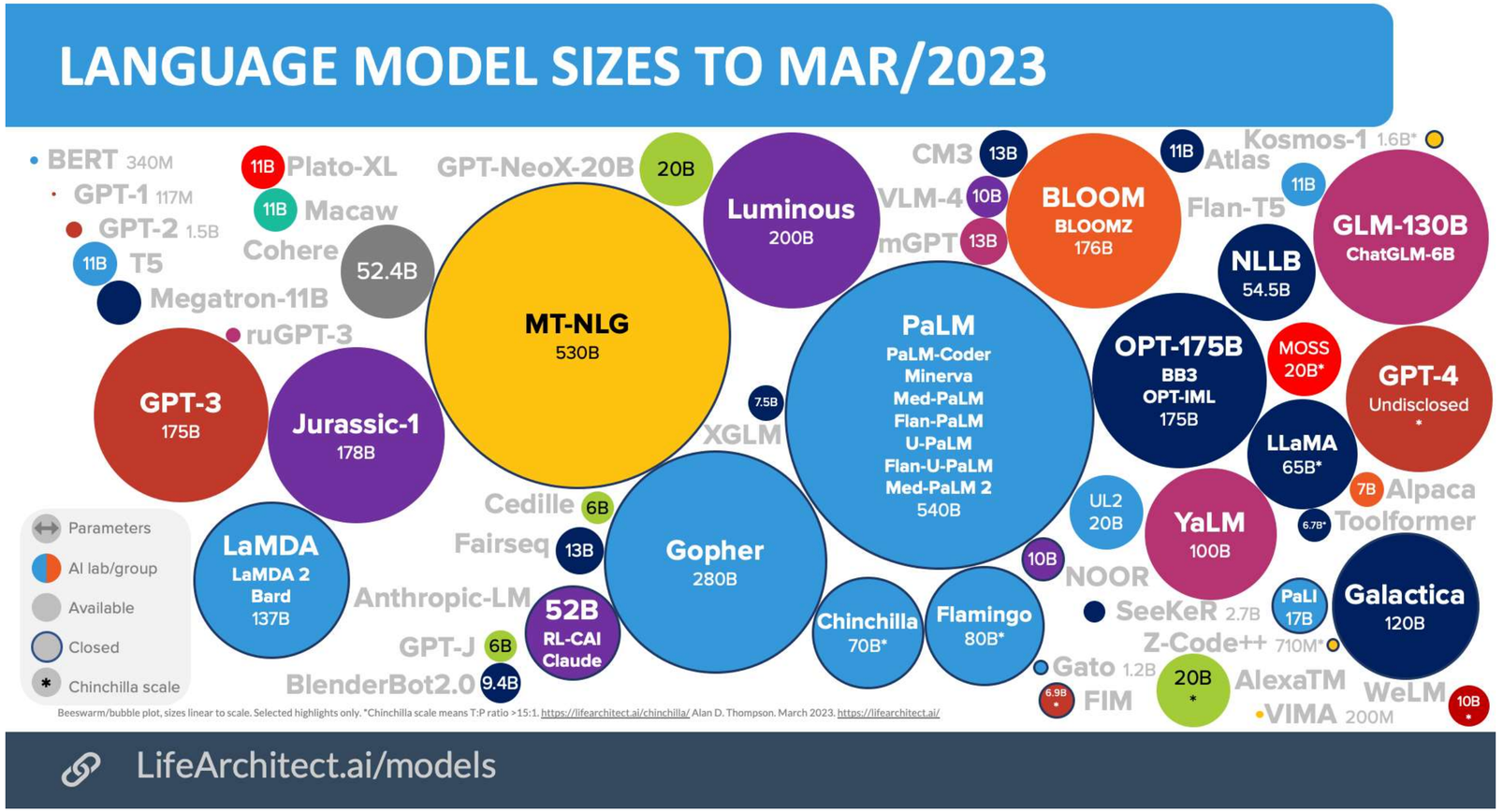
- 아직까지는 뚜렷한 해결책이 없는 상황

컴퓨팅 자원의 한계

- 모델 크기와 자원 요구량
  - 거대 언어모델은 수십억 ~ 수천억의 파라미터를 포함
  - 이러한 큰 모델을 학습하거나 실행하기 위해서는 상당한 컴퓨팅 자원이 필요
- 학습 시간
  - 대규모 데이터셋에서 거대 언어모델을 학습하는 데에는 수주 또는 수개월이 소요
  - 특히, 일반적인 하드웨어 환경에서는 학습이 불가능할 정도로 오랜 시간이 소요

