



LLM 시작하기



Sections



01 기본 용어 이해

05 RAG 논문의 주요 내용

02 전이학습의 두 단계

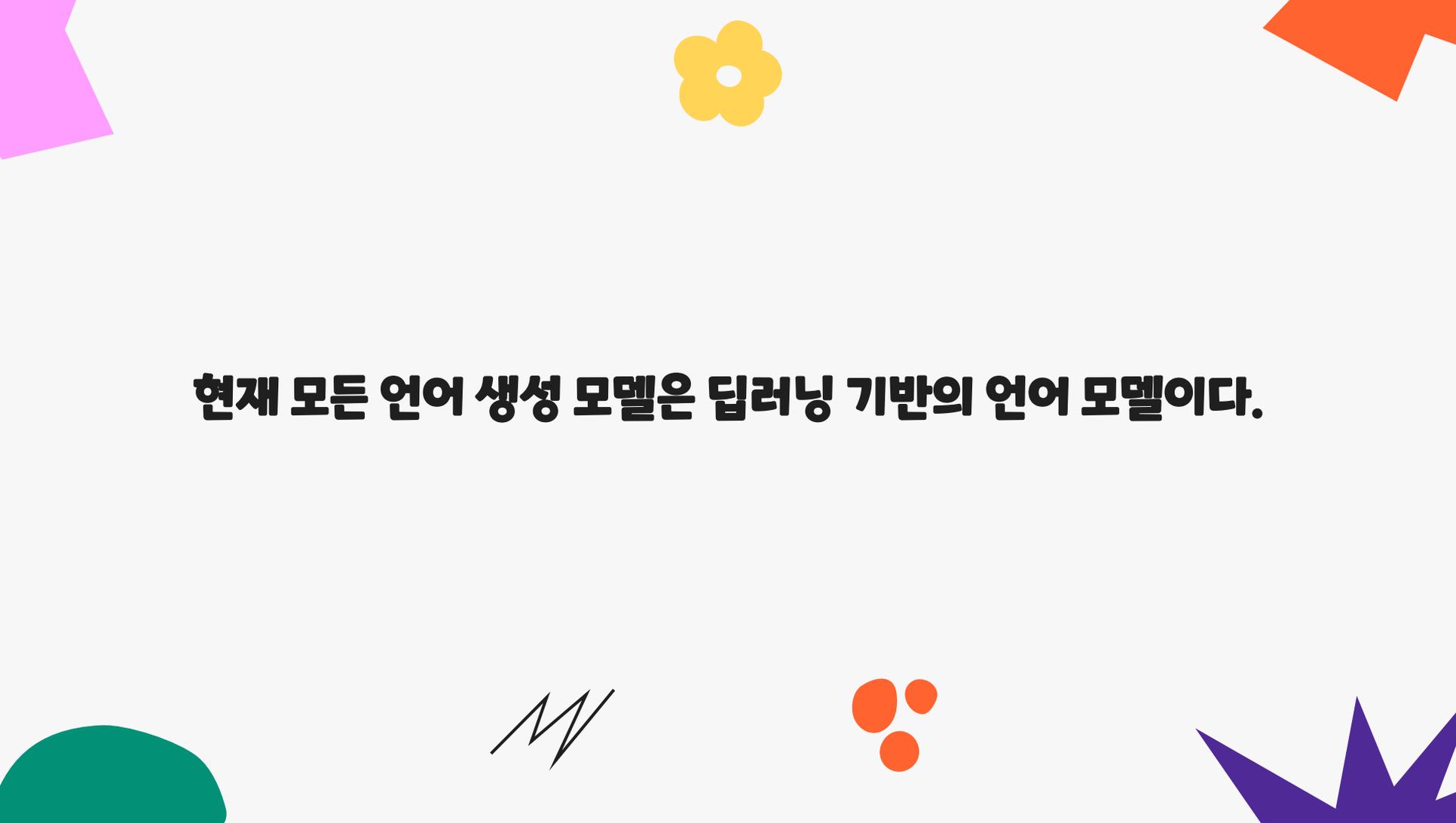
06 RAG 관련 라이브러리

03 현재의 ChatGPT까지의 과정

07 기본 용어 이해 - API

04 기본 용어 이해 - RAG





현재 모든 언어 생성 모델은 딥러닝 기반의 언어 모델이다.

