**Lecture Summary Bot Local**

딥러닝 60232062 조온유

**프로젝트 개요**

본 프로젝트는 강의노트 파일(.txt, .md)을 입력으로 받아, 완전히 로컬 환경에서 규칙 기반으로 핵심 내용을 요약하는 자동화 프로그램이다.

외부 API(GPT, Gemini 등)를 사용하지 않고, 텍스트 처리 기법(문장 분할, TF-IDF 기반 키워드 추출, 문서 청킹)을 통해 요약을 생성한다.

주요 특징

• 외부 API나 네트워크 연결 불필요 (완전 로컬 동작)

• 다양한 요약 타입 제공 (키워드 요약 / 문장 요약 / 혼합 요약)

• 긴 문서 자동 분할 및 처리

• 요약 결과를 Markdown 형식으로 저장

• 한국어 및 영어 텍스트 지원

• 재현 가능한 결과 제공 (랜덤 시드 기반)

**사용한 입력 텍스트**

본 프로젝트에서는 notes/ 폴더에 강의노트를 넣고 요약을 생성하였다.

예시 입력 파일:  
- notes/sample.md  
- notes/algorithm\_basics.txt

**실행 방법**

기본 실행:  
python summary\_gen.py --notes notes --out summaries

다양한 옵션:  
- 키워드 중심 요약: python summary\_gen.py --notes notes --out summaries --type keywords  
- 문장 중심 요약: python summary\_gen.py --notes notes --out summaries --type sentences  
- 혼합 요약: python summary\_gen.py --notes notes --out summaries --type mixed

실행결과는 스크린샷으로 첨부

**결론**

Lecture Summary Bot Local 프로젝트를 통해 외부 LLM 없이도 규칙 기반 접근으로 강의노트 요약을 자동 생성할 수 있음을 확인하였다.

다만, 규칙 기반 요약은 문맥 이해가 부족하여 의미적으로 부정확하거나 단순한 결과가 나올 수 있다는 한계가 있다.

향후 확장 아이디어로는 다양한 언어 지원, 요약 템플릿 커스터마이징, 통계 기반 대시보드 제공 등이 있다.

입력한 text:

====================

외부 LLM이나 네트워크를 전혀 사용하지 않는 “규칙 기반 퀴즈 생성기”를 Python으로 구현해줘. 프로젝트명은 lecture-quiz-bot-local.

## 목표

- 강의노트(.txt, .md)에서 규칙 기반으로 핵심 개념을 추출하고, Cloze/MCQ/T-F 형태의 문제를 자동 생성해 Markdown으로 저장.

- 재현성 보장(seed), 간단한 품질 보정(중복 제거/정답 유효성), 긴 문서 분할 처리.

## 기능 요구

1) 입력 스캔/전처리

- `notes/` 스캔, 확장자 필터(.txt, .md)

- 10k자 초과 시 문단/빈 줄 기준으로 6~8k자 단위 청크

- 문장 분할: 정규식 기반(., ?, !, 줄바꿈)

2) 키워드/개념 추출

- scikit-learn TfidfVectorizer(ngram 1~2)로 상위 토큰/바이그램 후보

- 간단 stopword 리스트(ko/en 혼합) 제공, 숫자/특수문자 제거, 길이≥2

3) 문제 유형

- Cloze: 상위 키워드로 마스킹, 정답/근거 1줄

- MCQ(4지선다): 정답=키워드, 오답=동일 문서 상위 유사 키워드에서 샘플링(중복·동의어 제거)

- T/F: 원문 True, 부정/치환으로 False 생성 + 근거

- 스타일: mcq|short|mix, 파일당 `--per-file` 개(부족하면 다른 유형으로 보충)

4) 출력

- quizzes/{basename}\_quiz.md

- 마크다운 헤더, 번호, 보기 A–D, \*\*정답\*\*, 해설, 마지막 “## Key Concepts”

5) CLI

- `python quiz\_gen.py --notes notes --out quizzes --per-file 8 --style mix --ext ".txt,.md" --lang ko --seed 42 --verbose`

6) 품질/안정성

- 유효성 검사: 각 문항에 정답 존재, 보기는 4개, 중복 제거, 빈 문자열 제거

- 로그: 처리 파일 수, 추출 키워드 상위 N, 생성 실패 시 경고

- 랜덤성: 난수 seed 고정

7) 테스트/문서

- tests/test\_rules.py: 문장 분할, 키워드 추출, MCQ 보기 유효성, cloze 생성 테스트

- README: 설치(pip), 실행 예시, 한계(규칙 기반이라 의미 해석 제한), 확장 아이디어

- Makefile: `make run`, `make test`

## 파일 구조

- quiz\_gen.py

- rules.py

- io\_utils.py

- validate.py

- tests/test\_rules.py

- requirements.txt (scikit-learn)

- README.md

- Makefile

- notes/sample.md

- .gitignore

## 구현 팁

- 한국어/영문 혼합 텍스트 처리 대비해 단순 토크나이즈(공백/문장부호 기준)와 n-gram 사용.

- T/F 생성 시 False 케이스는 숫자 ±1, 빈도 상위 키워드 교체 등 단순 규칙으로 만들되 해설에 근거 문장 출처를 남김.

- 최종 마크다운은 엄격히 포맷팅(번호, 보기, \*\*정답:\*\*, \*\*해설:\*\*)을 지켜 파싱 가능하게.

모든 코드에 한국어 주석과 타입 힌트를 달고, 샘플 노트에 대해 실제 생성된 샘플 퀴즈 파일을 만들어서 함께 포함해줘.

여기서 다 만들고 주제를 바꿔서 퀴즈 생성이 아니라 요약 생성으로 바꿔달라 해서 지금의 결과가 나왔습니다.