

Bitamin 12th Final Conference

Auto ReadMe Generator

프로젝트 발표자료를 기반으로 자동으로 **readme.md**를 만들어주는 장치

NAME OF PROJECT:

Auto ReadMe Generator

PRESENTED BY:

Project Maker조
12기 권도영, 김지원, 송규현



Table of Contents

| | |
|----|----------------------------|
| 01 | Introduction |
| 02 | [task1] PDF Processing |
| 03 | [task2] Image Classifier |
| 04 | [task3] Text summarization |
| 05 | Implementation |
| 06 | Conclusions |



01

Introduction

주제선정배경 | 서비스 플로우

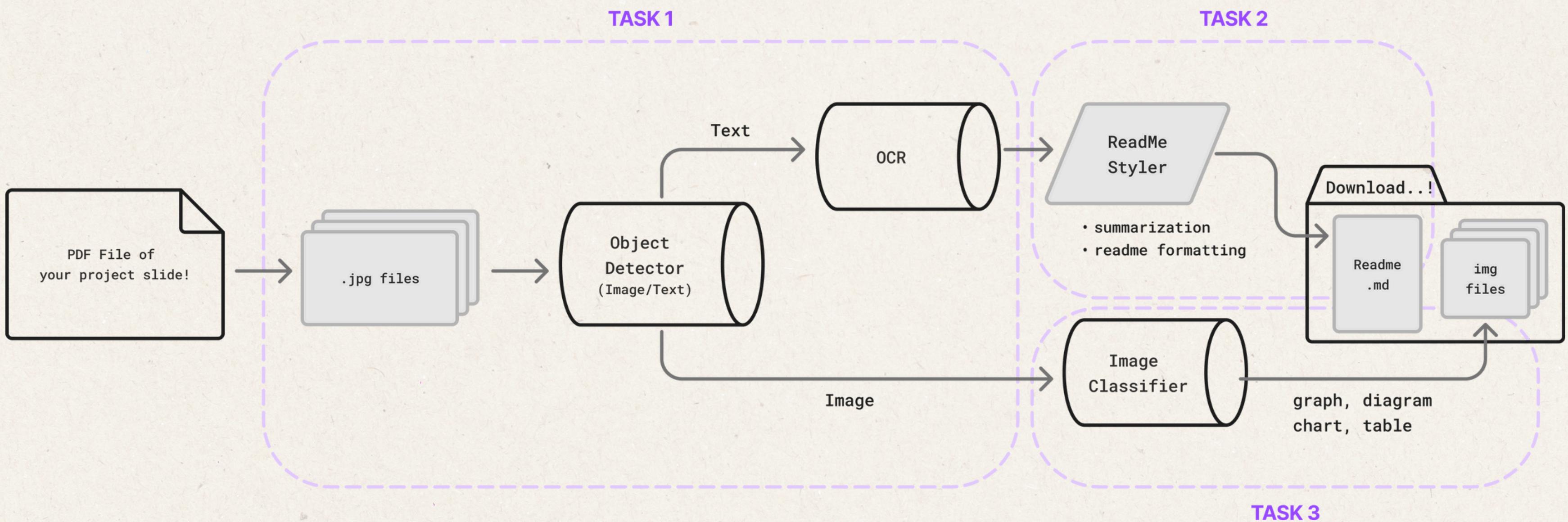
주제 선정 동기

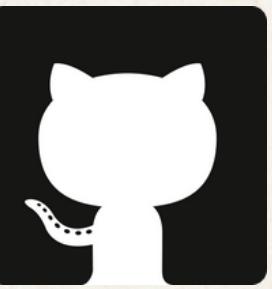
- 프로젝트를 진행하면서 깃헙 레포지토리를 만들 때마다 README 파일 작성에 상당한 시간이 소요됨
- 일반적으로 학회, 학술동아리, 연구실 등에서 발표하는 사용자는 이미 발표용 PPT를 가지고 있으며, PPT에는 프로젝트의 주요 내용이 포함되어 있음
- 기존의 README 자동 생성 프로그램들은 코드에 초점을 맞춰 프로젝트 내용은 요약할 수 없다는 한계가 있음
- PPT 제작 이후 README 파일을 또다시 작성해야 하는 번거로움을 해결하기 위해 발표용 PPT를 활용하여 빠르게 README 파일을 자동 생성해주는 프로그램의 필요성을 느낌

유사 서비스 비교 분석

| 이름 | readme | readme-ai | REGEN | EASYME | Mapify | ProjectMaker |
|---------|------------------------|--|-------------------------------------|---------------------------|------------------|---|
| 타겟 사용자 | 개발자 | 개발자 | 개발자 | 개발자 | 모든 사용자 | AI연구원, AI개발자 |
| AI 사용여부 | X | LLM | LLM | X | 생성형AI | YOLOv8, VGG19, LLM(GPT-4o-mini) |
| 인터페이스 | CLI | CLI | CLI | GUI | GUI | GUI |
| 입력 | readme파일 | github repository | github repository | 직접 입력 | PDF, 웹사이트, 이미지 등 | PDF |
| 출력 | 템플릿/커스텀 템플릿의 readme 파일 | code로부터 directory structure, 사용하는 modules 등을 파악하여 readme 파일 생성 | repo의 파일들을 읽고 LLM이 요약한 readme 파일 생성 | markdown preview 및 편집기 제공 | 주요 내용을 요약한 마인드맵 | pdf 파일에서 요약한 주요 내용과 추출한 주요 이미지, markdown preview 및 편집기 제공 |