현재의 LLM에까지의 기본 언어모델

구글에서 단어의 의미를 숫자로 표현하는 word2vec, 기계 번역 성능을 높이기 위한 트랜스포머 아키텍처, 2018년 GPT-1 모델 공개는 LLM에 영향을 끼친 중요한 것들 중의 하나이다.

01

2013년 워드투벡
word2Vec

02

2017년 트랜스포머 아키텍처

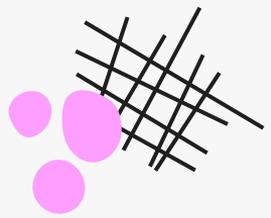
03

2018년 OpenAI GPT-1
(트랜스포머 활용)



01. 기본 용어 이해 - LLM

LLM은 "Large Language Model"의 약자입니다. 쉽게 말해서, 이걸 엄청나게 많은 글과 말을 학습한 인공지능(AI)이야. 마치 책을 수백만 권 읽고, 사람들의 대화를 엄청 많이 들어본 똑똑한 로봇이라고 생각하면 됩니다.



01. 기본 용어 이해 - LLM

왜? 대규모 언어 모델일까?

- 방대한 데이터 학습

수백만 권의 책, 인터넷에 있는 글, 사람들이 주고받는 대화 같은 걸 다 학습.

- 거대한 파라미터 수

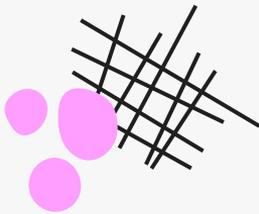
LLM은 많은 수의 파라미터(모델의 학습된 가중치)를 가지고 있어, 복잡한 언어 구조를 이해하고 생성하는 능력이 뛰어남.

- 다양한 작업 수행

대규모의 언어 모델은 텍스트 생성, 요약, 번역, 질문 응답 등 다양한 언어 관련 작업을 수행

01. 기본 용어 이해 - 토큰화

토큰화는 자연어 처리에서 중요한 개념 중 하나입니다. 토큰화란 말이나 글을 작은 조각으로 나누는 과정이며 대형 언어 모델의 가장 기본적인 단계 중의 하나입니다.



01. 기본 용어 이해 - 토큰화

토큰화는 어떻게 사용될까?

텍스트 입력 처리: 여러분이 "안녕하세요, 오늘 날씨가 좋네요!"라고 입력하면, LLM은 이 문장을 바로 이해할 수 없어요. 먼저 이 문장을 더 작은 조각들(토큰)으로 나누기. 예를 들면 "안녕", "하세요", ",", "오늘", "날씨", "가", "좋", "네요", "!" 같은 방법으로 나누기.

언어 이해: LLM은 각 토큰에 숫자(ID)를 부여하고, 이 숫자들을 이용해 문장을 이해. 이건 마치 사전에서 단어를 찾는 것과 비슷하다.

문맥 파악: 모델은 토큰들 사이의 관계를 분석해 전체 문장의 의미를 파악

응답 생성: 답변을 만들 때도 먼저 생각을 토큰 단위로 조합한 다음, 이를 자연스러운 문장으로 변환.

모델 크기 제한: LLM은 한 번에 처리할 수 있는 토큰 수가 제한되어 있어요. 예를 들어 어떤 모델은 한 번에 4,000토큰, 또 다른 모델은 100,000토큰까지 처리할 수 있죠. 이 제한이 여러분이 한 번에 입력할 수 있는 텍스트의 길이를 결정해요.

01. 기본 용어 이해 – 토큰화 과정

문장 분할하기

긴 글을 작은 조각으로 나누는 첫 단계. 예를 들어 "나는 학교에 갔어요"라는 문장이 있다면, 이것을 단어 단위로 먼저 쪼개기.

