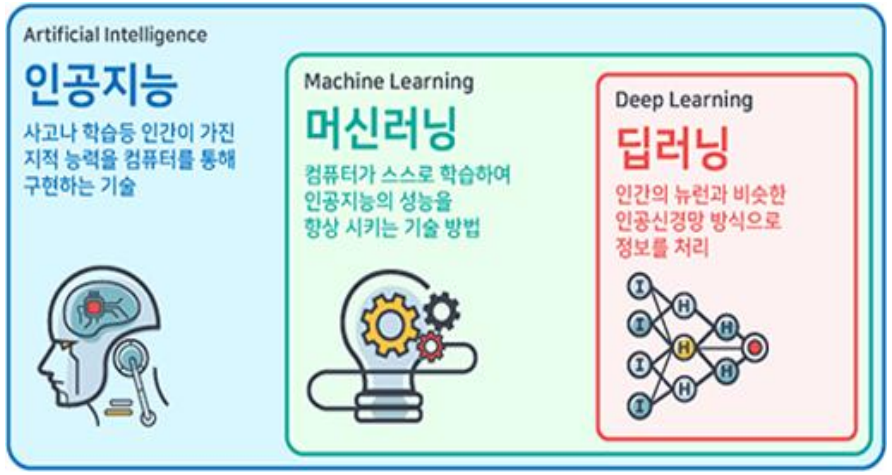
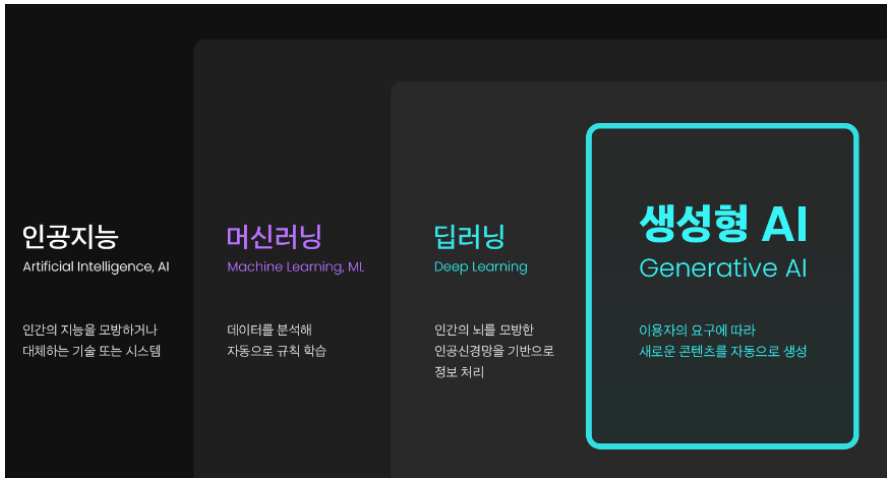
Deep-learning, Transformer, BERT, GPT

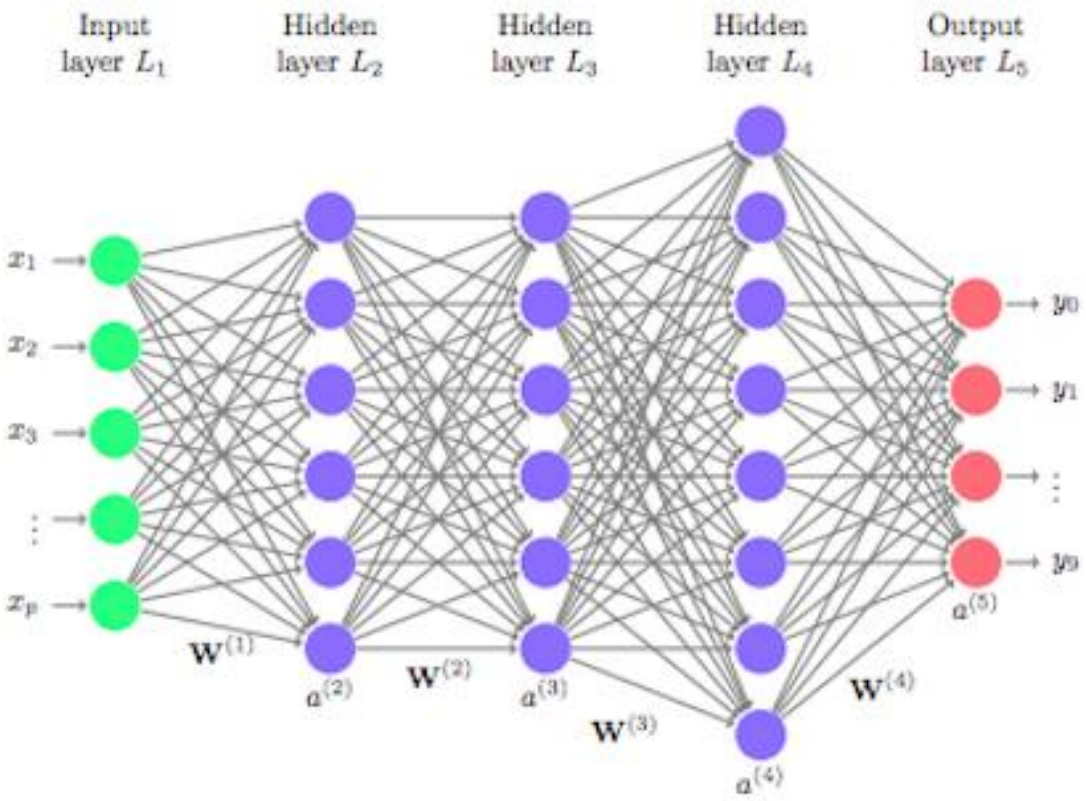
이종혁(경희대 미디어학과)

딥러닝(deep-learning)

딥러닝(deep-learning)

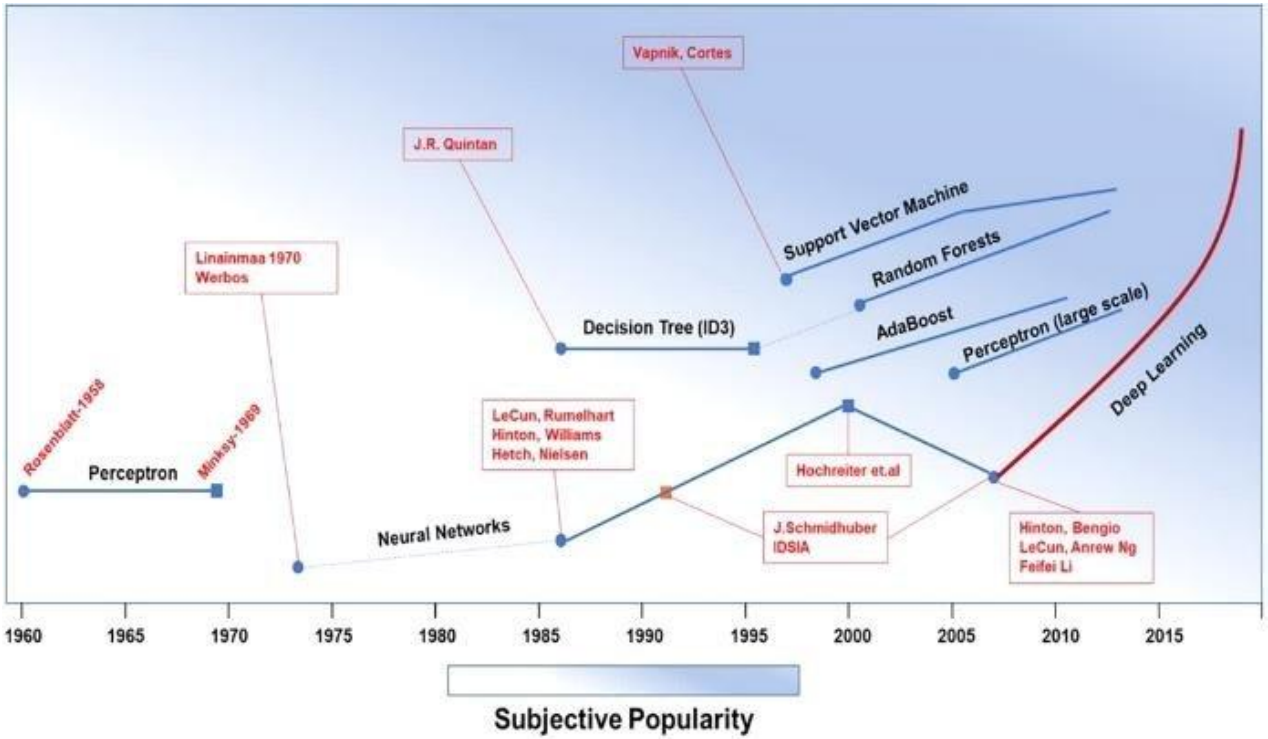


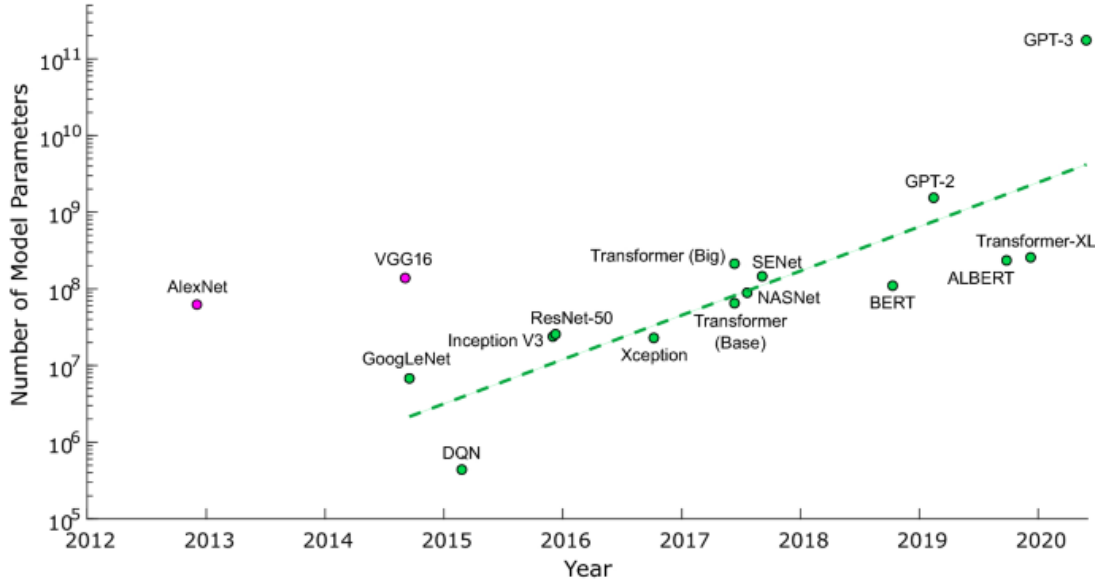


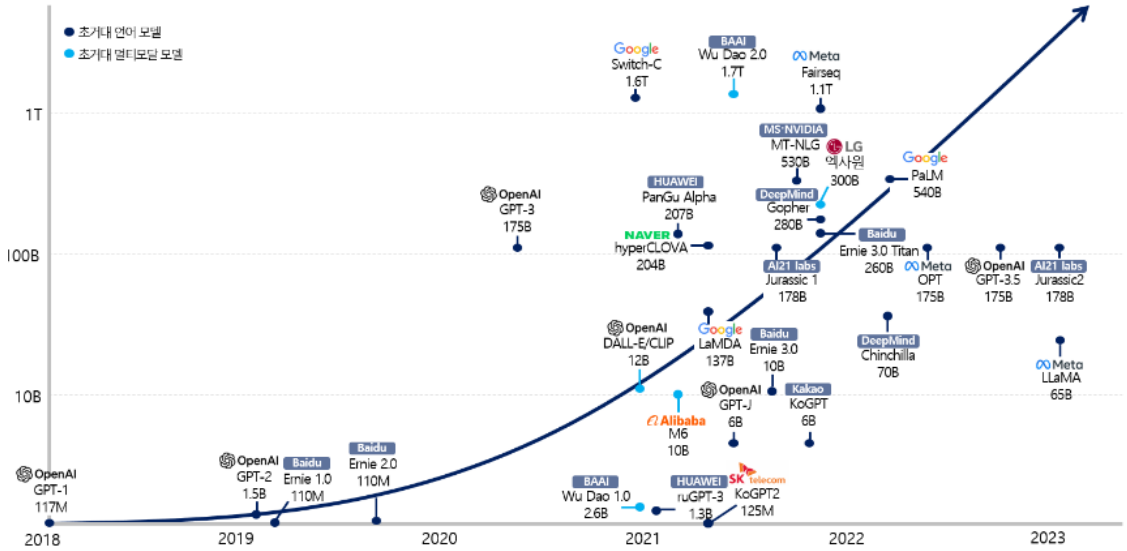


https://medium.com/@venkatesh.t.16072001/what-is-deep-learning-how-it-changes-the-world-f1dfb8b7e0ef
http://tobetong.com/?p=9393
https://news.skhynix.co.kr/post/all-around-ai-1

