
- 출처: LangChain 공식 문서 또는 해당 교재명
- 원본 URL: https://smith.langchain.com/hub/teddynote/summary-stuff-documents

---

2. `Cohere Reranker`

- `Cohere` = 기업이 인간-기계 상호작용을 개선할 수 있도록 돕는 자연어 처리 모델을 제공하는 캐나다의 스타트업
- `retriever` 에서 `Cohere` 의 `rerank` `Endpoint` 사용하는 방법

---

1) `기본 설정`

- 사전에 `VS Code` 터미널에 설치할 것

```
pip install -qU cohere
```

- **`Cohere API Key` 설정하기**
  - `API KEY 발급` \
    - `.env` 파일에 **`COHERE_API_KEY`** 값에 저장하기
  - 참고
    - `공식 다큐먼트`
    - `Reranker Model 리스트`

```
# API KEY를 환경변수로 관리하기 위한 설정 파일
from dotenv import load_dotenv

# API KEY 정보로드
load_dotenv()                       # true
```

숨겨진 출력 표시

- `m1 mac` 에서 `Cohere` 가 제대로 작동하지 않아 `google Colab` 으로 옮겨 다시 작업

```
from google.colab import userdata

# API KEY 정보로드
cohere_api_key = userdata.get('COHERE_API_KEY')
```

---

2) `사용법`

```
def pretty_print_docs(docs):
    print(
        f"\n{'-' * 100}\n".join(
            [f"Document {i+1}:\n\n" + d.page_content for i, d in enumerate(docs)]
        )
    )
```

- **`Cohere 다국어 지원 모델`**
  - Embedding: `embed-multilingual-v3.0`, `embed-multilingual-light-v3.0`, `embed-multilingual-v2.0`
  - Reranker: `rerank-multilingual-v3.0`, `rerank-multilingual-v2.0`

```
%pip install langchain-cohere
```

숨겨진 출력 표시

✦

```
%pip install cohere
```

숨겨진 출력 표시

```
%pip install langchain-community
```

숨겨진 출력 표시

```
%pip install faiss-cpu
```

숨겨진 출력 표시

```
from google.colab import drive
drive.mount('/content/drive')
```

숨겨진 출력 표시

```
from langchain_community.document_loaders import TextLoader
from langchain_community.vectorstores import FAISS
from langchain_text_splitters import RecursiveCharacterTextSplitter
from langchain_cohere import CohereEmbeddings

# 문서 로드
# Google Drive 마운트 후 실제 파일 경로를 사용해야 합니다.
# 예시: /content/drive/MyDrive/Your_Folder/appendix-keywords.txt
# TODO: 사용자는 이 경로를 실제 Google Drive 내 파일 경로로 수정
file_path = "/content/drive/MyDrive/Colab Notebooks/Lang_chain/11_Reranker/data/appendix-keywords.txt"
documents = TextLoader(file_path).load()

# 텍스트 분할기 설정
text_splitter = RecursiveCharacterTextSplitter(chunk_size=500, chunk_overlap=100)

# 문서 분할
texts = text_splitter.split_documents(documents)

# 검색기 초기화
retriever = FAISS.from_documents(
    texts, CohereEmbeddings(model="embed-multilingual-v3.0",
                            cohere_api_key=cohere_api_key)
).as_retriever(search_kwargs={"k": 10})

print("🎉 Cohere 임베딩 모델을 사용하여 리트리버가 성공적으로 초기화되었습니다!")
```

```
🎉 Cohere 임베딩 모델을 사용하여 리트리버가 성공적으로 초기화되었습니다!
```

- 🎉 Cohere 임베딩 모델을 사용하여 리트리버가 성공적으로 초기화되었습니다!

```
# 질의문
query = "Word2Vec 에 대해서 알려줘!"

# 문서 검색
docs = retriever.invoke(query)

# 문서 출력
pretty_print_docs(docs)
```