의미 단위 찾기

단어를 더 작은 의미 단위로 나눈다. "학교에"는 "학교"와 조사 "에"로 나눌 수 있고, "갔어요"는 동사 "가다"의 과거형 "갔"과 어미 "어요"로 나눌 수 있음.

불필요한 것 정리하기

문장에서 분석에 꼭 필요하지 않은 것들(특수 문자나 일부 구두점)을 처리. 예를 들어 "!"나 "@" 같은 특수 문자는 별도로 처리하거나 제외할 수 있음.

고유 번호 부여하기

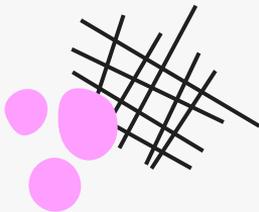
각 토큰(의미 단위)에 고유한 번호를 붙여요. 예를 들어 "나"는 123번, "는"은 456번, "학교"는 789번처럼 각각 다른 번호 부여.

01. 기본 용어 이해 - 임베딩

임베딩(Embedding)은 자연어 처리에서 중요한 개념중의 하나로서, 임베딩은 단어나 문장과 같은 이산적인 데이터를 연속적인 벡터 공간으로 변환하는 기술입니다.

임베딩은 기계 학습 모델 입력으로 사용되며, 성능 향상에 큰 기여를 하고 있습니다. 생성 AI 모델은 문장, 이미지, 코드 등을 생성할 때 임베딩을 활용하고 있습니다.

대표적인 임베딩 기법들은 Word2Vec, GloVe, BERT 등이 있음.



01. 기본 용어 이해 - 임베딩

토큰화

- 텍스트를 작은 단위(토큰)로 나누는 과정
- 각 토큰에 고유 번호(ID)를 부여함
- 예: "나는 학교에 갔어요" → ["나", "는", "학교", "에", "갔", "어요"] → [123, 456, 789, 101, 202, 303]

임베딩

- 토큰화 이후에 일어나는 과정
- 각 토큰 ID를 고차원 벡터(숫자 배열)로 변환
- 예: 토큰 ID 123 → [0.2, -0.5, 0.8, 0.1, ...] (수백 또는 수천 차원의 벡터)

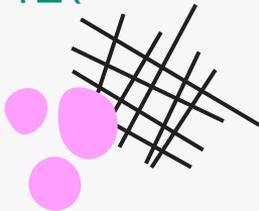
정리한다면

- 토큰화는 텍스트를 쪼개고 번호를 부여하는 과정
- 임베딩은 그 번호를 의미를 담은 벡터로 변환하는 과정

01. 기본 용어 이해 - 전이학습

전이학습은 이미 학습된 모델을 다른 문제에 적용하여 빠르게 성능을 향상시키는 방법입니다.

- 기존 모델의 일부 파라미터를 활용하여 새로운 문제에 맞게 모델을 학습
- 기존 모델이 가진 지식을 재활용할 수 있습니다.
- 이를 통해 데이터와 계산 자원이 부족한 상황에서도 빠르게 모델을 구축할 수 있습니다.
- ChatGPT를 사용하는 것은 전이학습(Transfer Learning)의 한 사례로 볼 수 있습니다.



02. 전이학습의 두 단계

01

대량의 데이터를 모델로 학습시키는 **사전 학습(pre-training)**.
전이학습의 핵심 아이디어로 모델이 사전에 일반적인 지식을 습득하고 이를 특정 문제 해결에 활용합니다.

02

특정한 문제를 해결하기 위한 데이터를 추가 학습시키는 **미세 조정(fine-tuning)**. 이를 통해 기존 모델이 가진 일반적인 지식을 활용하면서도 새로운 문제에 특화된 성능을 낼 수 있습니다.

03. 현재의 ChatGPT까지의 과정

01 RNN

트랜스 포머가 개발되기전 RNN을 활용해서 텍스트를 생성

02 Transformer(2017)

RNN의 한계를 극복하기 위해 등장한 Attention 메커니즘 기반의 Transformer

03 BERT(2018)

Transformer 기반의 사전 학습 언어 모델

04 GPT(2018-2022)

BERT와 유사한 Transformer 기반의 사전 학습 언어 모델

05 ChatGPT(2022)

GPT-3기반의 대화형 AI 모델. 강화학습 기법을 통해 대화 능력이 크게 향상.