딥러닝의 발전

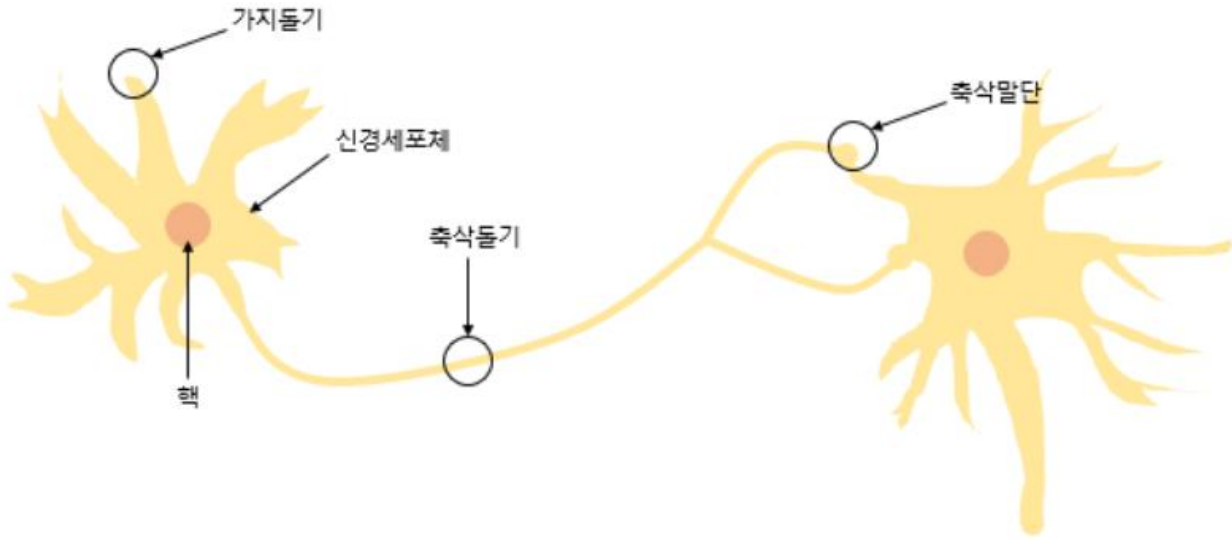




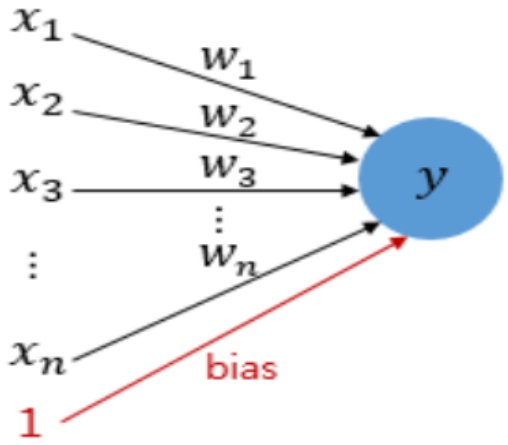


https://www.researchgate.net/figure/Historical-evolution-of-machine-learning-and-deep-learning_fig1_349864030
https://towardsdatascience.com/2019-year-of-bert-and-transformer-f200b53d05b9
https://www.nature.com/articles/s41598-021-82543-3
https://eiec.kdi.re.kr/publish/reviewView.do?ridx=14&idx=143&fcode=000020003600003

퍼셉트론(Perceptron)



- 프랑크 로젠블라트(Frank Rosenblatt)가 1957년에 제안한 초기 형태의 인공 신경망
- 다수의 입력 값을 계산해 하나의 결과를 내보내는 알고리즘
- 뇌를 구성하는 신경 세포 '뉴런' 기능과 유사: 뉴런은 가지돌기에서 신호를 받아들이고, 이 정보가 일정 수준 이상의 크기를 가지면, 축삭돌기를 통해 다음 뉴런에 전달.
- 입력 값과 가중치 곱의 전체 합이 임계치(threshold)를 넘으면, 인공 뉴런은 출력 신호로서 1을 출력하고, 그렇지 않으면 0을 출력 -> 아핀 함수(affine function)+ 활성화 함수(activation function)



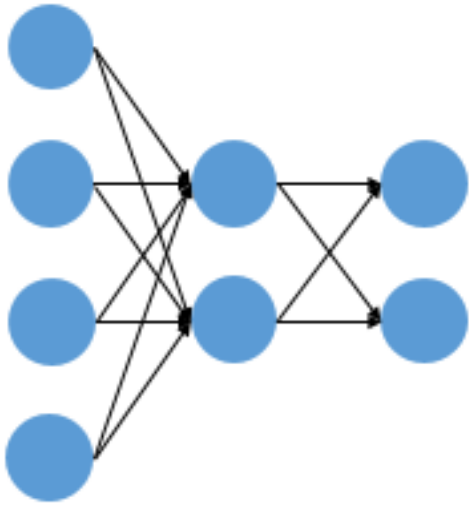
$$if \sum_i^n w_i x_i + b \geq 0 \rightarrow y = 1$$

$$if \sum_i^n w_i x_i + b < 0 \rightarrow y = 0$$

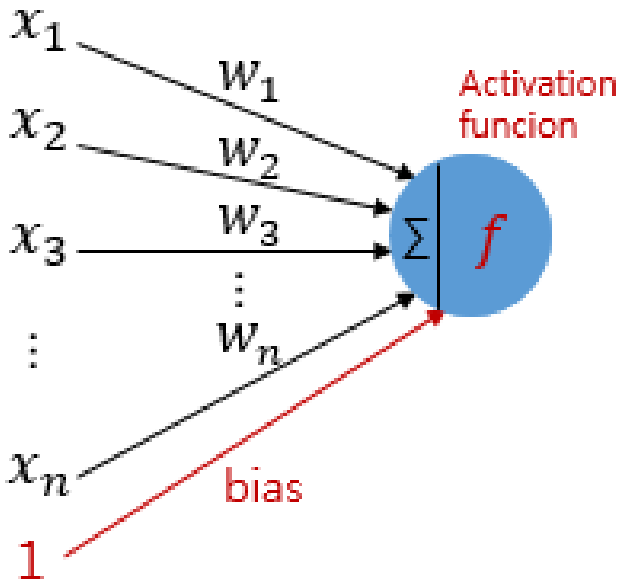
https://wikidocs.net/book/2155

딥러닝 모형의 구성

- 피드 포워드 신경망(Feed-Forward Neural Network, FFNN)
  - 입력층->은닉층->출력층 방향으로 연산되는 신경망
- 층(layer)
  - 연산 처리 과정을 층층이 나눠서 구분한 것
  - 입력층, 은닉층, 출력층
  - 전결합층(Fully-connected layer, FC) 또는 밀집층(Dense layer): 모든 뉴런이 이전 층의 모든 뉴런과 연결돼 있는 층
- 활성화 함수(Activation Function)
  - 은닉층과 출력층의 뉴런에서 가중치 연산 후 비선형 변환으로 출력 값을 결정하는 함수



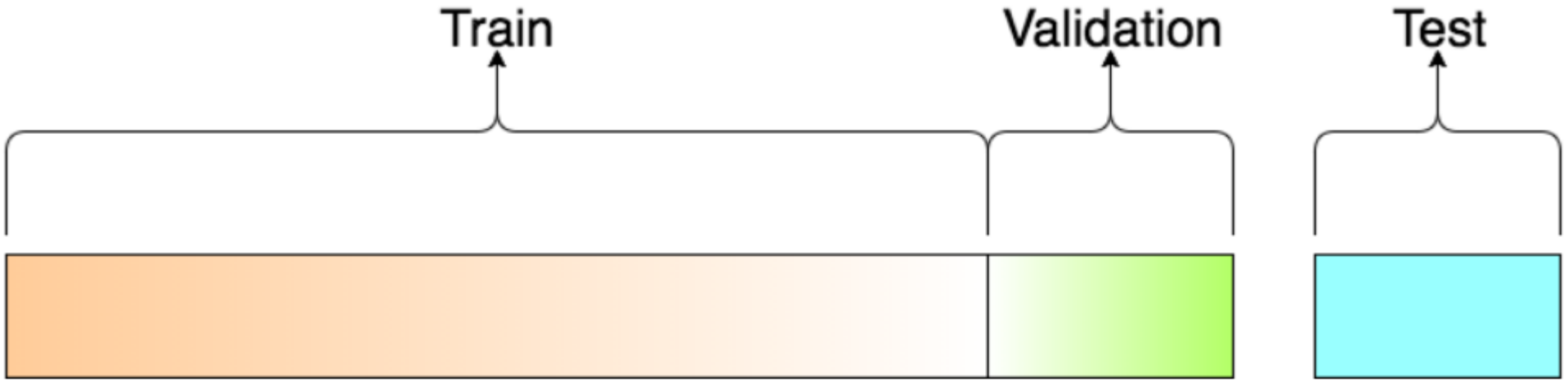
순방향 신경망(Feed-Forward Neural Network)



https://wikidocs.net/book/2155

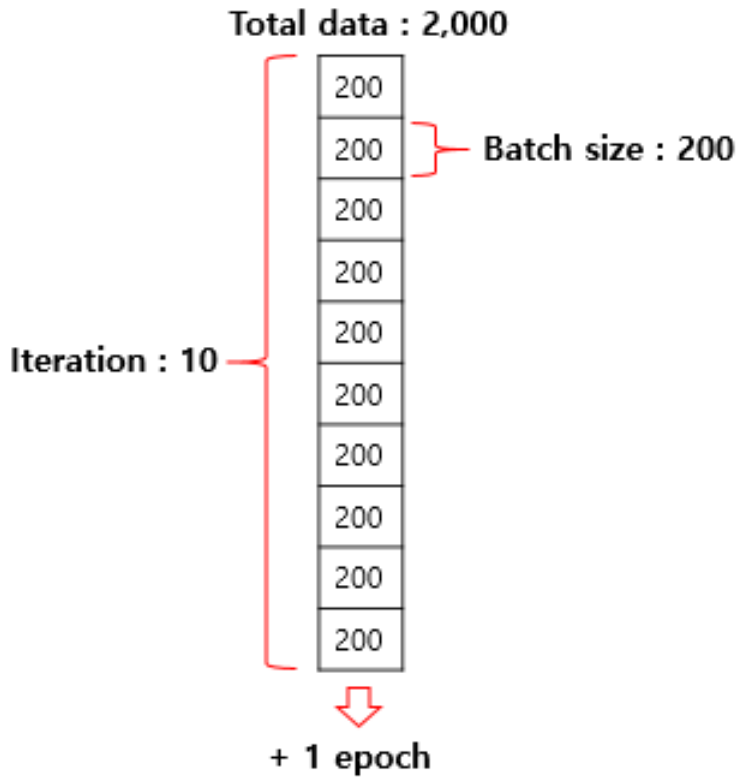
데이터의 분리(Splitting Data)