Flowchart



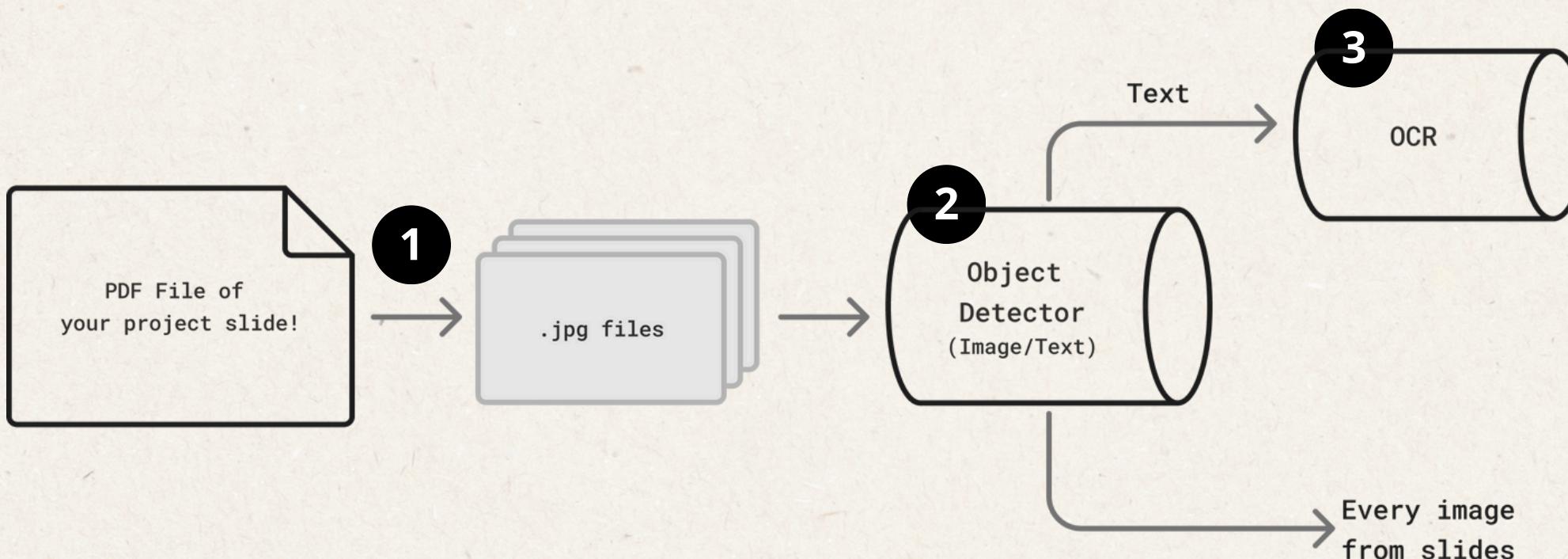


02

[Task1]

PDF Processing

개요



01 **pdf2jpg** 파일을 이용하여 pdf 파일을 N장의 jpg 파일들로 변환

02 **Object Detector**로 슬라이드 내의 모든 이미지와 텍스트를 검출, 서버에 저장

03 텍스트들은 **OCR**로 처리하여 서버에 .txt파일로 저장, 이미지들은 다음 단계로 넘김

개요

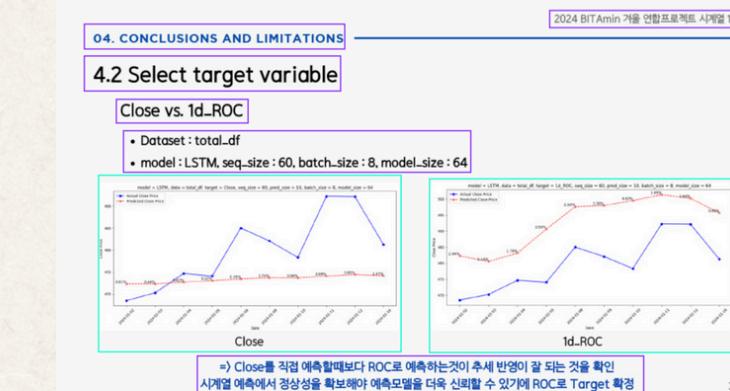
01 목표 : 슬라이드의 모든 문자 정보들을 추출하여 이를 바탕으로 readme(요약본)를 생성

- image상 문자를 encoded text로 변환하는 방법으로 OCR (Optical Character Recognition) : 광학문자기술을 이용

| 접근방식 1 | 접근방식 2 |
|--|--|
| 슬라이드 전체를 OCR 모델의 input으로 입력 | 각 슬라이드 별로 문자가 적혀 있는 부분을 탐지, 해당 부분을 crop하여 OCR모델의 input으로 입력 |
| 한계: 표, 다이어그램 속 문자까지 모두 추출되어 텍스트가 정돈되지 않음 | <접근방식1> 의 한계를 개선, 직접적으로 readme에 들어갈 내용과 관련된 문자들만 텍스트로 정리할 수 있게 됨 |

02 접근방식2 구현 :

- 슬라이드 별 image / text를 별도 label로 탐지할 수 있는 Object Detection Model (finetuned)
- 탐지된 text (이미지 형태)를 encoded text로 변환할 수 있는 OCR



모델 평가 및 선정

01 Object Detection Model

- : 처음에는 5 class (text, chart, diagram, table, unknown img)로 탐지하도록 훈련시켰으나 ‘text’ class에 대한 False Positive가 높아 2 class (text, image)로 탐지하도록 방향성 수정
- : RoboFlow(AutoTrain), YOLOv8, DETR 3개의 backbone model에 대하여 파인튜닝,
Bitamin 12기가 겨울방학/학기 프로젝트에 활용했던 모든 슬라이드를 훈련 데이터로 활용 (총 551장)

| (Avg) | RoboFlow | YOLOv8 (best) | DETR (best) |
|-----------|--------------|----------------------|-------------|
| Precision | 74.9% | 77.3% | 52.6% |
| Recall | 81.5% | 77.3% | 72.7% |
| mAP@50 | 79.2% | 79.9% | 76.2% |
| mAP@50-95 | 60.0% | 63.5% | 52.6% |

