

S.A.V.E

Speech And Vision Enhancement



1. Intro

-
- 1) 프로젝트 유형
 - 2) 서비스 소개
 - 3) 프로젝트 주제 및 선정 배경
 - 4) 프로젝트 목적 및 기대효과

프로젝트 유형

Project Types

1

모델을 직접 구현하고 기존 모델과 비교하는 유형

2

이미 구현된 모델을 인공지능 플랫폼에서 가져와 파인튜닝하는 유형

3

OpenAPI를 사용하는 유형

4

웹으로 인공지능 서비스를 제공하는 유형

5

자바 웹과 파이썬 인공지능 플랫폼을 연동하는 유형

서비스 소개

Service Introduction

SAVE

speech and vision enhancement

실시간 객체 탐지 및 캡셔닝을 활용한
시각장애인 음성 안내 서비스

프로젝트 주제 및 선정 배경

Project Topic and Selection Background

2023년 기준 우리나라 시각장애인 수는 **238,000명**

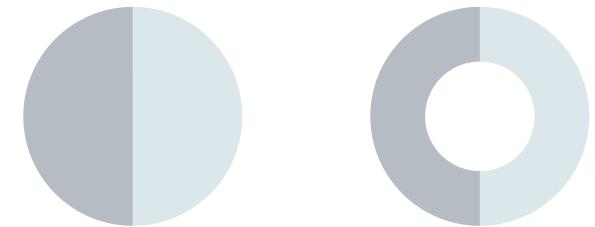
반면, 활동 중인 시각장애인 안내견은 불과 **70여 마리**

약0.029%



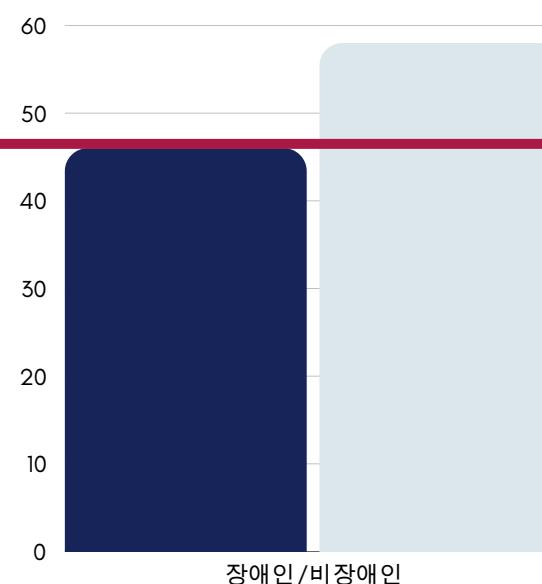
실제 보도 이용 만족도 조사

불만족 50%:
편의시설 부족



시각장애인의 보도 이용만족도

비장애인 평균에
비해 낮음



프로젝트 목적 및 기대효과

Project Objectives and Expected Outcomes

일상 생활의 편의성 증가

사회적 상호작용 강화

사고 예방

- 시각장애인들이 스스로 물건을 찾고, 이해하고, 필요한 작업을 수행하는 데 도움을 줄 수 있습니다.
- 상호작용에 대한 이해도 상승 및 원활한 소통과 관계 형성에도 도움이 될 수 있습니다.
- 시각장애인들이 타인과의 소통에서 더 많은 정보를 얻을 수 있습니다.
- 도로 위 위험 물체가 나타났을 때 음성으로 알려줌으로써 사고를 예방할 수 있다.



