


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


07_텍스트 분할(Text Splitter)

개념 설명

- [텍스트 분할기](#)
- [LangChain TextSplitter](#)
- [Chunk 분할 시각화 사이트](#)
 - Greg Kamradt 가 만든 [Chunk Visualization](#) 사이트
 -  Chunk 분할 시각화 사이트

- **RAG** (Retrieval-Augmented Generation) 시스템의 두 번째 단계
- 로드된 문서들을 **효율적으로 처리** 하고, 시스템이 정보를 보다 **잘 활용** 할 수 있도록 준비하는 중요한 과정

목적


- 크고 복잡한 문서 → **LLM** 이 받아들일 수 있는 **효율적인 작은 규모의 조각** 으로 나누는 작업
- 나중에 사용자가 입력한 질문에 대하여 **보다 효율적인 정보만 압축** • **선별** 하여 가져오기 위함
 - 예시: 구글이 엔트로픽에 투자한 금액은 얼마야?
 -  예시 화면

분할의 필요성

- **정확성**: 핀포인트 정보 검색
 - 문서 세분화 함 → **질문(Query)** 에 **연관성** 이 있는 **정보만** 가져오는데 도움
 - **각각의 단위** 는 **특정 주제 나 내용** 에 초점을 맞춤 → **관련성이 높은 정보** 제공
- **효율성**: 리소스 최적화
 - 전체 문서를 **LLM** 으로 입력시 단점 → 비용이 많이 발생
 - **효율적인 답변** 을 많은 정보 속에 발췌하여 **답변하지 못함**
 - 때로는 이러한 문제가 **할루시네이션** 으로 이어짐
 - 따라서, **답변에 필요한 정보만 발췌하기 위한 목적** 도 있음

문서분할 과정

- 문서 구조 파악

- PDF 파일, 웹 페이지, 전자 책 등 다양한 형식의 문서 → 구조 파악
- 문서의 헤더, 푸터, 페이지 번호, 섹션 제목 등을 식별 하는 과정 포함
- 단위 선정
 - 문서를 어떤 단위 로 나눌지 결정 → 문서의 내용 과 목적 에 따라 달라짐
 - 페이지별, 섹션별, 또는 문단별 등
- 단위 크기 선정 (chunk size)
 - 문서를 몇 개의 토큰 단위 로 나눌 것인지를 결정
- 청크 오버랩 (chunk overlap)
 - 분할된 끝 부분 에서 맥락이 이어질 수 있도록 일부를 겹쳐서(overlap) 분할하는 것이 일반적
 -  청크 크기 & 청크 오버랩

▼

코드

- 사전 VS Code 터미널에 설치할 것

```
pip install -qU langchain-text-splitters
```

```
from langchain_text_splitters import RecursiveCharacterTextSplitter

# 단계 2: 문서 분할(Split Documents)
text_splitter = RecursiveCharacterTextSplitter(
    chunk_size=1000,          # 분할된 텍스트 청크 최대 크기 지정
    chunk_overlap=50         # 분할된 텍스트 청크 간 중복되는 문자 수 지정
)

# 중간 과정 중략

# 단계 3: 문서 분할
splits = text_splitter.split_documents(docs)
```

- next: **문자 텍스트 분할 (CharacterTextSplitter)**