02 OCR Model

- : 여러가지 모델로 비교 실험, 지표로 소요시간, CER, WER 사용

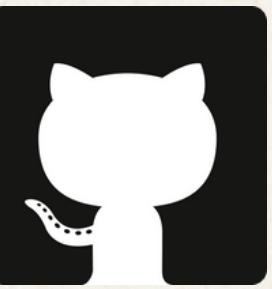
Character Error Rate (CER):

$$CER = \frac{\text{Number of incorrect characters}}{\text{Total number of characters in the reference text}} \times 100\%$$

Word Error Rate (WER):

$$WER = \frac{\text{Number of incorrect words}}{\text{Total number of words in the reference text}} \times 100\%$$

| (Avg) | tesseract | easy-ocr | paddle-ocr |
|----------|------------|----------|-------------------|
| Time (s) | 151 | 160 | 482 |
| WER | 0.2563 | 0.3046 | 0.5531 |
| CER | 0.1854 | 0.0625 | 0.2906 |



03

[Task2]

Image Classifier

사용 데이터 및 전처리

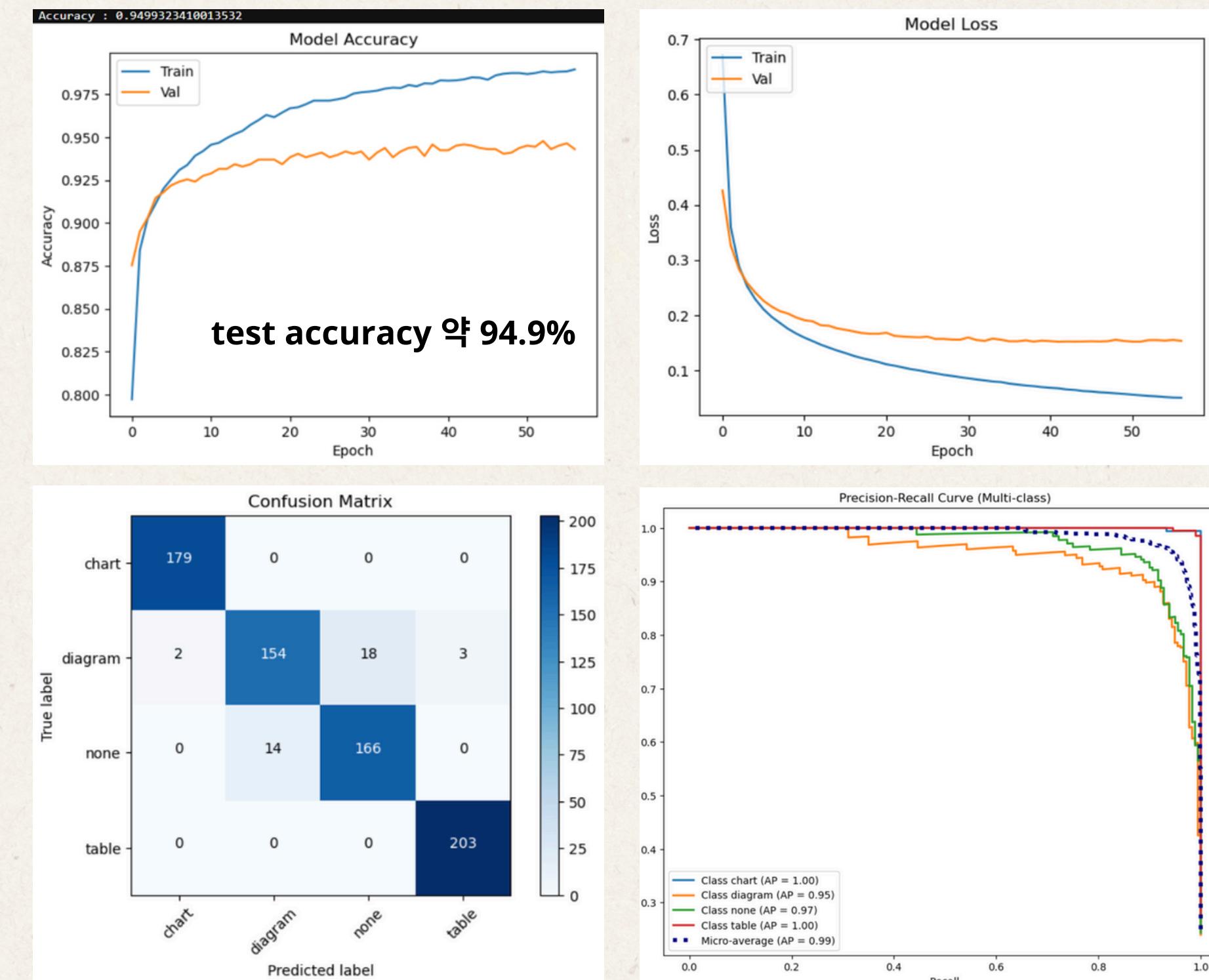
- 목표 : 4개의 class로 이미지를 분류, web에 프로젝트 내용과 유관한 이미지만 전송하기 위함

| class | class 1 : chart | class 2 : diagram | class 3 : table | class 4 : None |
|-------------|--|--|---|---|
| data source | 1. 일반 차트 이미지 2. 복잡한 구조의 line plot, bar plot 3. 무작위 생성 confusion matix, heatmap, boxplot, violin plot | 1. <i>A Diagram Is Worth A Dozen Images</i> 논문 데이터셋 일부 2. 인공지능 분야 논문의 Model architecture, use-case, flow, ERD 등의 구글링 이미지 캡쳐 | 1. <i>TNCR: Table Net Detection and Classification Dastaset</i> 논문 데이터셋 일부 2. <i>Image-based table recognition: data, model, and evaluation</i> 논문 데이터셋 일부 | 1. <i>A Diagram Is Worth A Dozen Images</i> 논문 데이터셋 일부 (class 2 : diagram 으로 분류하지 않는 종류의 이미지 선택) |
| # of imgs | 1901장 | 1815장 | 1867장 | 1800장 |
| example | | | | |