2. Product Serving

-
- 1) 프로젝트 흐름도
 - 2) 시스템 아키텍처
 - 3) 프로젝트 수행 절차 및 방법 #1, #2
 - 4) 프로젝트 방법론

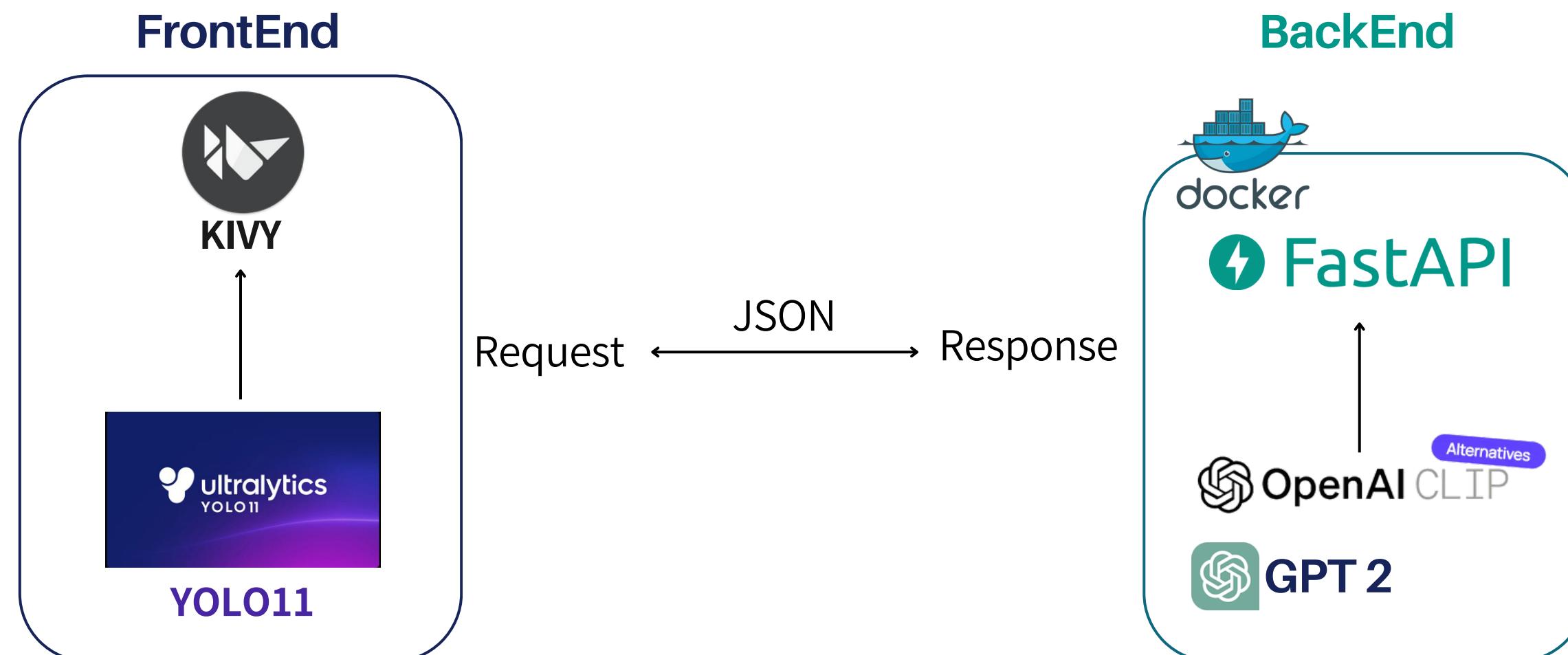
프로젝트 흐름도

Project Flowchart



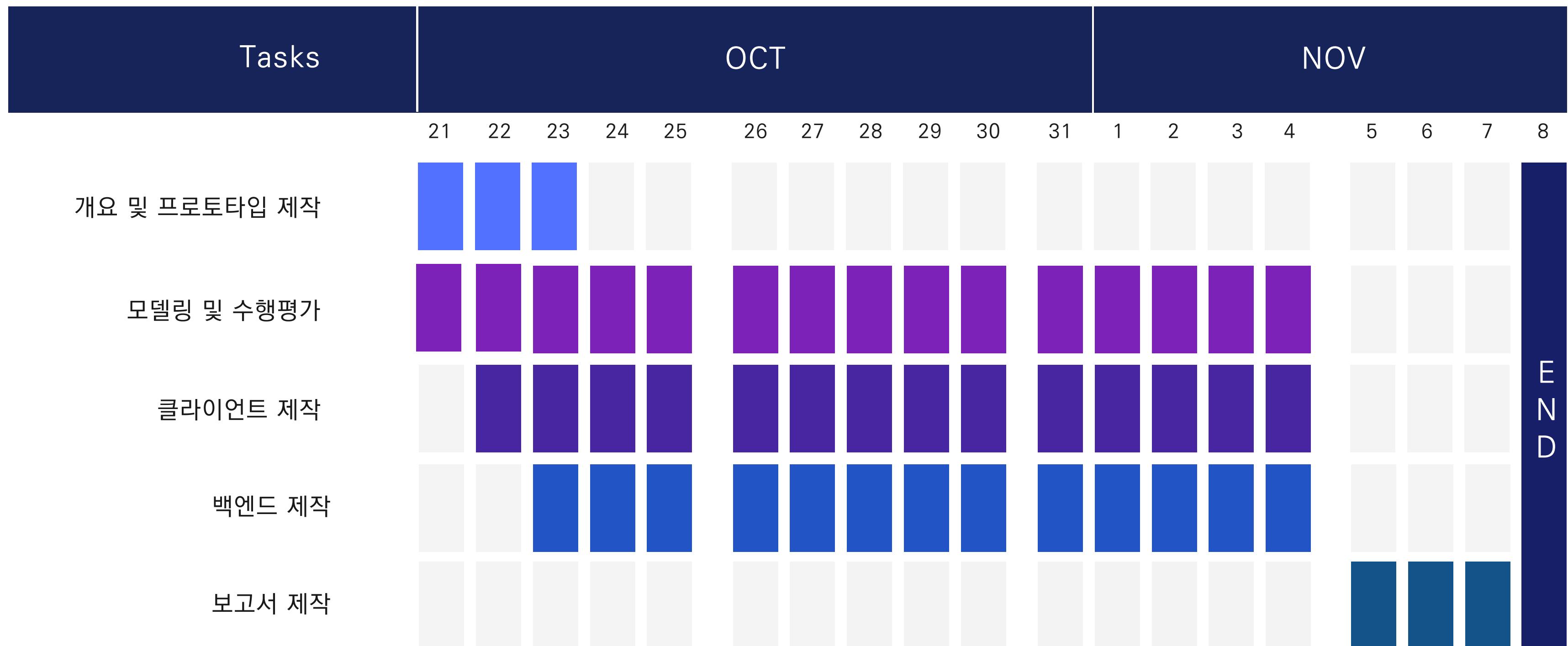
시스템 아키텍처

System Architecture



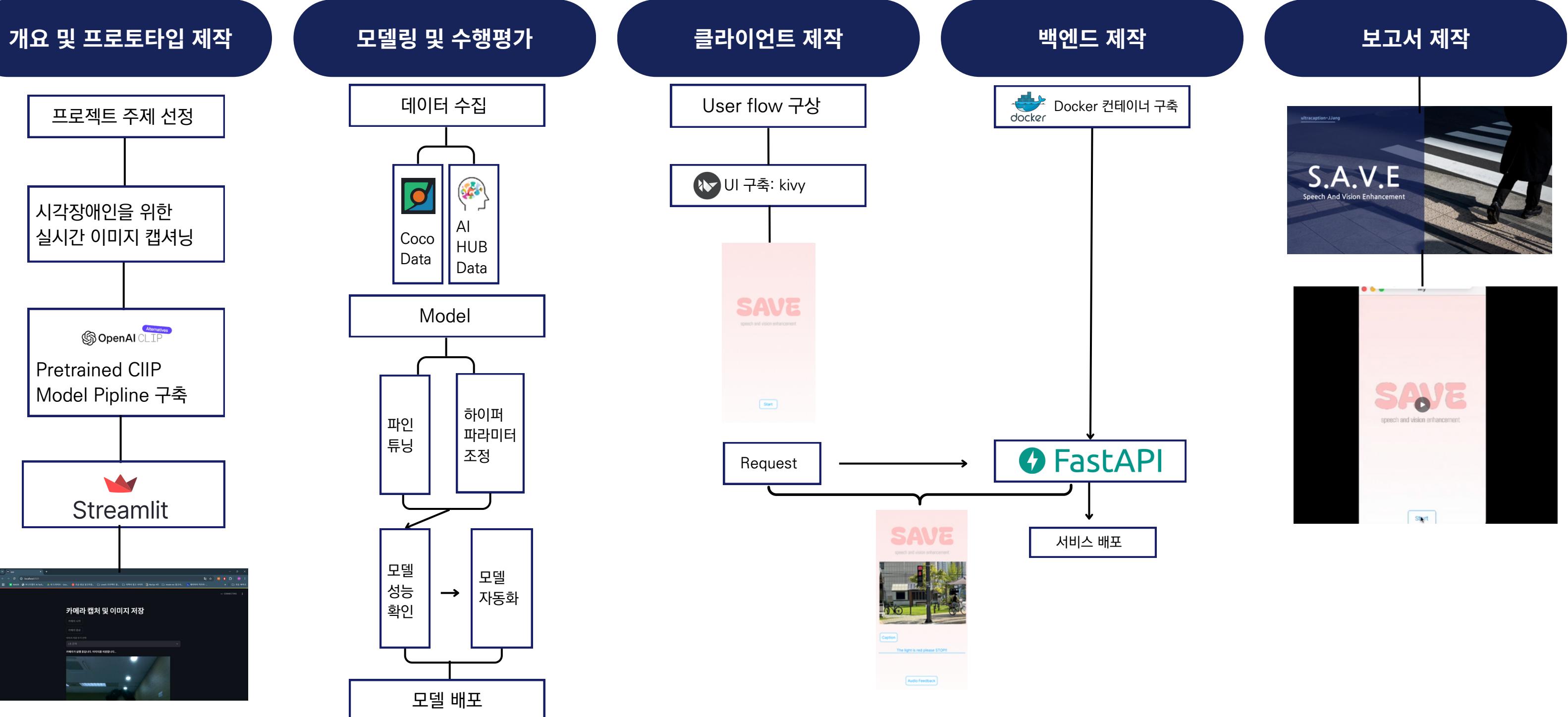
프로젝트 수행 절차 및 방법 #1

Project Execution Procedure and Method #1



프로젝트 수행 절차 및 방법 #2

Project Execution Procedure and Method #2



프로젝트 방법론

Project Methodology

Agile



3. Research

-
- 1) 프로젝트 환경 및 도구
 - 2) 데이터 수집 전략
 - 3) EDA #1
 - 4) EDA #2

프로젝트 환경 및 도구 (기술 스택)

Project Environment and Tools (Technology Stack)

Language	Client	Api Serving	Analysis
 3.12	 1.32.0  2.3.0  1.2.0	 0.115.3  4.34.3	 2.5.1  4.46.2
Dev Tools	Tools	Hardware/Server	
    	  	 Windows 11  macOS	 v11  5.22.0  Weights & Biases  2.2.2  4.10.0.84  3.8.4  2.5.3