숨겨진 출력 표시

- `Cohere-embedding 모델 결과`

```
Document 1:

Crawling

정의: 크롤링은 자동화된 방식으로 웹 페이지를 방문하여 데이터를 수집하는 과정입니다. 이는 검색 엔진 최적화나 데이터 분석에 자주 사용됩니다.
예시: 구글 검색 엔진이 인터넷 상의 웹사이트를 방문하여 콘텐츠를 수집하고 인덱싱하는 것이 크롤링입니다.
연관키워드: 데이터 수집, 웹 스크래핑, 검색 엔진

Word2Vec

정의: Word2Vec은 단어를 벡터 공간에 매핑하여 단어 간의 의미적 관계를 나타내는 자연어 처리 기술입니다. 이는 단어의 문맥적 유사성을 기반으로 벡터를
예시: Word2Vec 모델에서 "왕"과 "여왕"은 서로 가까운 위치에 벡터로 표현됩니다.
연관키워드: 자연어 처리, 임베딩, 의미론적 유사성
LLM (Large Language Model)
--------------------------------------------------------------------------------
Document 2:
```

Token

정의: 토큰은 텍스트를 더 작은 단위로 분할하는 것을 의미합니다. 이는 일반적으로 단어, 문장, 또는 구절일 수 있습니다.
예시: 문장 "나는 학교에 간다"를 "나는", "학교에", "간다"로 분할합니다.
연관키워드: 토큰화, 자연어 처리, 구문 분석

Tokenizer

정의: 토크나이저는 텍스트 데이터를 토큰으로 분할하는 도구입니다. 이는 자연어 처리에서 데이터를 전처리하는 데 사용됩니다.
예시: "I love programming."이라는 문장을 ["I", "love", "programming", "."]으로 분할합니다.
연관키워드: 토큰화, 자연어 처리, 구문 분석

VectorStore

정의: 벡터스토어는 벡터 형식으로 변환된 데이터를 저장하는 시스템입니다. 이는 검색, 분류 및 기타 데이터 분석 작업에 사용됩니다.
예시: 단어 임베딩 벡터들을 데이터베이스에 저장하여 빠르게 접근할 수 있습니다.
연관키워드: 임베딩, 데이터베이스, 벡터화

SQL

Document 3:

Semantic Search

정의: 의미론적 검색은 사용자의 질의를 단순한 키워드 매칭을 넘어서 그 의미를 파악하여 관련된 결과를 반환하는 검색 방식입니다.
예시: 사용자가 "태양계 행성"이라고 검색하면, "목성", "화성" 등과 같이 관련된 행성에 대한 정보를 반환합니다.
연관키워드: 자연어 처리, 검색 알고리즘, 데이터 마이닝

Embedding

정의: 임베딩은 단어나 문장 같은 텍스트 데이터를 저차원의 연속적인 벡터로 변환하는 과정입니다. 이를 통해 컴퓨터가 텍스트를 이해하고 처리할 수 있게 합
예시: "사과"라는 단어를 [0.65, -0.23, 0.17]과 같은 벡터로 표현합니다.
연관키워드: 자연어 처리, 벡터화, 딥러닝

Token

Document 4:

SQL

정의: SQL(Structured Query Language)은 데이터베이스에서 데이터를 관리하기 위한 프로그래밍 언어입니다. 데이터 조회, 수정, 삽입, 삭제 등 다
예시: SELECT * FROM users WHERE age > 18;은 18세 이상의 사용자 정보를 조회합니다.
연관키워드: 데이터베이스, 쿼리, 데이터 관리

CSV

정의: CSV(Comma-Separated Values)는 데이터를 저장하는 파일 형식으로, 각 데이터 값은 쉼표로 구분됩니다. 표 형태의 데이터를 간단하게 저장하고
예시: 이름, 나이, 직업이라는 헤더를 가진 CSV 파일에는 홍길동, 30, 개발자와 같은 데이터가 포함될 수 있습니다.
연관키워드: 데이터 형식, 파일 처리, 데이터 교환

JSON

Document 5:

Parser

정의: 파서는 주어진 데이터(문자열, 파일 등)를 분석하여 구조화된 형태로 변환하는 도구입니다. 이는 프로그래밍 언어의 구문 분석이나 파일 데이터 처리에
예시: HTML 문서를 구문 분석하여 웹 페이지의 DOM 구조를 생성하는 것은 파싱의 한 예입니다.
연관키워드: 구문 분석, 컴파일러, 데이터 처리

TF-IDF (Term Frequency-Inverse Document Frequency)

정의: TF-IDF는 문서 내에서 단어의 중요도를 평가하는 데 사용되는 통계적 척도입니다. 이는 문서 내 단어의 빈도와 전체 문서 집합에서 그 단어의 희소성을
예시: 많은 문서에서 자주 등장하지 않는 단어는 높은 TF-IDF 값을 가집니다.
연관키워드: 자연어 처리, 정보 검색, 데이터 마이닝