04 거대 언어모델의 한계와 도전

컴퓨팅 자원의 한계

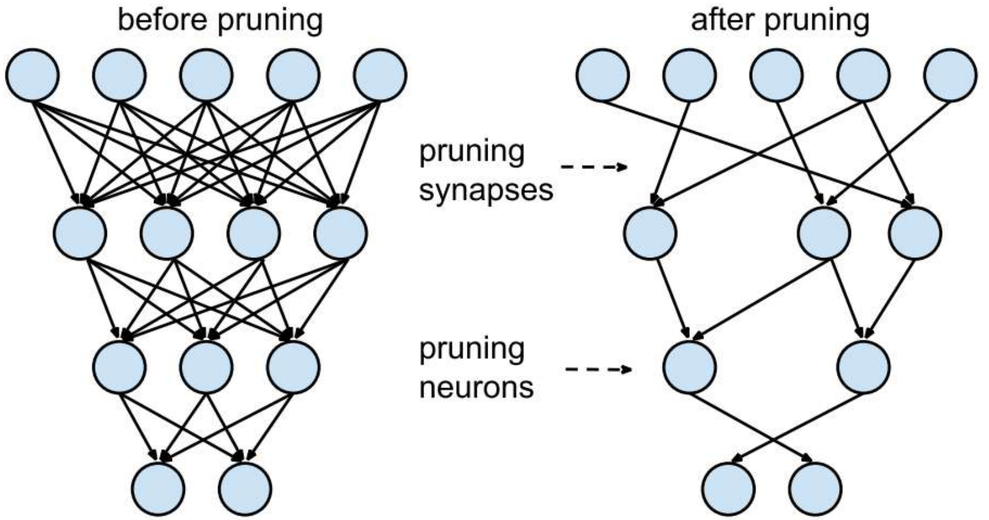


컴퓨팅 자원의 한계

- 해결 방안
  - 모델 최적화: 더 적은 파라미터로도 높은 성능을 달성할 수 있는 모델 아키텍처의 연구
  - 전이 학습: 사전 학습된 모델을 활용하여 추가 학습을 진행, 학습 시간과 자원을 절약
  - 모델 경량화: Pruning, Quantization 등의 기법을 활용하여 모델 크기를 줄이고 실행 속도를 향상

컴퓨팅 자원의 한계

- Model pruning
  - 모델의 크기를 줄이고, 추론 속도를 빠르게 하기 위한 기법
  - 파라미터 중 중요하지 않은 것들을 제거
    - 가중치 가지치기 (Weight pruning)
      - 가중치의 절대값이 특정 임계값보다 작은 경우 해당 가중치를 0으로 설정하여 제거
    - 유닛/뉴런 가지치기 (Neuron pruning)
      - 특정 뉴런의 가중치 벡터의 L2 노름이 임계값보다 작은 경우 해당 뉴런을 제거



컴퓨팅 자원의 한계

- Model quantization
  - 모델의 파라미터를 더 작은 비트로 표현함으로써 모델의 크기를 축소
    - 대부분의 모델은 32비트 부동소수점(float32)으로 가중치와 활성화 값을 저장
    - 양자화는 이러한 값을 16비트(float16) 또는 8비트(int8)와 같은 저정밀도로 변환
  - 두 가지 양자화 방식이 존재
    - 가중치 양자화: 모델의 가중치만 저정밀도로 변환
      - 학습 후에 수행
    - 동적 양자화: 모델의 가중치와 중간 계산 값을 저정밀도로 변환
      - 추론 시간에 수행