- 학습용(training)-검증용(Validation)-평가용(test) 데이터로 분리
- 학습용: 모델의 학습에 사용하는 데이터. 대부분이 학습에 사용됨 -> 문제풀기
- 검증용: 학습 단계별로 모델의 성능을 점검하는 데에 사용 -> 모의고사
- 평가용: 학습 완료된 모델의 최종 성능을 측정하는 데에 사용 -> 실제 시험



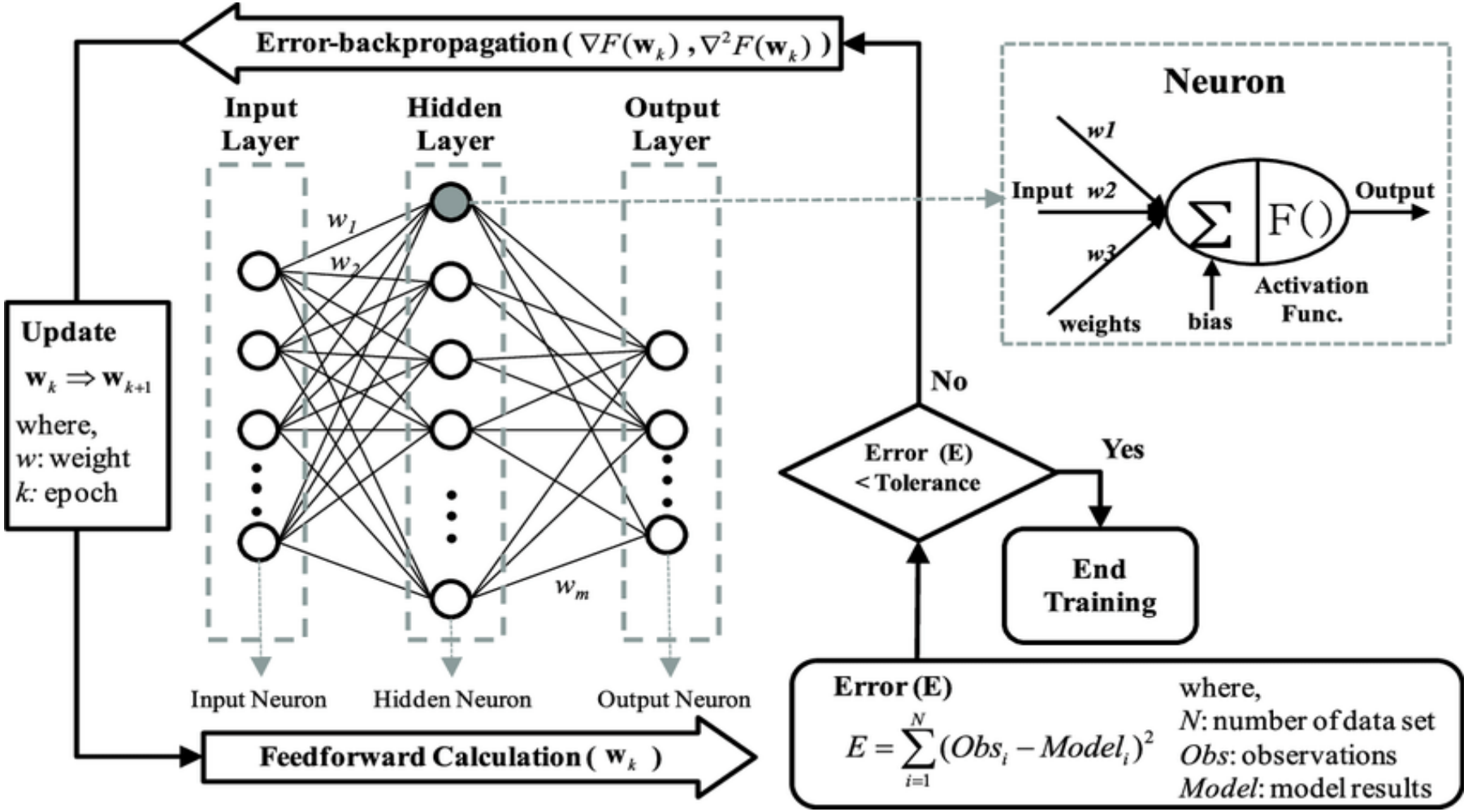
배치 크기(Batch size), 이터레이션(Iteration), 에포크(Epoch)

- 배치 크기(Batch size):
  - 전체 데이터 가운데 몇 개의 단위로 파라미터를 업데이트 하는지를 의미.
  - 전체 데이터가 2000일 때, 배치 크기를 200으로 하면 배치의 수는 10.
- 이터레이션(Iteration) 또는 스텝(Step)
  - 1회 에포크를 끝내기 위해서 필요한 배치의 수.
  - 전체 데이터가 2000일 때, 배치 크기를 200으로 한다면 이터레이션 수는 10. 에포크 1회당 파라미터 업데이트가 10번 이루어짐.
- 에포크(Epoch)
  - 전체 데이터에 대해서 순전파와 역전파가 1회 끝난 상태
  - 에포크가 10이라고 하면, 전체 데이터 단위로 총 10번 학습



https://wikidocs.net/book/2155

딥러닝의 작동 원리(종합)



https://www.researchgate.net/figure/Schematic-diagram-of-backpropagation-training-algorithm-and-typical-neuron-model_fig2_275721804

분류 성능 평가

- 혼동 행렬(Confusion Matrix)
- 정확도(Accuracy), 정밀도(Precision), 재현율(Recall), F1

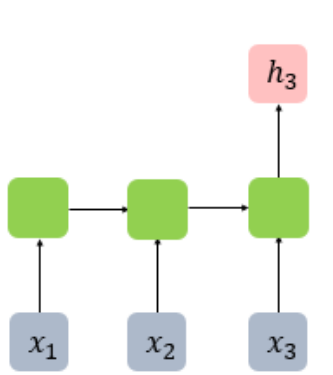
		Predicted		
		Negative (0)	Positive (1)	
Actual	Negative (0)	True Negative **TN**	False Positive **FP** (Type I error)	**Specificity** $= \frac{TN}{TN + FP}$
	Positive (1)	False Negative **FN** (Type II error)	True Positive **TP**	**Recall**, **Sensitivity**, True positive rate (**TPR**) $= \frac{TP}{TP + FN}$
		Accuracy $= \frac{TP + TN}{TP + TN + FP + FN}$	**Precision**, Positive predictive value (**PPV**) $= \frac{TP}{TP + FP}$	**F1-score** $= 2 \times \frac{Recall \times Precision}{Recall + Precision}$

https://velog.io/@jjw9599/ConfusionMatrix-ClassificationEvaluation

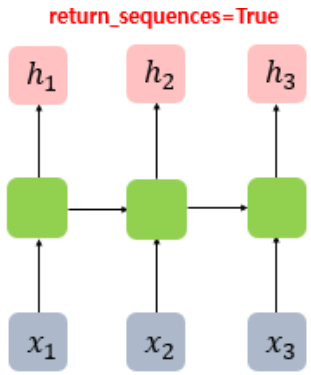
트랜스포머(Transformer)