Deep Learning

Document 6:

InstructGPT

정의: InstructGPT는 사용자의 지시에 따라 특정한 작업을 수행하기 위해 최적화된 GPT 모델입니다. 이 모델은 보다 정확하고 관련성 높은 결과를 생성하
예시: 사용자가 "이메일 초안 작성"과 같은 특정 지시를 제공하면, InstructGPT는 관련 내용을 기반으로 이메일을 작성합니다.
연관키워드: 인공지능, 자연어 이해, 명령 기반 처리

Keyword Search

정의: 키워드 검색은 사용자가 입력한 키워드를 기반으로 정보를 찾는 과정입니다. 이는 대부분의 검색 엔진과 데이터베이스 시스템에서 기본적인 검색 방식으로
예시: 사용자가 "커피숍 서울"이라고 검색하면, 관련된 커피숍 목록을 반환합니다.
연관키워드: 검색 엔진, 데이터 검색, 정보 검색

Page Rank

Document 7:

정의: LLM은 대규모의 텍스트 데이터로 훈련된 큰 규모의 언어 모델을 의미합니다. 이러한 모델은 다양한 자연어 이해 및 생성 작업에 사용됩니다.
예시: OpenAI의 GPT 시리즈는 대표적인 대규모 언어 모델입니다.
연관키워드: 자연어 처리, 딥러닝, 텍스트 생성

FAISS (Facebook AI Similarity Search)

정의: FAISS는 페이스북에서 개발한 고속 유사성 검색 라이브러리로, 특히 대규모 벡터 집합에서 유사 벡터를 효과적으로 검색할 수 있도록 설계되었습니다.
예시: 수백만 개의 이미지 벡터 중에서 비슷한 이미지를 빠르게 찾는 데 FAISS가 사용될 수 있습니다.
연관키워드: 벡터 검색, 머신러닝, 데이터베이스 최적화

Open Source

Document 8:

Deep Learning

정의: 딥러닝은 인공신경망을 이용하여 복잡한 문제를 해결하는 머신러닝의 한 분야입니다. 이는 데이터에서 고수준의 표현을 학습하는 데 중점을 둡니다.
예시: 이미지 인식, 음성 인식, 자연어 처리 등에서 딥러닝 모델이 활용됩니다.
연관키워드: 인공신경망, 머신러닝, 데이터 분석

Schema

정의: 스키마는 데이터베이스나 파일의 구조를 정의하는 것으로, 데이터가 어떻게 저장되고 조직되는지에 대한 청사진을 제공합니다.
예시: 관계형 데이터베이스의 테이블 스키마는 열 이름, 데이터 타입, 키 제약 조건 등을 정의합니다.
연관키워드: 데이터베이스, 데이터 모델링, 데이터 관리

DataFrame

Document 9:

JSON

정의: JSON(JavaScript Object Notation)은 경량의 데이터 교환 형식으로, 사람과 기계 모두에게 읽기 쉬운 텍스트를 사용하여 데이터 객체를 표현
예시: {"이름": "홍길동", "나이": 30, "직업": "개발자"}는 JSON 형식의 데이터입니다.
연관키워드: 데이터 교환, 웹 개발, API

Transformer

정의: 트랜스포머는 자연어 처리에서 사용되는 딥러닝 모델의 한 유형으로, 주로 번역, 요약, 텍스트 생성 등에 사용됩니다. 이는 Attention 메커니즘을 기
예시: 구글 번역기는 트랜스포머 모델을 사용하여 다양한 언어 간의 번역을 수행합니다.
연관키워드: 딥러닝, 자연어 처리, Attention

HuggingFace

Document 10:

HuggingFace

정의: HuggingFace는 자연어 처리를 위한 다양한 사전 훈련된 모델과 도구를 제공하는 라이브러리입니다. 이는 연구자와 개발자들이 쉽게 NLP 작업을 수행
예시: HuggingFace의 Transformers 라이브러리를 사용하여 감정 분석, 텍스트 생성 등의 작업을 수행할 수 있습니다.
연관키워드: 자연어 처리, 딥러닝, 라이브러리

Digital Transformation

정의: 디지털 변환은 기술을 활용하여 기업의 서비스, 문화, 운영을 혁신하는 과정입니다. 이는 비즈니스 모델을 개선하고 디지털 기술을 통해 경쟁력을 높이는
예시: 기업이 클라우드 컴퓨팅을 도입하여 데이터 저장과 처리를 혁신하는 것은 디지털 변환의 예입니다.
연관키워드: 혁신, 기술, 비즈니스 모델

```
Crawling
```