- 전처리 : (244,244) 사이즈로 resize & normalization
 - 별도의 augmentation이 성능 향상에 도움이 되지 않아 미적용

모델 훈련

- pre-trained vgg19 image classifier 모델을 backbone으로 활용
- 마지막 fc layer에서 class수만 변경 후 fine-tuning 진행
- 훈련 조건
 - criterion : CrossEntropy
 - optimizer : Adam
 - train epoch : 300
 - batch size : 20
 - earlystopping : used (patience = 15)
 - gpu : used
 - 총 훈련 시간 : 약 1시간
- ImageNet (1000 class) : 현재 OmniVec2 - 93.6%가 최고 정확도
 CIFAR-10 (10 class) : 현재 최고 정확도가 efficient adaptive ensembling - 99.6%
 (ref: [paperswithcode](#)[Papers with Code - Image Classification](#))
- 여러 비타민 프로젝트 pdf를 test한 결과 새로운 유형의 이미지, 특히 diagram을 잘 분류하지 못함. 따라서 데이터셋의 다양성 및 개수를 보강할 필요가 있음.





04

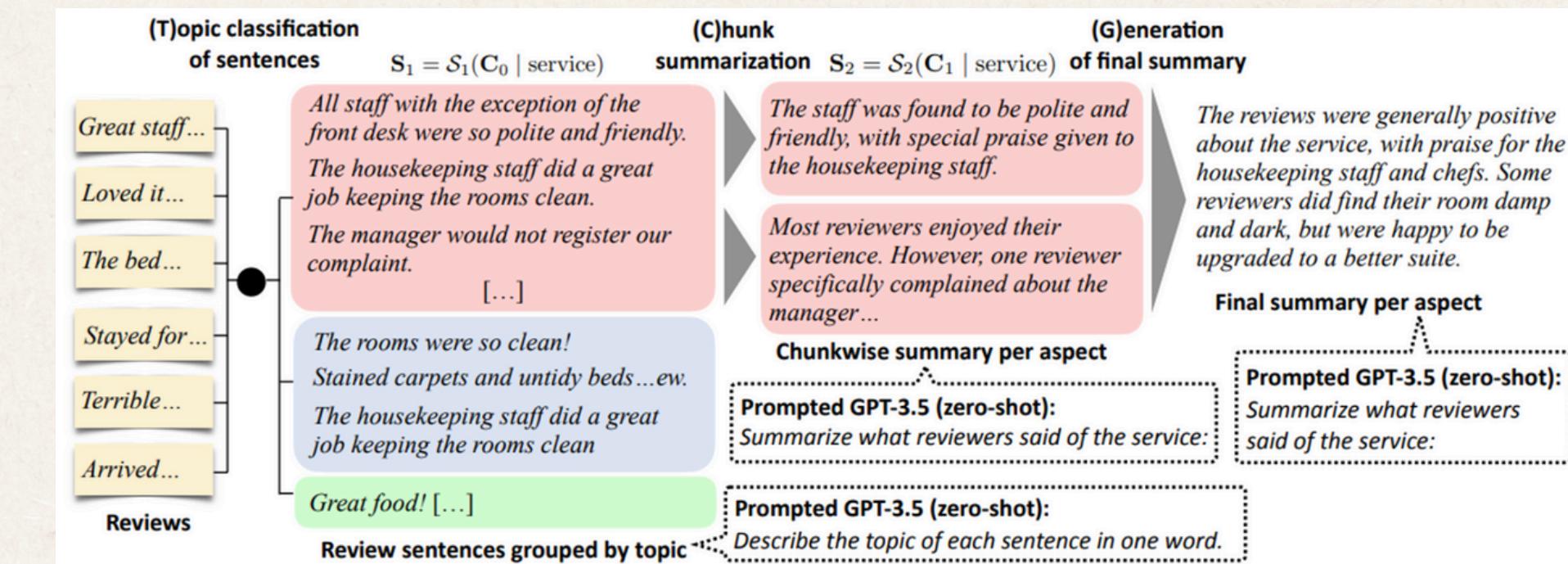
[Task3]