06

- Google Gemini
- Anthropic Claude
- OpenAI GPT-4o
- Facebook Llama

04. 기본 용어 이해 - RAG

RAG(Retrieval Augmented Generation)는 대규모 언어 모델에 정보 검색 기능을 결합한 모델 아키텍처.

RAG는 단일 모델이 아닌, 정보 검색(retrieval)과 텍스트 생성을 결합하는 개념적인 접근 방식으로 볼 수 있다.

논문 : Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks(2020년 10월 발표). Facebook AI Research의 연구자 멤버

04. 기본 용어 이해 - RAG

Perplexity AI

Anthropic Claude Pro + Retrieval

- 사용자가 업로드한 문서를 바탕으로 질의응답 가능
- 기업 문서, 연구 자료 등을 분석하고 답변 제공

Genspark

KB 국민카드 - BELLA QNA, 신한투자증권

사우스캐롤라이나 의과대학 - RAG를 활용 MRI 이미지 분석

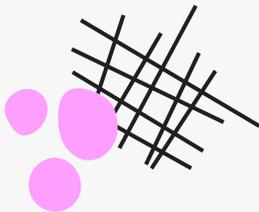
04. 기본 용어 이해 - 트랜스포머(Transformer)

자연어 처리 분야에서 주로 사용되는 딥러닝 모델의 한 종류. 2017년 구글의 연구팀의 "Attention is All You Need"의 논문에서 처음 소개.

01 어텐션 메커니즘 : 입력 데이터의 모든 부분을 동시에 고려 가능.

02 병렬처리 : 전통적인 순환 신경망(RNN)과 달리, 트랜스포머는 입력 시퀀스를 병렬로 처리할 수 있어, 학습 속도가 빠르다.

03 인코더-디코더 구조 : 인코더와 디코더로 구성되어 있어, 인코더는 입력 데이터를 처리(문장을 읽고, 이해)하고, 디코더는 출력데이터를 생성.



05. RAG 논문의 주요 내용

- 모델 구조 : RAG 모델의 설계와 아키텍처, 정보 검색 모듈과 생성 모듈의 상호 작용에 대해 설명.
- 실험 결과 : 다양한 자연어 처리 작업에서 RAG 모델의 성능을 기존의 모델들과 비교하여 설명.
- 응용분야에 대한 설명. RAG 모델이 질문 응답 시스템, 대화 생성과 같은 여러 자연어 처리 태스크에서 어떻게 활용될 수 있는지를 설명

06. RAG 관련 라이브러리

01

Hugging Face Transformers

- 이 라이브러리는 RAG 모델을 지원하며, 사전 학습된 모델과 함께 제공되는 예제 코드를 사용하여 RAG의 개념을 쉽게 구현
- RAG는 검색과 생성 두 가지 요소를 통합하여, 먼저 정보 검색 단계에서 관련 문서를 찾고, 그 후 생성 모델이 이를 기반으로 답변을 생성하는 방식으로 작동

02

Haystack

- Haystack은 RAG뿐만 아니라 다양한 정보 검색 및 QA(Question Answering) 시스템을 지원하는 오픈 소스 프레임워크입니다.
- Haystack은 문서 저장소에서 정보를 검색한 후 이를 생성 모델에 전달하여 응답을 생성할 수 있는 RAG 파이프라인을 제공

07. 기본 용어 이해 - API

01

API는 한 프로그램이 다른 프로그램의 기능을 사용할 수 있도록 도와주는 "다리" 역할.

02

01 기능 제공 : 예를 들어 ChatGPT API를 사용하면 자연어 처리를 위한 기능을 쉽게 사용할 수 있습니다.

02 통합 : 다양한 소프트웨어 시스템이나 애플리케이션을 통합하고 상호작용하도록 합니다. 예를 들어, 결제 시스템 API를 통해 전자상거래 사이트에서 결제 처리를 구현