3) `CohereReranker`을 사용한 재정렬

- 기본 `retriever`를 `ContextualCompressionRetriever`로 감싸기
- `CohereReranker`(Cohere 재정렬 엔드포인트 사용해 반환된 결과를 재정렬) 추가하기
- *`CohereReranker`에서 모델 이름을 지정하는 것이 필수!*

```
from langchain.retrievers.contextual_compression import ContextualCompressionRetriever
from langchain_cohere import CohereRerank

# 문서 재정렬 모델 설정
compressor = CohereRerank(
    model="rerank-multilingual-v3.0",
    cohere_api_key=cohere_api_key
    )

# 문맥 압축 검색기 설정
compression_retriever = ContextualCompressionRetriever(
    base_compressor=compressor, base_retriever=retriever
)

# 압축된 문서 검색
compressed_docs = compression_retriever.invoke("Word2Vec 에 대해서 알려줘!")

# 압축된 문서 출력
pretty_print_docs(compressed_docs)
```

숨겨진 출력 표시

- `CohereReranker` 사용한 결과

```
Document 1:

Crawling

정의: 크롤링은 자동화된 방식으로 웹 페이지를 방문하여 데이터를 수집하는 과정입니다. 이는 검색 엔진 최적화나 데이터 분석에 자주 사용됩니다.
예시: 구글 검색 엔진이 인터넷 상의 웹사이트를 방문하여 콘텐츠를 수집하고 인덱싱하는 것이 크롤링입니다.
연관키워드: 데이터 수집, 웹 스크래핑, 검색 엔진

Word2Vec

정의: Word2Vec은 단어를 벡터 공간에 매핑하여 단어 간의 의미적 관계를 나타내는 자연어 처리 기술입니다. 이는 단어의 문맥적 유사성을 기반으로 벡터를
예시: Word2Vec 모델에서 "왕"과 "여왕"은 서로 가까운 위치에 벡터로 표현됩니다.
연관키워드: 자연어 처리, 임베딩, 의미론적 유사성
LLM (Large Language Model)
--------------------------------------------------------------------------------
Document 2:

Token

정의: 토큰은 텍스트를 더 작은 단위로 분할하는 것을 의미합니다. 이는 일반적으로 단어, 문장, 또는 구절일 수 있습니다.
예시: 문장 "나는 학교에 간다"를 "나는", "학교에", "간다"로 분할합니다.
연관키워드: 토큰화, 자연어 처리, 구문 분석

Tokenizer

정의: 토크나이저는 텍스트 데이터를 토큰으로 분할하는 도구입니다. 이는 자연어 처리에서 데이터를 전처리하는 데 사용됩니다.
예시: "I love programming."이라는 문장을 ["I", "love", "programming", "."]으로 분할합니다.
연관키워드: 토큰화, 자연어 처리, 구문 분석

VectorStore

정의: 벡터스토어는 벡터 형식으로 변환된 데이터를 저장하는 시스템입니다. 이는 검색, 분류 및 기타 데이터 분석 작업에 사용됩니다.
예시: 단어 임베딩 벡터들을 데이터베이스에 저장하여 빠르게 접근할 수 있습니다.
연관키워드: 임베딩, 데이터베이스, 벡터화

SQL
--------------------------------------------------------------------------------
Document 3:
```

정의: LLM은 대규모의 텍스트 데이터로 훈련된 큰 규모의 언어 모델을 의미합니다. 이러한 모델은 다양한 자연어 이해 및 생성 작업에 사용됩니다.
예시: OpenAI의 GPT 시리즈는 대표적인 대규모 언어 모델입니다.
연관키워드: 자연어 처리, 딥러닝, 텍스트 생성

FAISS (Facebook AI Similarity Search)

정의: FAISS는 페이스북에서 개발한 고속 유사성 검색 라이브러리로, 특히 대규모 벡터 집합에서 유사 벡터를 효과적으로 검색할 수 있도록 설계되었습니다.
예시: 수백만 개의 이미지 벡터 중에서 비슷한 이미지를 빠르게 찾는 데 FAISS가 사용될 수 있습니다.
연관키워드: 벡터 검색, 머신러닝, 데이터베이스 최적화

Open Source

- *next:* ***`03. Jina Reranker`***