Text Summarization

텍스트 요약 방법 관련선행연구 조사

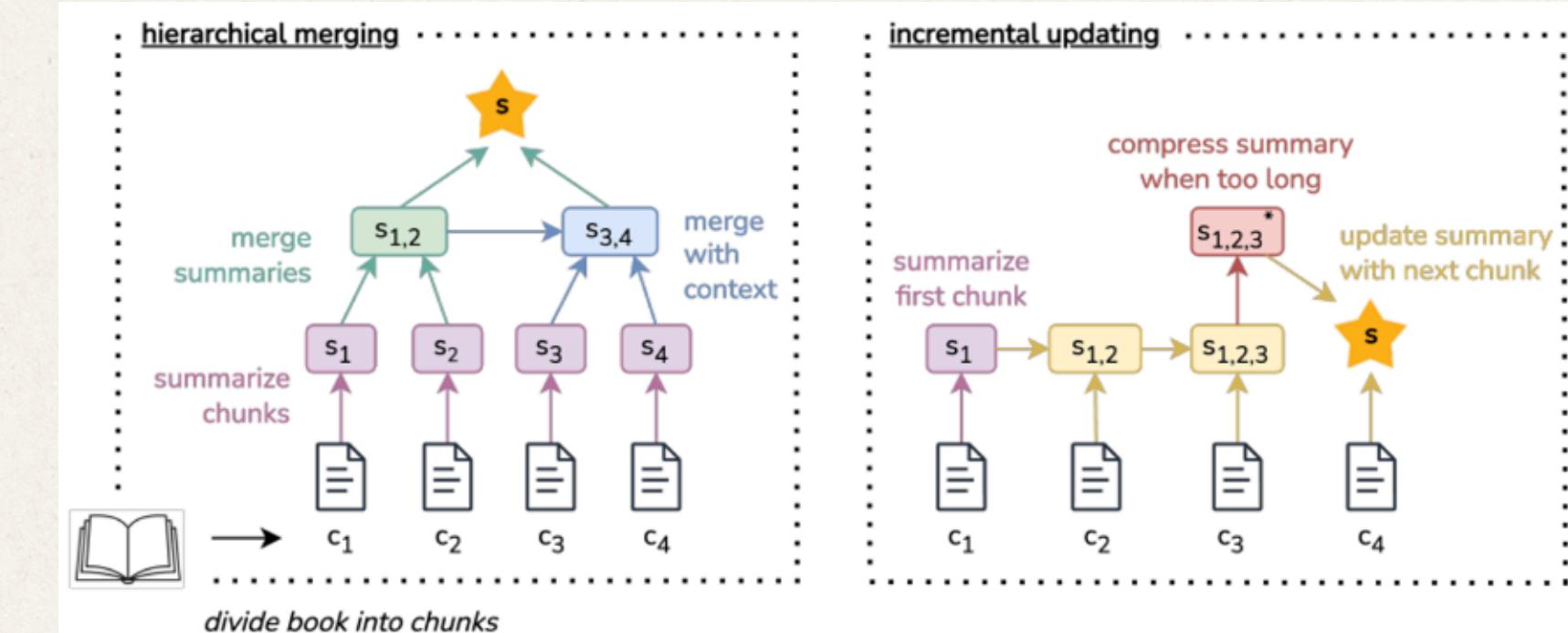
01 Bhaskar et al. (2022). Zero-shot opinion summarization with GPT-3

- 주제 클러스터링-청킹-생성 파이프라인을 통한 계층적 요약과 추출적 요약 모델을 이용한 사전 추출 방법을 소개함
- 사전 추출 단계는 입력 데이터의 길이가 길어질 때 GPT-3.5가 생성한 요약문의 사실성과 신뢰성이 떨어지는 문제를 해결할 수 있음



02 Chang et al. (2024). BoookScore: Book-length summarization with LLMs

- GPT-3가 강화 학습을 통해 각 청크를 요약하도록 미세 조정한 후, 청크 수준의 요약들을 계층적으로 병합
- 인간이 평가한 결과 8개의 평가항목 중 불일치성을 제외한 모든 항목에서 hierarchical merging 방식의 성능이 더 높게 평가됨