RNN의 발전

Recurrent Neural Network

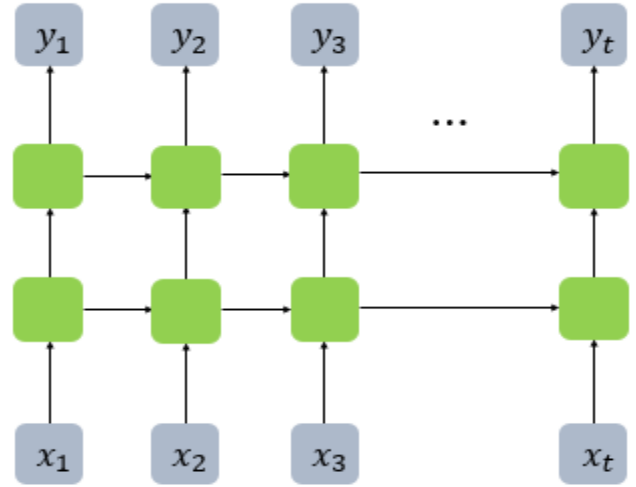


다음층으로 마지막 은닉 상태만 전달

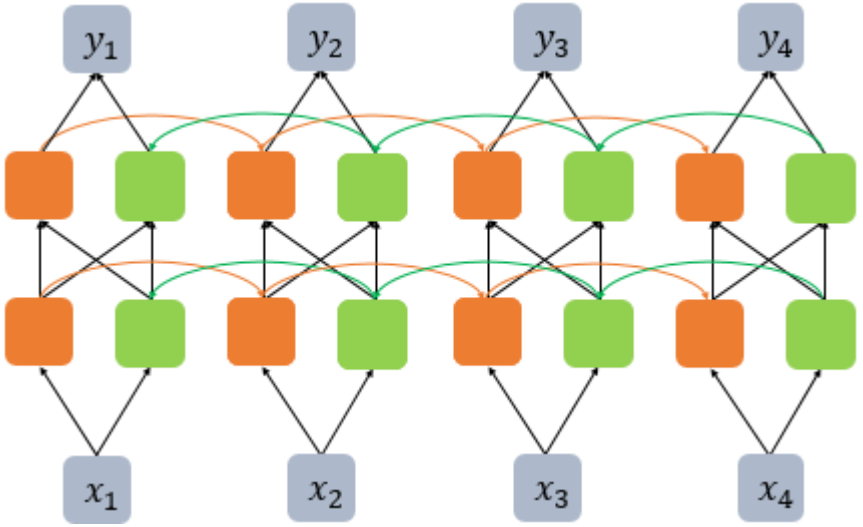


다음층으로 모든 은닉 상태 전달

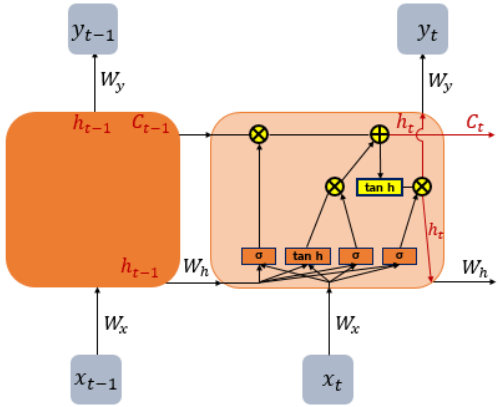
Deep Recurrent Neural Network



Bidirectional Recurrent Neural Network

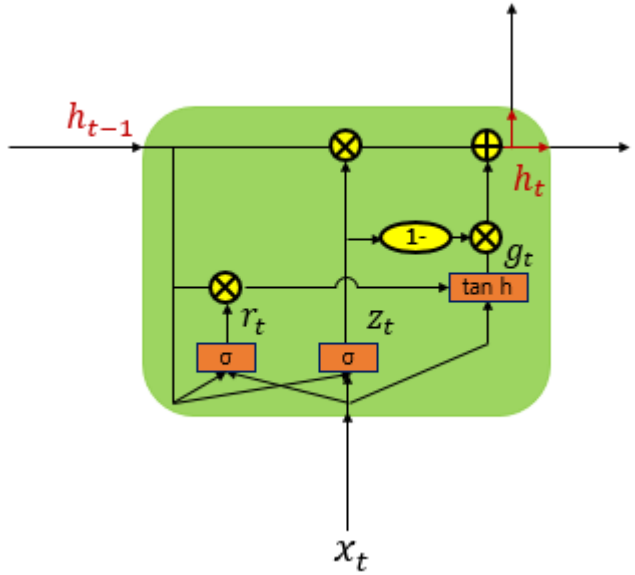


Long Short-Term Memory



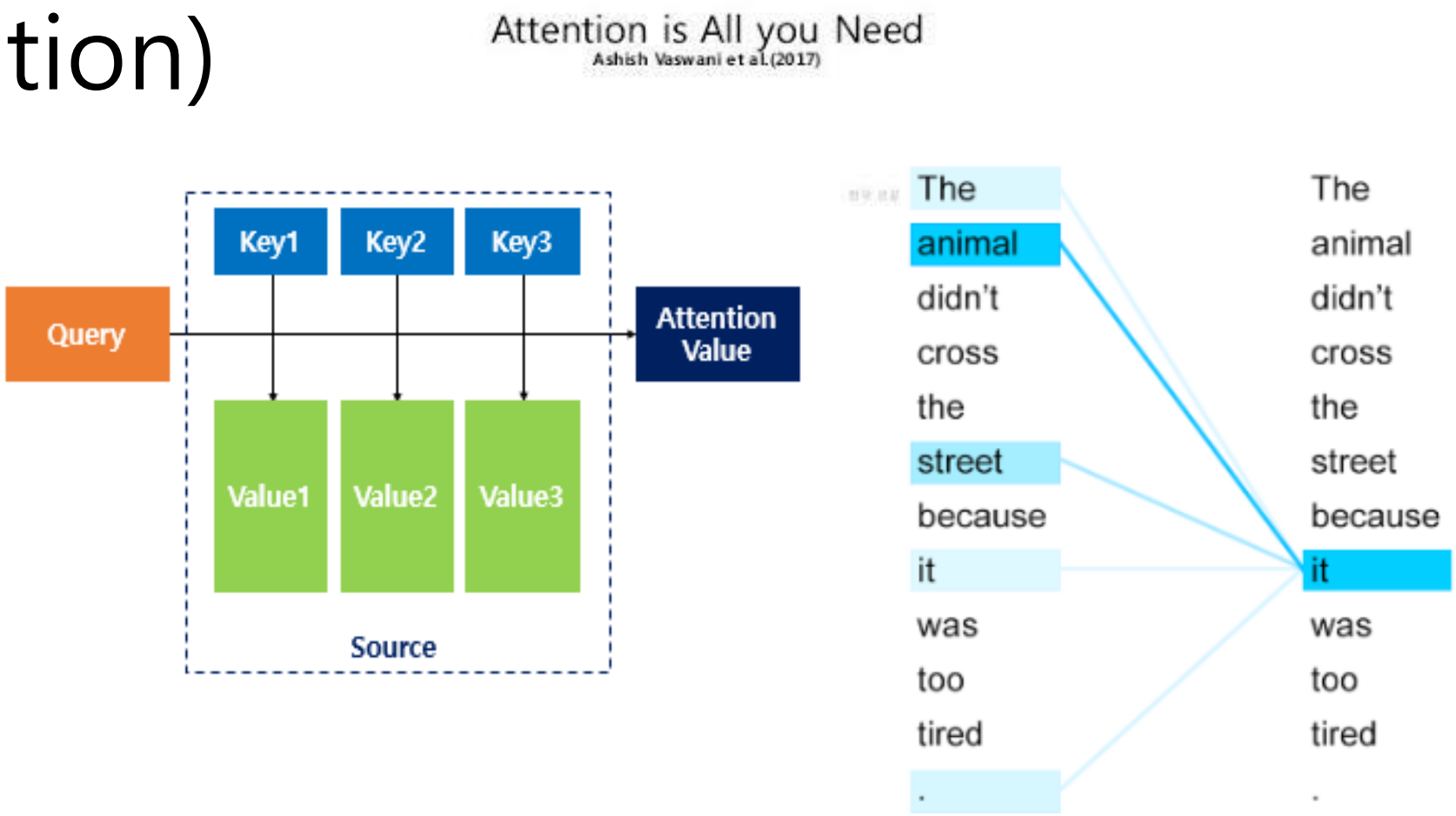
https://wikidocs.net/book/2155

Gated Recurrent Unit



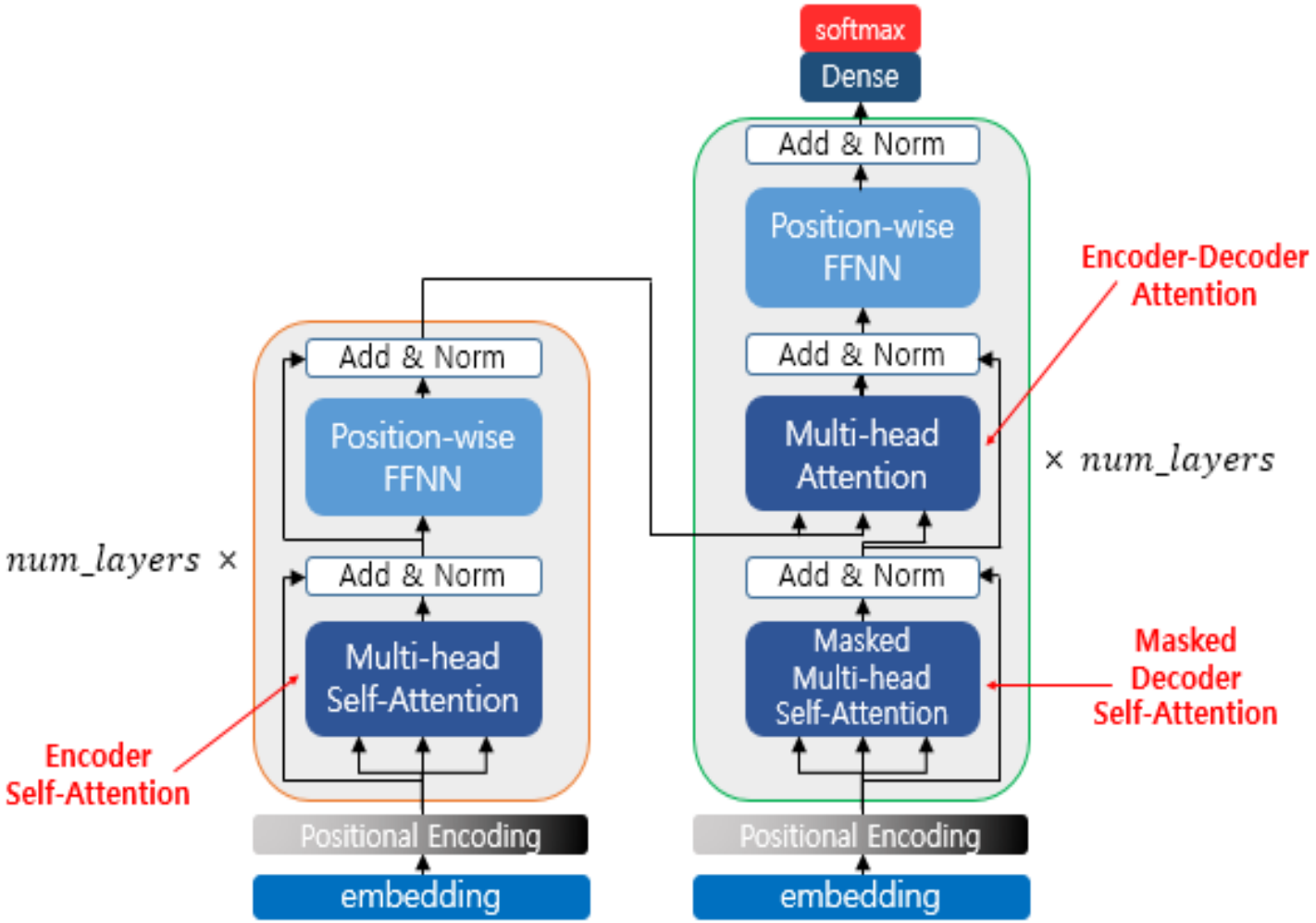
어텐션(attention)

- 단어별로 Q, K, V 벡터 생성
- 특정 단어의 Q에 대해 문장 내 모든 단어의 K와 내적 값 구하기
- 이 값을 가중치로 V 계산 (softmax 통과)
- 모든 단어의 계산된 V를 합해 attention 벡터 산출



https://wikidocs.net/book/2155

트랜스포머(Transformer)



- Input embedding vector 512 차원: 초기값으로 word embedding 방식 이용
- Positional encoding vector 512차원(embedding과 더해 512차원 유지): 가까울수록 높은 값 형성
- Encoder 6개
- Decoder 6개
- Multi-head 각 8개
- 번역이나 챗봇에 유용

- 번역에서 데이터 입력과 출력 사례
  - 인코더 입력: ["나는", "학생입니다"]
  - 디코더 입력: ["<sos>", "I", "am", "a", "student"]
  - 디코드 출력: ["I", "am", "a", "student", "<eos>"]

BERT

(Bidirectional Encoder Representations from Transformers)

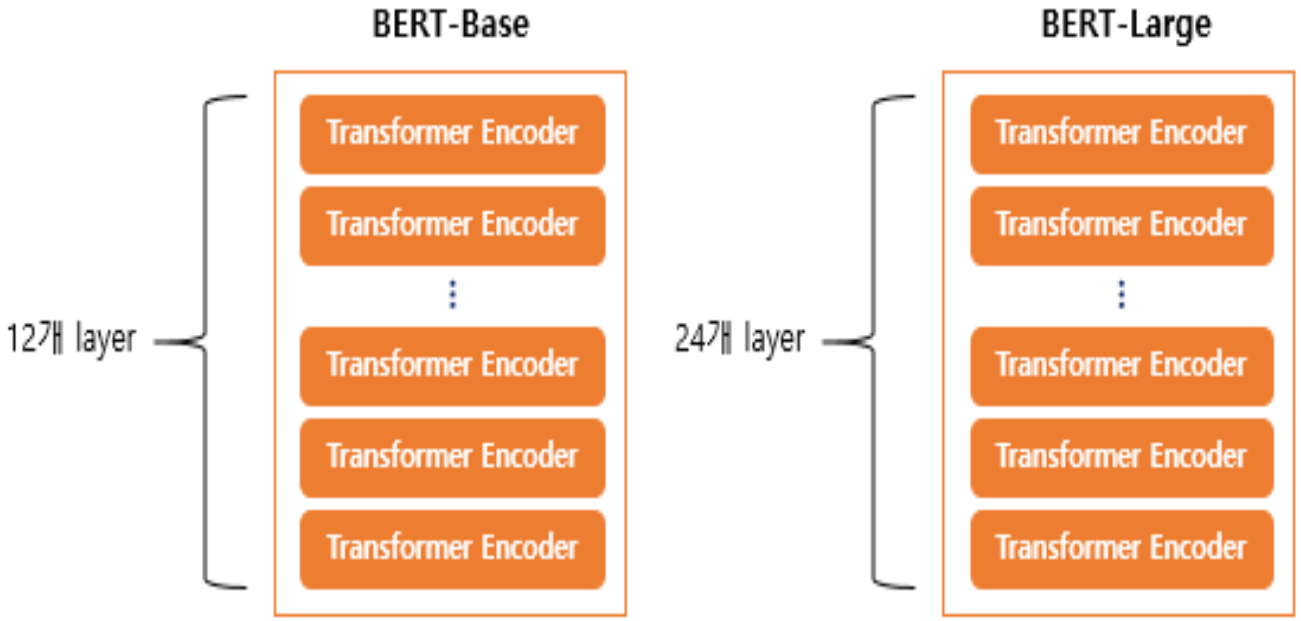
BERT

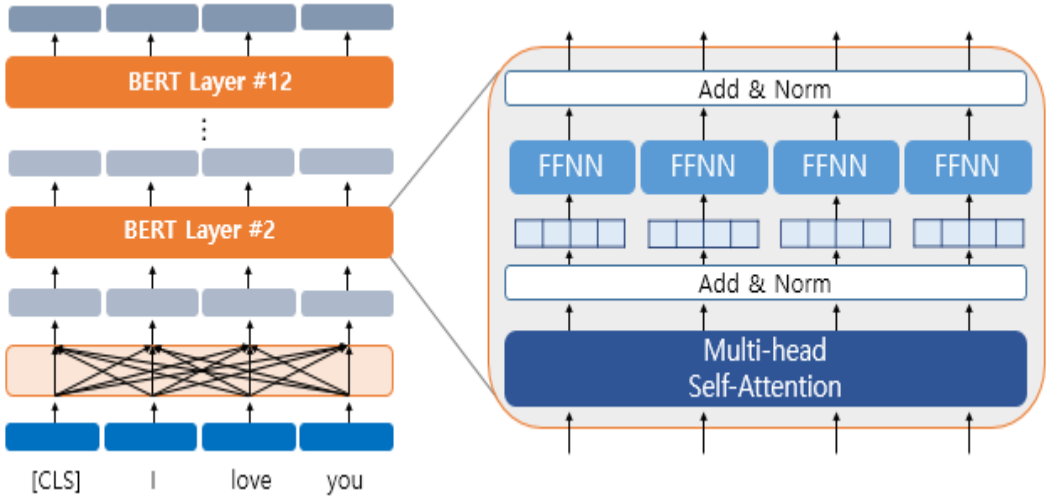
- 2018년에 구글이 공개한 사전 훈련된 언어 모델
- 위키피디아(25억 단어)와 BooksCorpus(8억 단어)와 같은 레이블이 없는 텍스트 데이터로 사전 훈련
- 전이학습(Transfer Learning) 활용
  - 사전학습된 모델(pre-trained model)을 기반으로, 특정 딥러닝 모형을 비교적 적은 규모의 데이터 학습과 파인 튜닝(Fine-tuning, 파라미터 재조정)을 통해 개발
- NLP 태스크에서 최고 성능

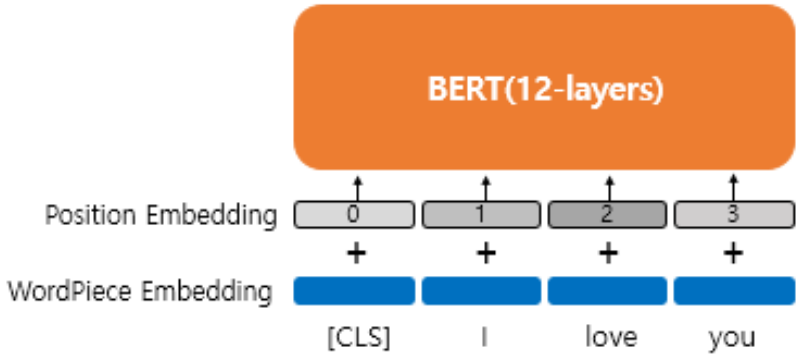
BERT의 구조

- 트랜스포머의 인코더를 여러 개 쌓아 올린 구조
- BERT-Base: L=12, D=768, A=12 : 110M개의 파라미터
- BERT-Large: L=24, D=1024, A=16 : 340M개의 파라미터

(L=인코더 층의 수, D=d_model의 크기(embedding 차원), A=셀프어텐션 헤드 수)







https://wikidocs.net/book/2155

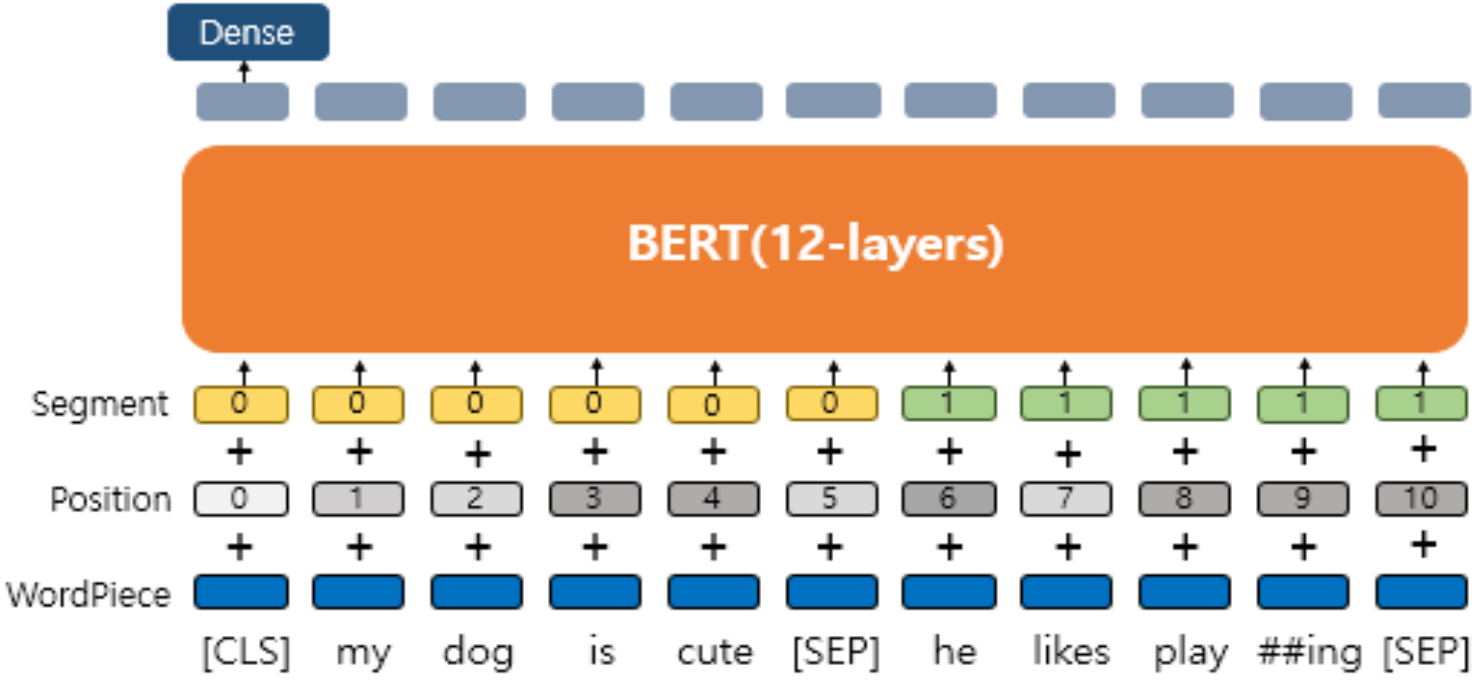
WordPiece embedding

- 단어보다 더 작은 단위로 쪼개는 subword tokenizer 사용
- 자주 등장하는 단어는 그대로 단어 집합에 추가
- 자주 등장하지 않는 단어는 더 작은 단위인 서브워드로 분리되어 단어 집합에 추가
- 단어 집합이 만들어지고 나면, 이 단어 집합을 기반으로 토큰화

```
from transformers import BertTokenizer
tokenizer = BertTokenizer.from_pretrained("bert-base-uncased")
tokenizer.tokenize('Here is the sentence I want embeddings for.')
['here', 'is', 'the', 'sentence', 'i', 'want', 'em', '##bed', '##ding', '##s', 'for', '.']
```

Position Embedding & Segment Embedding

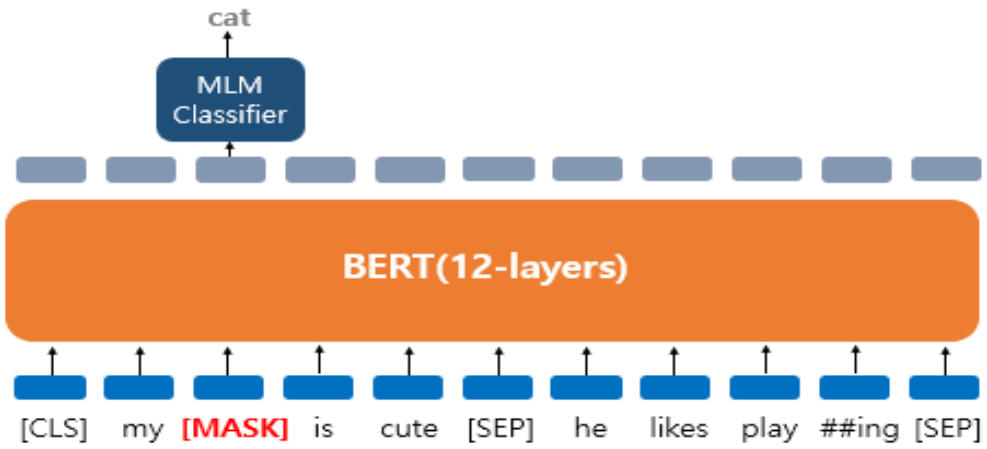
- 문장의 최대 길이 512로 제한
- 총 512개의 포지션 임베딩 벡터 할당 가능
- 두개 문장 입력 가능: 문장 시작에는 [cls], 문장 나누기와 끝에는 [sep]



https://wikidocs.net/book/2155

마스크드 언어 모델(Masked Language Model, MLM) 학습

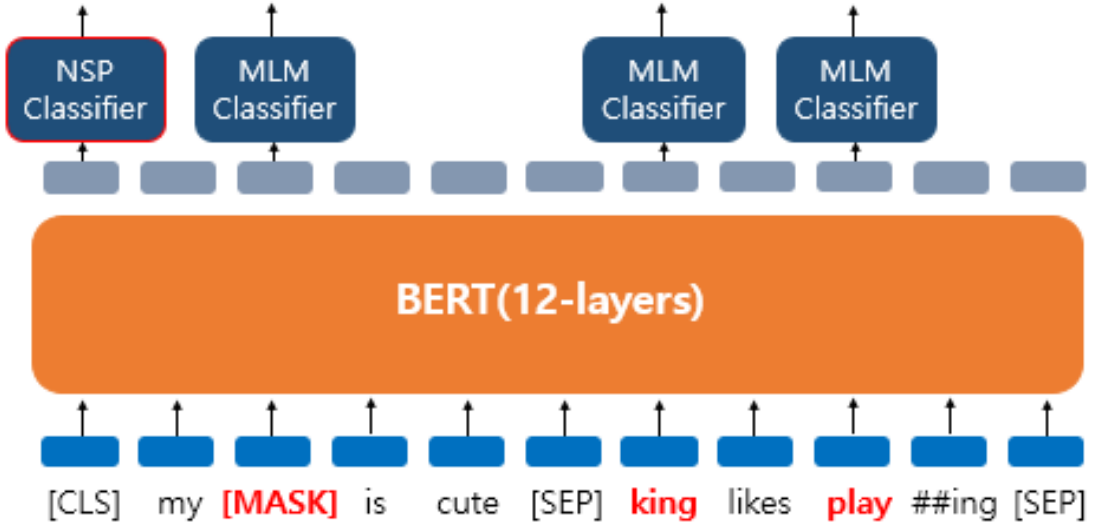
- BERT는 사전 훈련에서 입력 단어의 15%를 무작위로 마스킹(Masking)한 뒤 가려진 단어들을 예측하도록 학습
  - '나는 [MASK]에 가서 국어와 [MASK]를 공부했다'를 주고 '학교'와 '수학'을 맞추게 연습
  - 선택된 단어들은 세부적으로 다음과 같이 활용
    - 80%의 단어들은 [MASK]로 변경
    - 10%의 단어들은 랜덤으로 단어 변경
    - 10%의 단어들은 동일하게 유지



https://wikidocs.net/book/2155

다음 문장 예측(Next Sentence Prediction, NSP) 학습

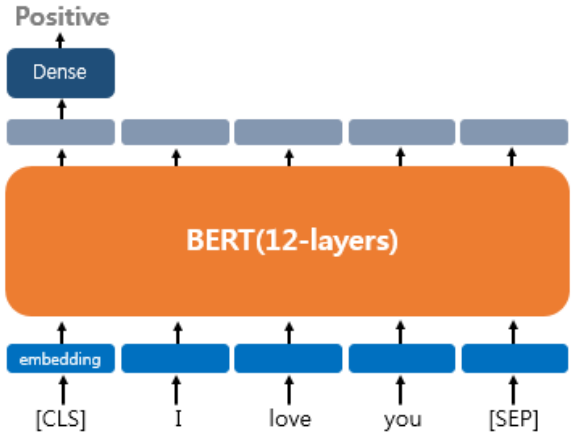
- 두 개의 문장을 주고 두번째 문장이 이어지는 문장인지 아닌지를 맞추는 학습
- 실제 이어지는 두 개의 문장과 무작위로 이어붙인 두 개의 문장을 50:50 비율로 주고 학습