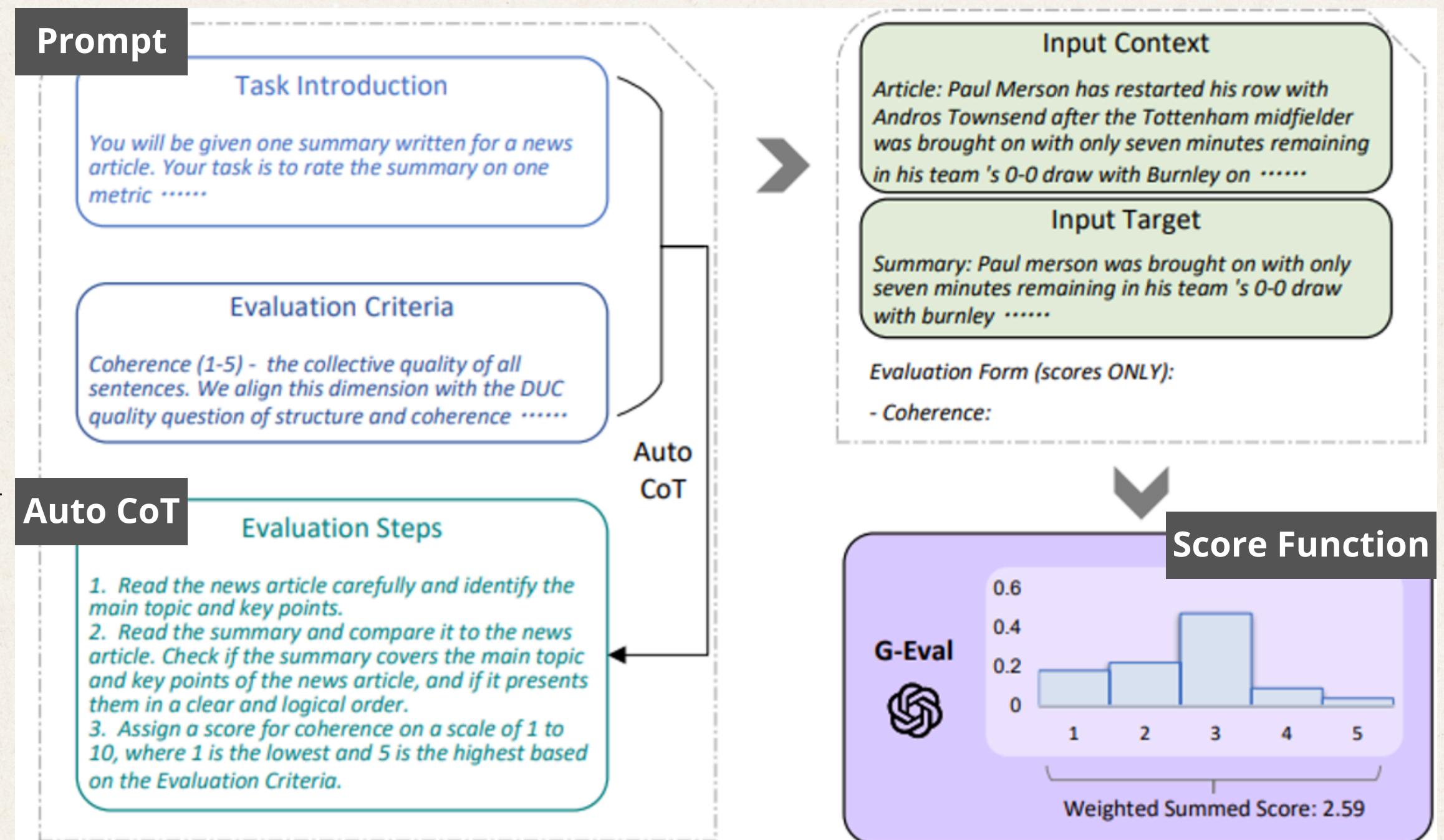


텍스트 요약 방법 비교

| | method1 | method2 | method3 | method4 | method5 |
|-----------|---|--|--|---|--|
| 특징 | <p>단계별 정보 추출 및 요약 추출적 요약 방법을 활용하여 주요 정보를 사전에 필터링 함</p> | <p>클래스 분류를 활용한 요약 전체 텍스트가 사전에 지정한 8개의 클래스에 따라 분류되도록 함</p> | <p>대주제 기반 텍스트 분할 요약 사전에 추출한 대주제를 바탕으로 전체 텍스트를 분할한 후, 각 파트별 요약</p> | <p>계층적 병합 요약 전체 목차를 4개로 분할한 후, 해당되는 텍스트 추출 및 요약 진행 후 전체 병합</p> | <p>클러스터링을 활용한 요약 목차를 기준으로 클러스터 수를 설정하여 클러스터링 한 후 각각 요약 진행</p> |
| 장점 | 사전에 추출한 정보를 활용하여 대주제와 관련된 텍스트 위주로 요약 가능 | 요약 과정이 비교적 간단하고 간결하여 소요 시간을 단축할 수 있음 | 텍스트 분할 과정을 통해 요약 과정에서의 텍스트 누락을 최소화할 수 있음 | 각 대주제 리스트의 특징에 따라 개별적으로 요약 프롬프트 작성 가능 | 요약에 pdf의 모든 페이지 정보가 균일하게 포함됨 |
| 단점 | 사전 정보 추출 과정에서 대주제가 누락될 경우, 관련 내용이 누락될 수 있음 | 분류 과정에서 일부 PPT 슬라이드에 해당되는 텍스트 전체가 누락되는 문제가 발생 | 사전 정보 추출 과정에서 대주제가 누락될 경우 생성된 요약본의 계층적 구조에 오류가 발생할 수 있음 | 텍스트 추출 과정에서 대주제 리스트와 관련 없는 내용을 추출하는 오류가 다수 발생함 | 클러스터링에 많은 시간 소요하고 요약문의 길이가 상대적으로 깊. 목차와 클러스터링이 매칭되지 않음 |

G-EVAL을 활용한 요약 평가: G-EVAL 아키텍쳐

평가하려는 태스크에 대한
지시사항과 추구하는 평가 기준을
정의



LLM이 자동으로 생성한 평가 단계를
제공하며, 제시된 프롬프트를
수행하기 위한 자세한 평가 지침을
생성

프롬프트, Auto-CoT, 입력 텍스트,
평가할 텍스트를 한 번에 전달받아
요약의 품질을 평가하는 함수
(GPT-4o-mini는 output 토큰에 대한
확률값을 제공하지 않아 20번 샘플링
한 점수의 가중평균으로 대체함)