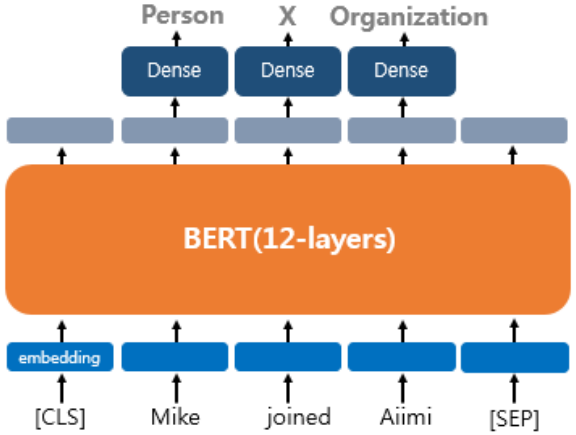
https://wikidocs.net/book/2155

파인 튜닝(Fine-tuning)

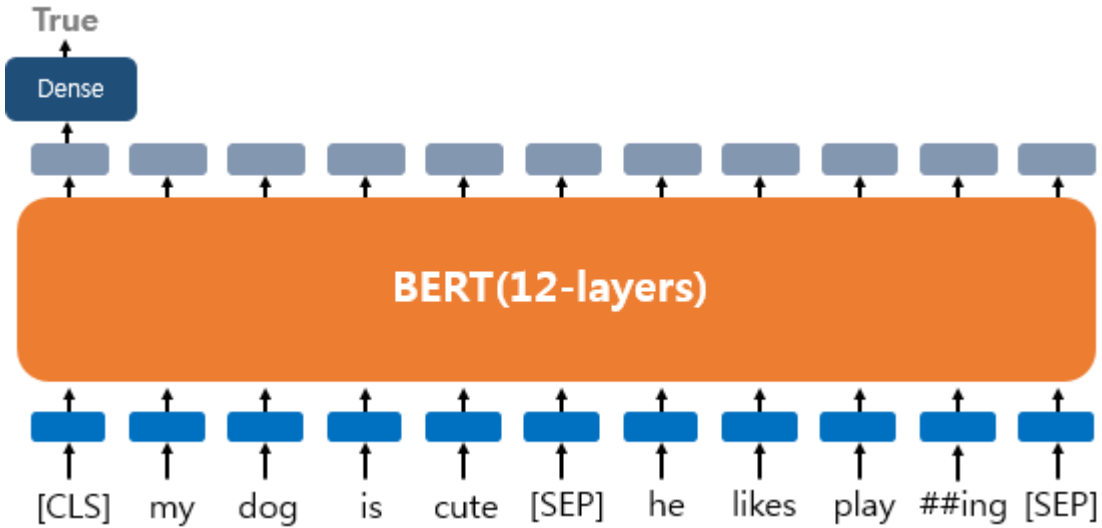
한 문장의 성격 분류



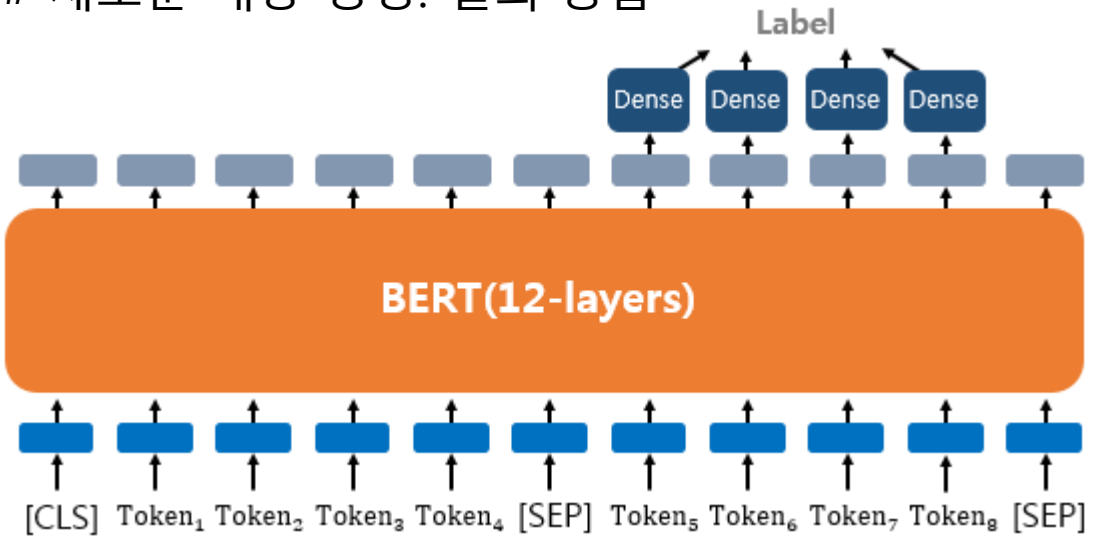
단어에 대한 태깅: 개체명이나 품사 분석



문장 간 관계 분석: 자연어 추론



새로운 내용 생성: 질의 응답

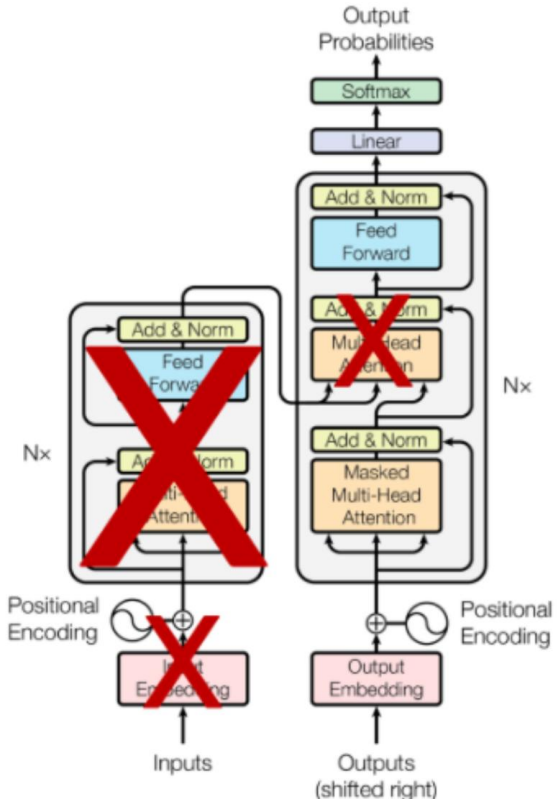


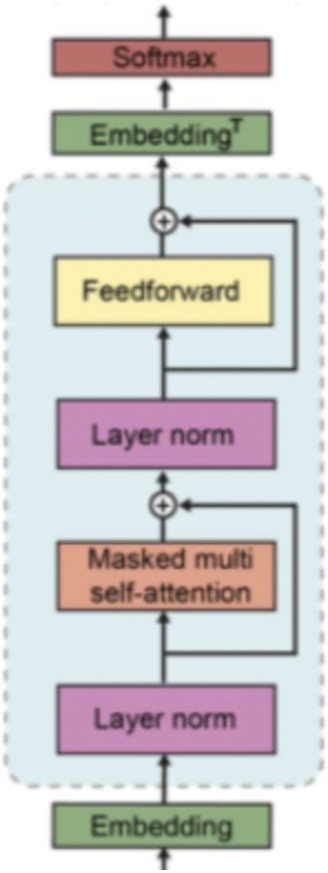
GPT
(Generative Pre-trained Transformer)

GPT

- 생성형 대형 언어 모델 (large language model, LLM): Next Word Prediction

구 분	BERT(Google)	GPT(OpenAI)
출시일	2018.11.	2018.6.
모 델	인코더 모델(Encoder only model)	디코더 모델(Decoder only model)
연산 방법	양방향 연산 나는 [____] 피자를 먹었다.	단방향 연산 나는 어제 피자를 [____].
강점	문장 이해, 분류에 강점	문장 생성에 강점
관련 모델	RoBERTa, ALBERT 등	GPT-2~4 등

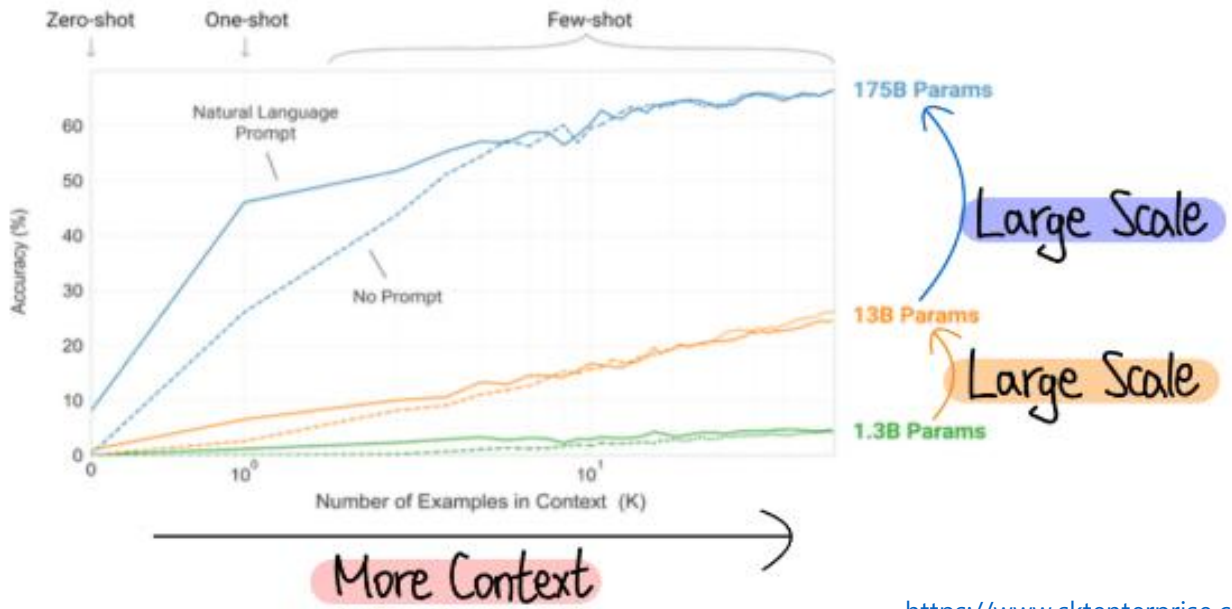


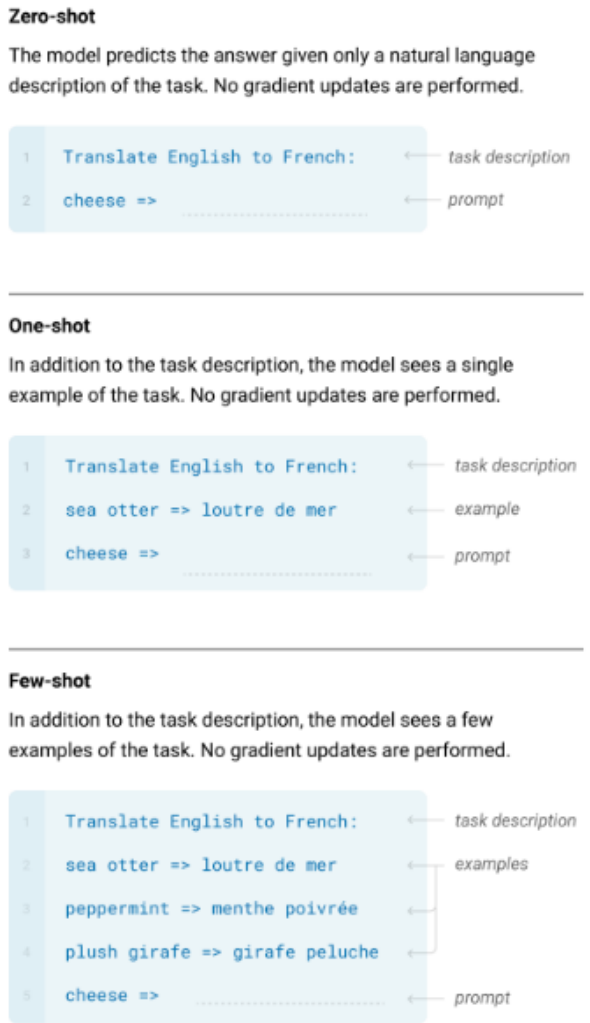


https://sooftware.io/gpt/
https://eiec.kdi.re.kr/publish/reviewView.do?ridx=14&idx=143&fcode=000020003600003

GPT의 발전

- 파라미터 수와 학습 규모 대폭 증가
  - GPT-1은 1억1700만개, GPT-2는 15억개, GPT-3는 1750억개, GPT-4는 1조7000억개
- 모델 학습: 지시(instructions)-응답(response) 데이터셋 + 인간 피드백을 통한 강화 학습(RLHF, Reinforcement Learning from Human Feedback)으로 발전
- 프롬프트 통한 학습: in-context learning

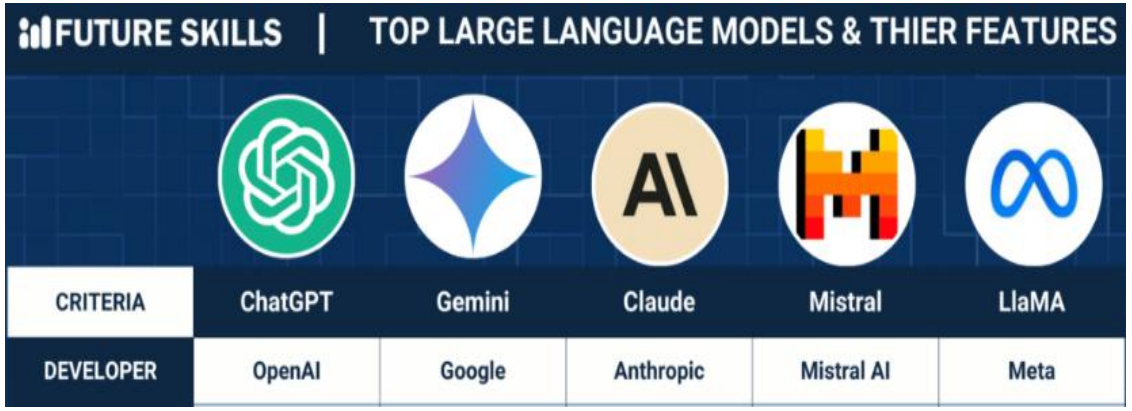


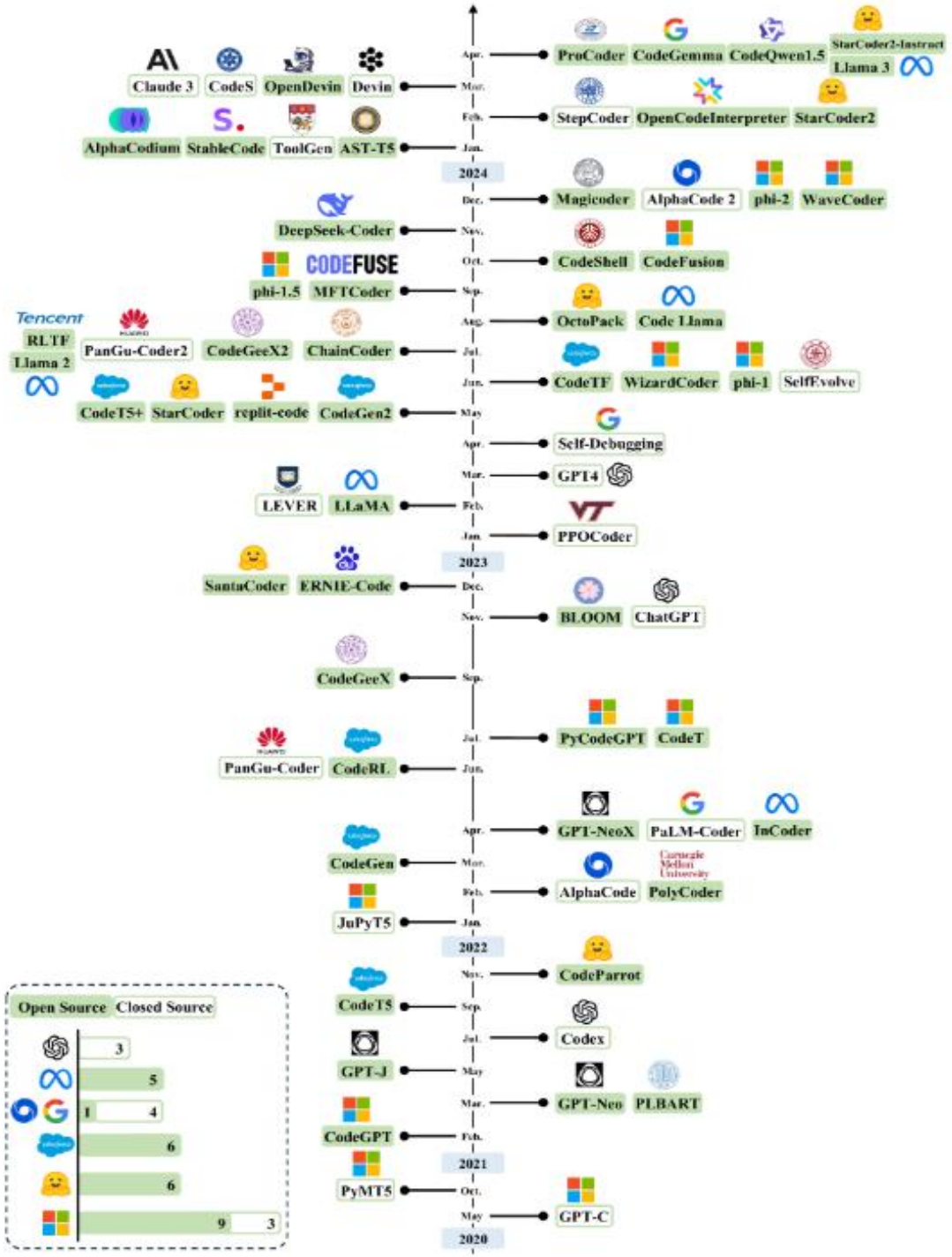


https://www.sktenterprise.com/bizInsight/blogDetail/dev/4222
https://ffighting.net/deep-learning-paper-review/language-model/gpt-3/

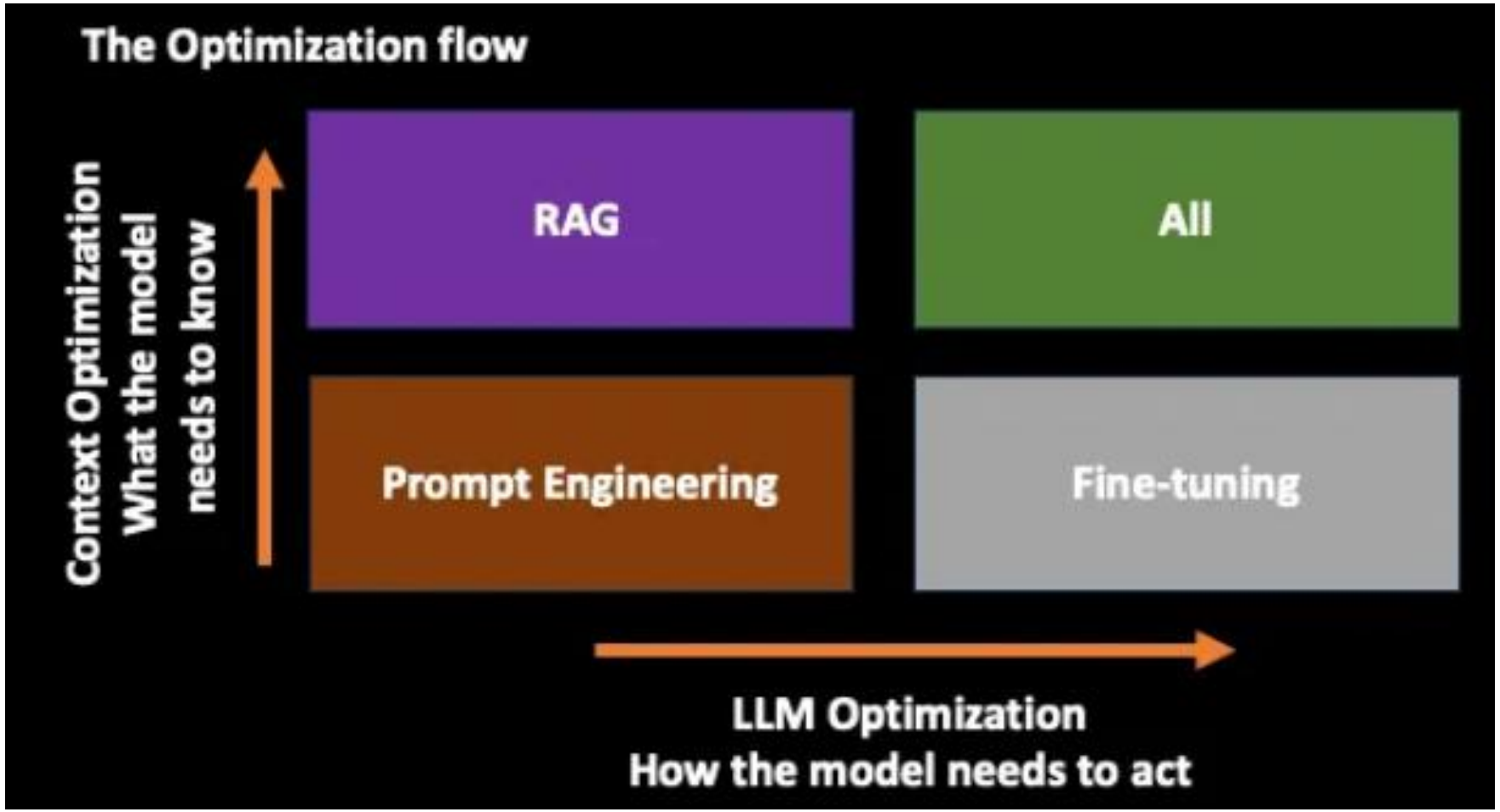
LLM 전성시대:

오픈소스 모델(publicly available) 포함





GPT 활용과 조정:
Prompt Engineering, RAG, Fine-tuning



https://humanloop.com/blog/optimizing-llms

프롬프트 엔지니어링(Prompt engineering)

•**페르소나 부여**: 역할 기반 응답

당신은 뉴스 분석 전문가입니다.
아래 뉴스 기사에 등장하는 정보원을 추출하시오.
뉴스 기사: 실제 기사

•**예시 제공(few shot learning)**: 형식과 스타일 학습

다음은 정보원 추출의 예입니다:
예시 기사:
환경부는 국내 미세먼지 농도가 개선되고 있다고 발표했다.
하지만 환경단체 그린피스는 "정부의 발표가 과장되었다"며
"독립적인 검증이 필요하다"고 강조했다.
예시 결과: [환경부, 그린피스]
아래 뉴스 기사에 등장하는 정보원을 추출하시오.
뉴스 기사: 실제기사