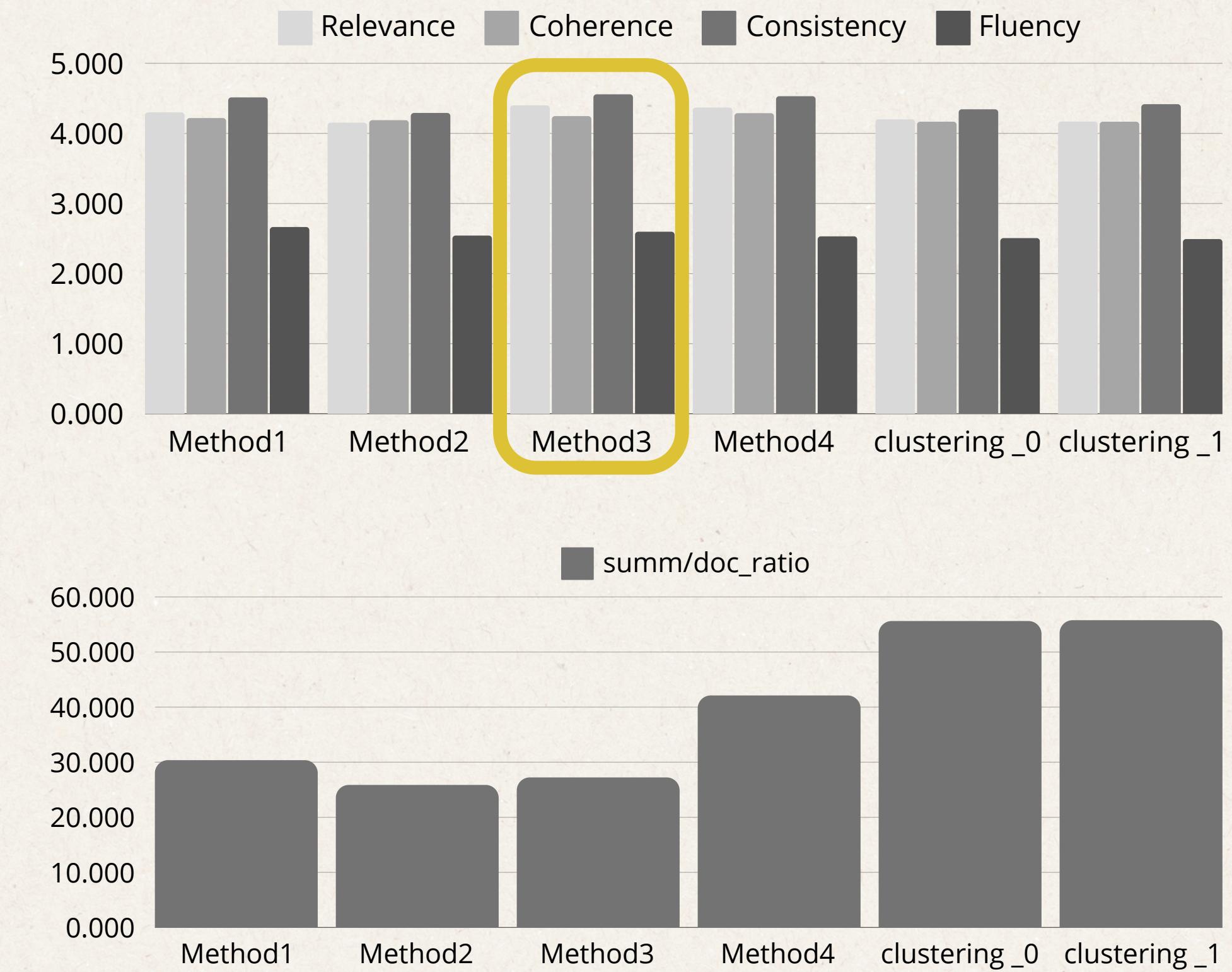
G-EVAL을 활용한 요약 평가: G-EVAL 프롬프트 설정

- OpenAI의 SummEval 벤치마크 평가 기준과 Auto-CoT을 기반으로, 해당 프로젝트에 적합하게 평가 기준을 조정
- 생성된 요약본이 태그 형식으로 정리되어 있기 때문에 이 구조를 파악할 수 있도록 CoT를 수정하였고, 프롬프트에도 태그 구조 관련 정보를 추가함

| 평가 기준 | 평가 내용 | 점수 범위 |
|--------------------|--------------------------------|-------|
| Relevance | 요약이 원본 문서의 핵심 정보를 잘 전달하는가? | 1 ~ 5 |
| Coherence | 요약이 전체적인 논리적 흐름과 구조를 갖고 있는가? | 1 ~ 5 |
| Consistency | 요약이 원본 문서의 사실, 정보를 정확하게 반영하는가? | 1 ~ 5 |
| Fluency | 문장이 자연스럽고 문법적으로 올바른가 ? | 1 ~ 3 |

G-EVAL 평가 결과 및 텍스트 요약 방법 선택

| | Relevance | Coherence | Consistency | Fluency | summ/ doc_ratio |
|----------------|--------------|--------------|--------------|--------------|--------------------|
| Method1 | 4.300 | 4.219 | 4.514 | 2.664 | 30.367 |
| Method2 | 4.153 | 4.189 | 4.292 | 2.542 | 25.873 |
| Method3 | 4.400 | 4.247 | 4.558 | 2.597 | 27.236 |
| Method4 | 4.369 | 4.289 | 4.531 | 2.531 | 42.116 |
| clustering _0 | 4.200 | 4.167 | 4.344 | 2.506 | 55.625 |
| clustering _1 | 4.169 | 4.167 | 4.417 | 2.492 | 55.777 |



README Formatting

- 요약된 .txt 파일을 마크다운 형식으로 자동 변환하여, 사전 정의된 <subject>, <team>, <index> 태그에 따라 프로젝트명, 팀원명, 목차를 정리함
- <main>, <sub>, <content> 태그를 기반으로 헤딩과 불렛포인트로 내용이 입력되게 하여 README의 기본 뼈대를 자동생성함

The screenshot shows a text editor interface with a toolbar at the top and a code editor window below. The code editor displays a .txt file with the following content:

```
# 🍊 안저 데이터를 이용한 질병 분류
📅 (프로젝트 진행기간을 입력하세요. #####.##.## ~ #####.##.##)
### 🚀 Team
권도영, 권민지, 김서윤, 김채원, 박서진

## 📄 Table of Contents
- [1주제 배경 및] (#section_1)
- [2데이터 소개 및 EDA] (#section_2)
- [3전처리] (#section_3)
- [4모델링] (#section_4)
- [5결과] (#section_5)
<br>
<a name='section_1'></a>

## 🔳 주제 배경 및
#### 프로젝트 배경
- 안저검사 망막 사진을 이용하여 자폐스펙트럼장애를 예측하는 모델이 개발되었으며, 이는 진단 및 예후 예측 시스템 구축에 활용될 수 있다.

<br>
<a name='section_2'></a>

## 🔳 데이터 소개 및 EDA
#### 데이터 소개
```

To the right of the code editor, the generated HTML output is shown:

안저 데이터를 이용한 질병 분류

📅 (프로젝트 진행기간을 입력하세요. #####.##.## ~ #####.##.##)

Team

권도영, 권민지, 김서윤, 김채원, 박서진

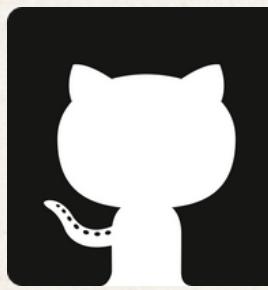
Table of Contents

- 1주제 배경 및
- 2데이터 소개 및 EDA
- 3전처리
- 4모델링
- 5결과

주제 배경 및

프로젝트 배경

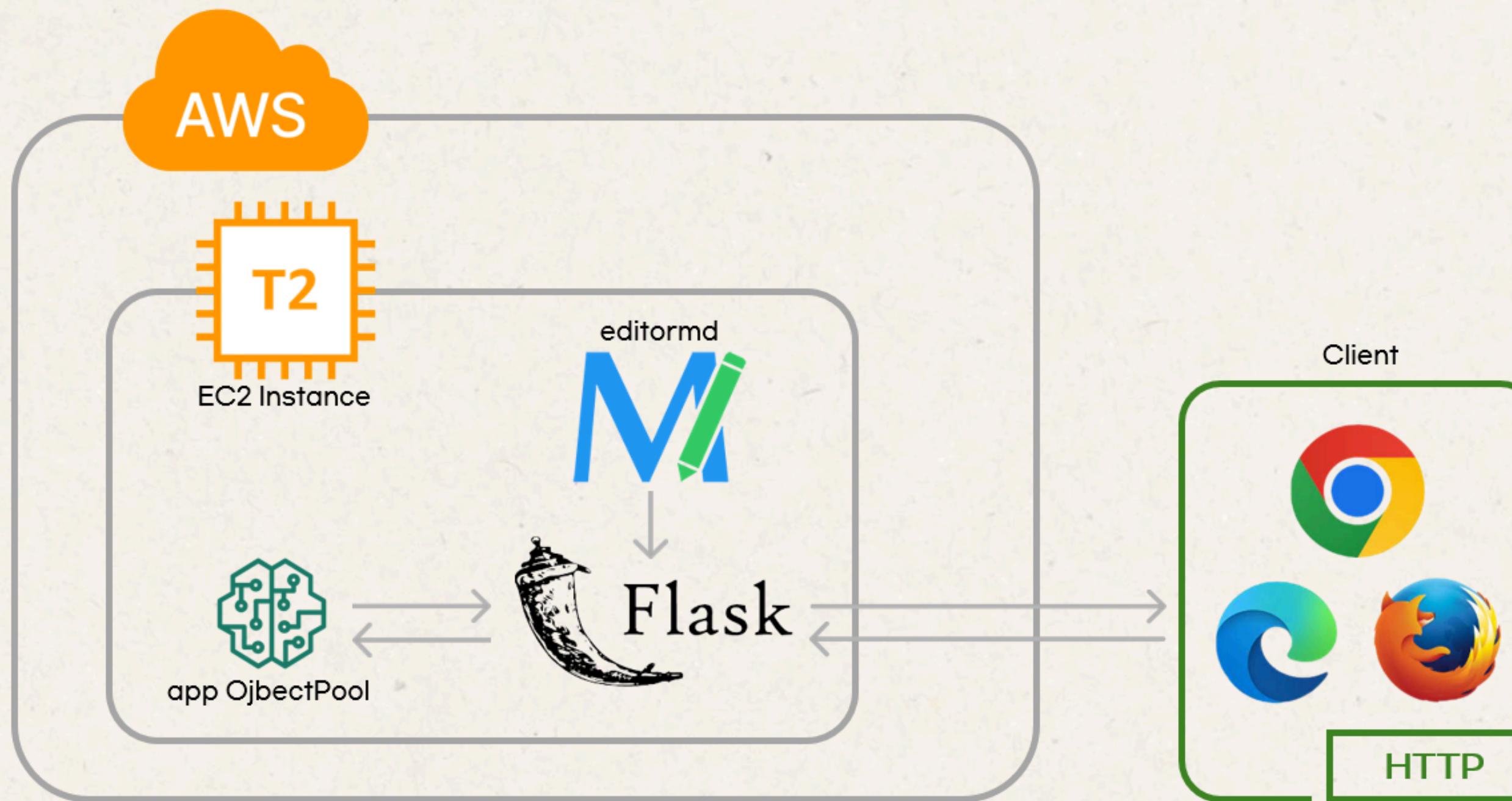
- 안저검사 망막 사진을 이용하여 자폐스펙트럼장애를 예측하는 모델이 개발되었으며, 이는 진단 및 예후 예측 시스템 구축에 활용될 수 있다.



05

Implementation

서비스 아키텍쳐



* 서비스 시연 연상 : [youtube](#)



06

Conclusion

한계점 | 의의

한계점

OCR

- OCR 모델의 'Text recognition' 단계에서 별도의 fine-tuning 없이 기존의 오픈소스 모델을 그대로 사용
-> 한국어와 영어가 섞이거나, 텍스트 이미지가 상하 두 줄로 배치된 경우 모델이 잘 인식하지 못하는 문제가 발생
- 이러한 인식 오류가 최종 평가의 fluency 점수에 부정적인 영향을 미쳤을 가능성이 있음

Image Classifier

- 데이터를 직접 구성하여 다양성과 개수가 부족했고, VGG19보다 최신 모델을 실험해보지 못함
- 이미지 위주로 구성된 PPT의 경우, 텍스트가 부족해 요약에 어려움 존재. DQA-NET처럼 이미지를 기반으로 관련 텍스트를 생성해 보완하면 요약에 도움이 될 것이라고 생각됨

Text Summarization

- G-EVAL보다 최신의 LLM 기반 평가 방법(예: Prometheus)이 존재함, 인력 부족으로 human-eval을 진행하지 못함
- 정답 요약문을 만들기 어려워 Rouge와 Bert Score와 같은 reference-based 정량 평가를 시도하지 못함
- Supert, QAFactEval과 같은 reference-free 정량 평가는 긴 텍스트에 대해 점수가 너무 낮게 산정되는 경향이 있어 비교가 어려움
- OpenAI API를 활용하여 요약하는 과정에서 프로젝트 결론 부분이 과도하게 축약되어 주요 내용이 누락되는 문제가 있음

의의

- README 파일의 내용을 계층적으로(대주제-소주제-세부내용) 구성하여 정보를 체계적으로 전달할 수 있음
- OpenAI API를 사용한 프롬프트 엔지니어링을 통해 텍스트 요약 과정에서 주요 정보의 누락을 최소화하도록 설계함
- 기본적인 틀이 완성되었으므로, 지속적으로 서비스를 운영하며 한계점을 개선해 나갈 수 있음

Thank you

NAME OF PROJECT:

Auto ReadMe Generator

PRESENTED BY:

Project Maker조
12기 권도영, 김지원, 송규현