•**배경-맥락 설명**: 작업 이해도 증가

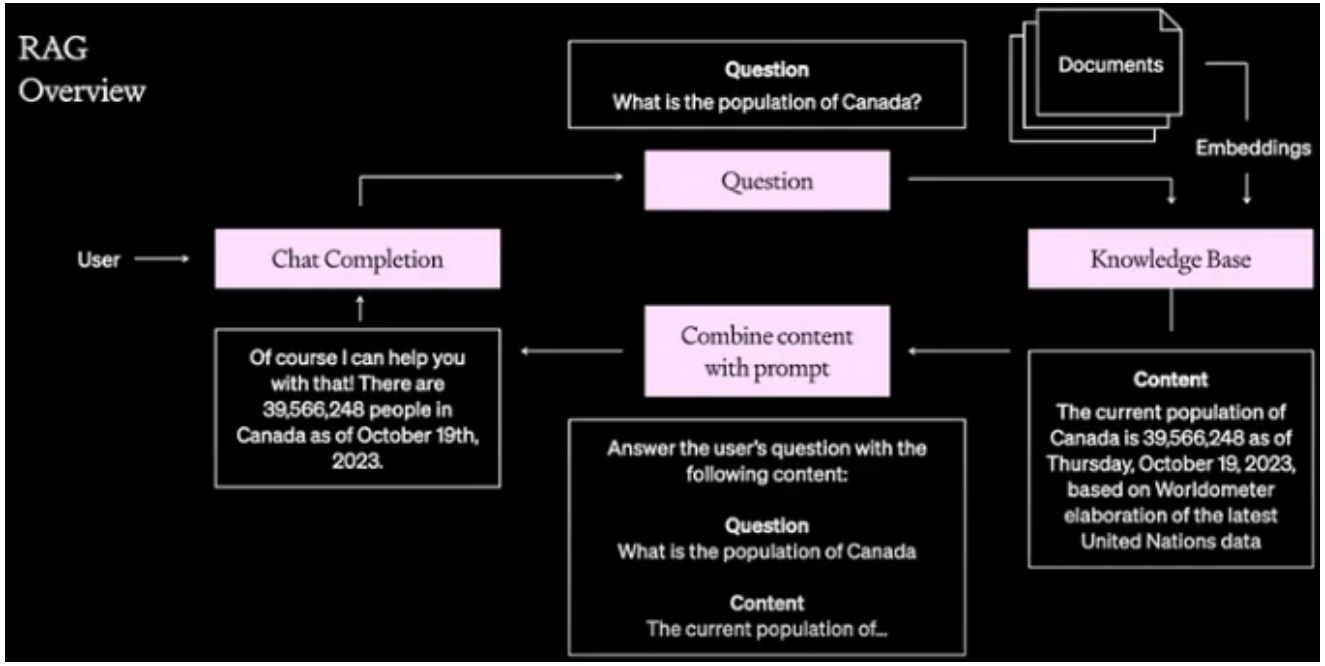
정보원은 기사 내용의 전달자이며, 기사에서 실명이나 익명으로
출처가 표시되어 드러납니다.
아래 뉴스 기사에 등장하는 정보원을 추출하시오.
뉴스 기사: 실제 기사

•**논리적 추론(**CoT) **유도**: 복잡한 문제 해결

뉴스 기사에서 정보원을 추출하고, 각 정보원의 유형을 분류하시오.
뉴스 기사: 실제기사
단계:
1. 주요 발언을 확인하세요.
2. 발언의 정보원를 추출하세요.
3. 정보원의 유형을 [공식, 비공식, 익명] 가운데 하나로 분류해.

RAG(Retrieval Augmented Generation)

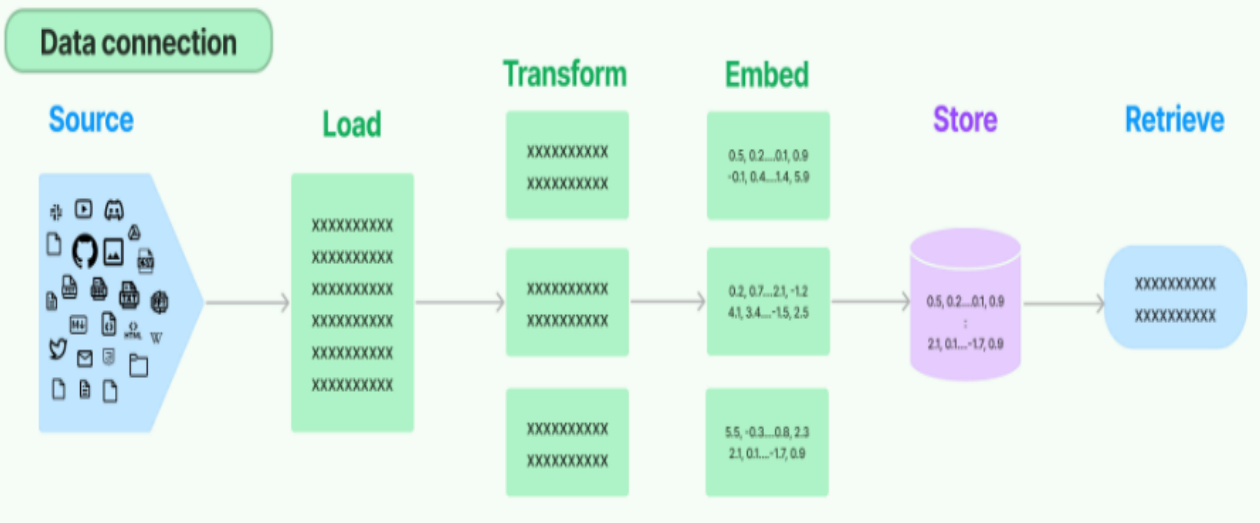
- LLM은 학습 데이터에 없는 최신 정보나 특정 도메인 지식은 제공하기 어려움
- 모델이 외부 데이터 소스를 활용해 최신 정보나 특정 도메인 지식을 통합하도록 함
  - 예를 들어, 금융 자문에서는 주가 정보나 경제 보고서를, 헬스 챗봇에서는 연구 논문이나 건강 지침을 참조하도록 함

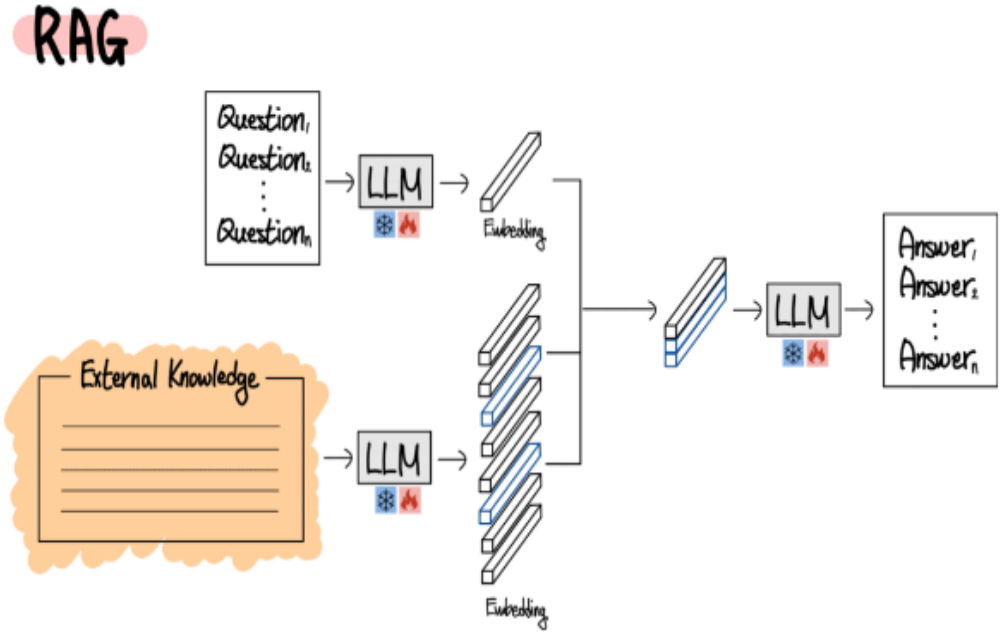


https://humanloop.com/blog/optimizing-llms

RAG(Retrieval Augmented Generation)

- RAG: 외부 데이터 소스를 활용하기 위해 '지식 검색'과 '언어 생성'을 결합한 프레임워크
- LangChain 사용해 원본 데이터 -> 적재(load) -> 분할(split) -> 임베딩 -> 벡터 공간 저장과 반환(vector store & retriever) -> 생성(generation)

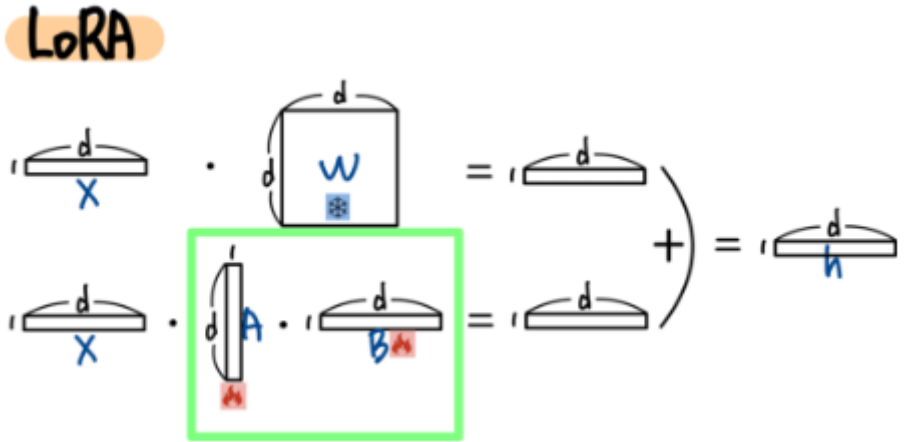




https://ffighting.net/deep-learning-paper-review/language-model/rag/
https://velog.io/@udonehn/RAG%EB%A5%BC-%EC%A0%81%EC%9A%A9%ED%95%9C-%EC%A7%88%EC%9D%98%EC%9D%91%EB%8B%B5-%EC%B1%97%EB%B4%87-LangChanin
https://wikidocs.net/231600

PEFT(Parameter Efficient Fine Tuning)

- 사전학습된 LLM의 파라미터 대부분을 프리징(freezing)하고, 일부 파라미터만을 파인튜닝한 뒤, 원하는 데이터셋을 학습시켜 특정 도메인에서 우수한 성능을 나타내는 모델 개발 가능
- 방법: LoRA(Low-Rank Adaptation), IA3(**I**nfused **A**dapter by **I**nhibiting and **A**mplifying **I**nner **A**ctivations)
- 장점
  - 필요한 계산량 감소: 모델 크기를 줄이고, 중복된 파라미터를 제거
  - 추론 속도의 향상: 모델 크기가 작아지고 계산 복잡도가 줄어들면서 추론 속도 빨라짐
  - 적은 용량으로 웹 배포 용이
  - 비용 절감: API 사용 없이 자신만의 모델 활용
- https://huggingface.co/models?library=peft



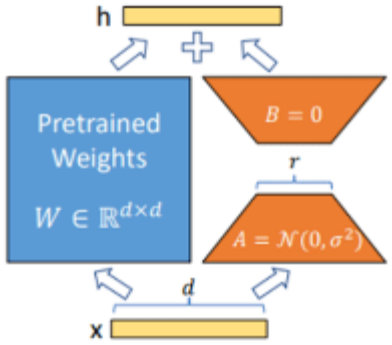


Figure 1: Our reparametrization. We only train A and B.

https://ffighting.net/deep-learning-paper-review/language-model/lora/

Hugging Face

- https://huggingface.co/
- AI community
- 각종 pretrained model, PEFT, nvidia NeMo 등 저장과 제공