데이터 수집 전략

Data Collection Strategy

데이터
수집



COCO
Common Objects in Context

MS COCO 캡션 데이터셋

데이터
클래스

80개 클래스

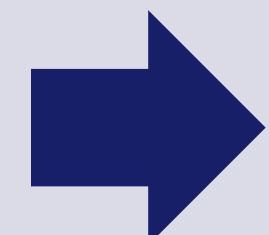
클래스
선정
기준

- 1) 움직이는 사물
- 2) 도로에서 볼 수 있는 사물
- 3) 생필품



인도보행영상 데이터셋

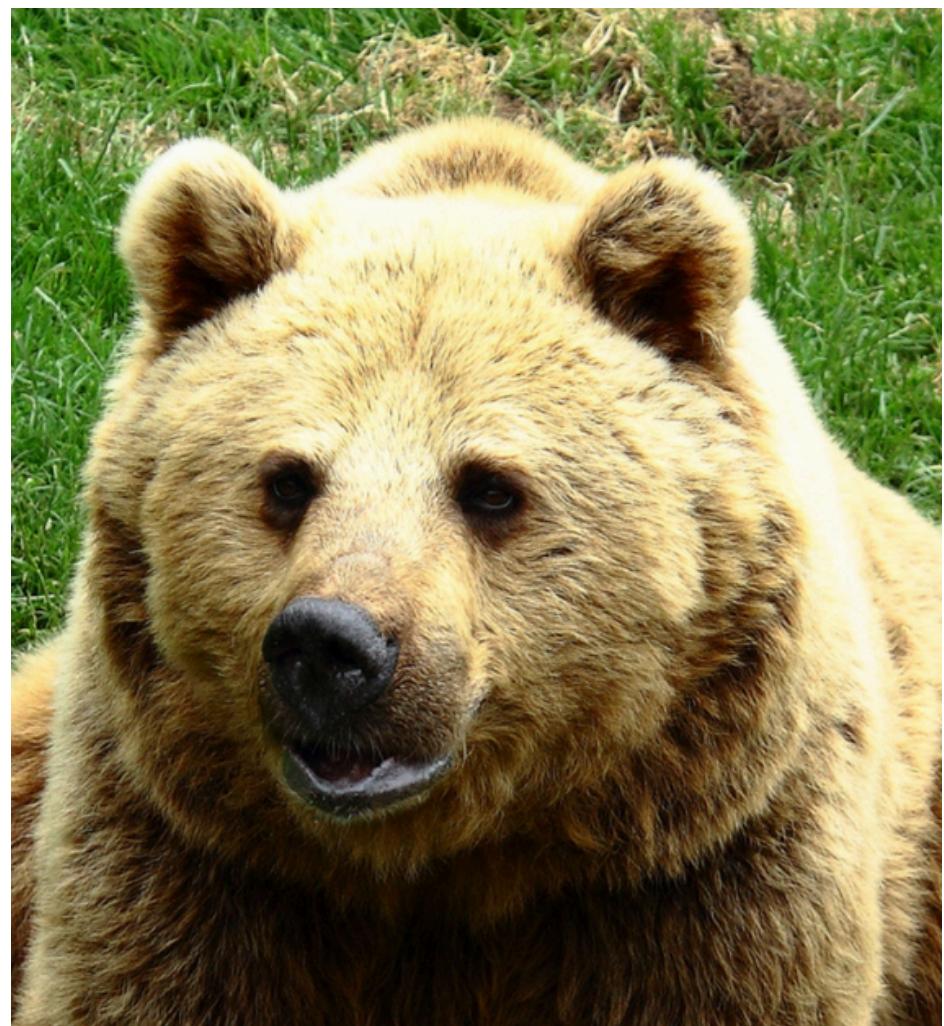
27개 클래스



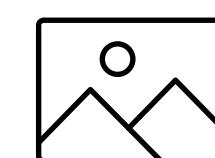
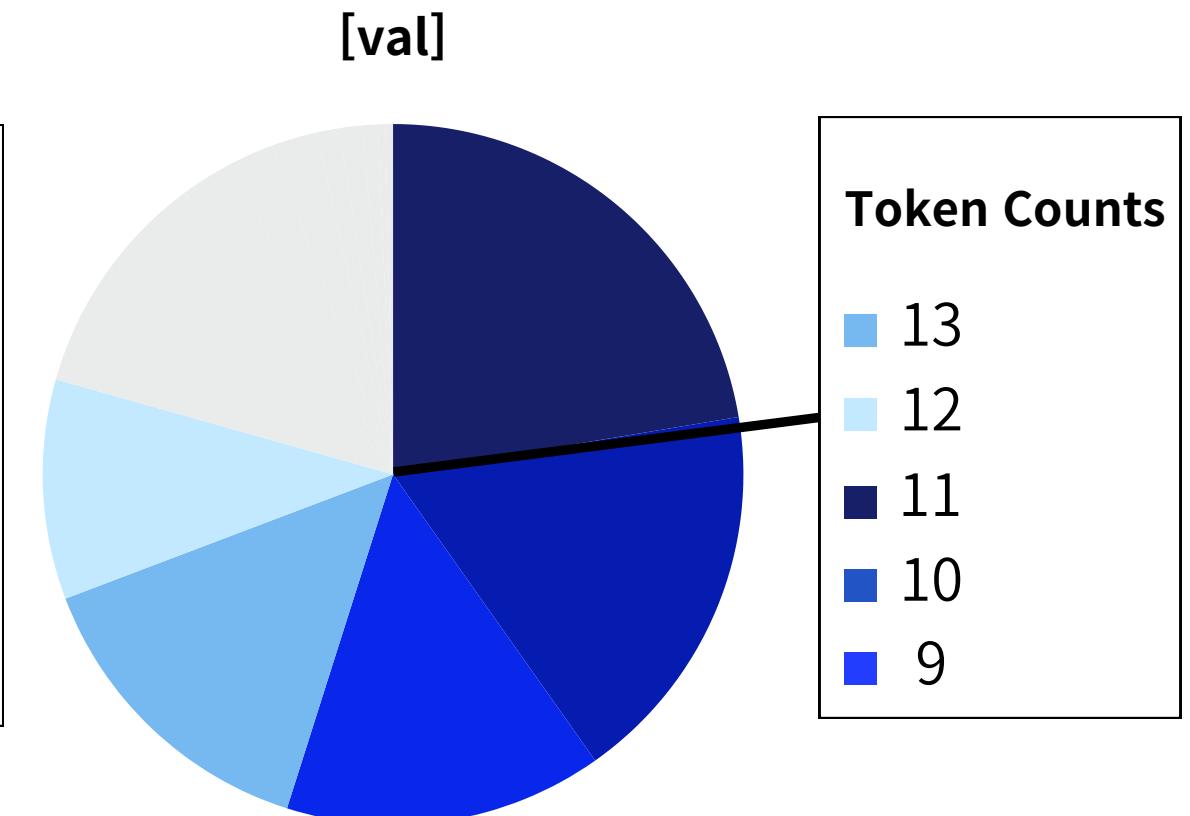
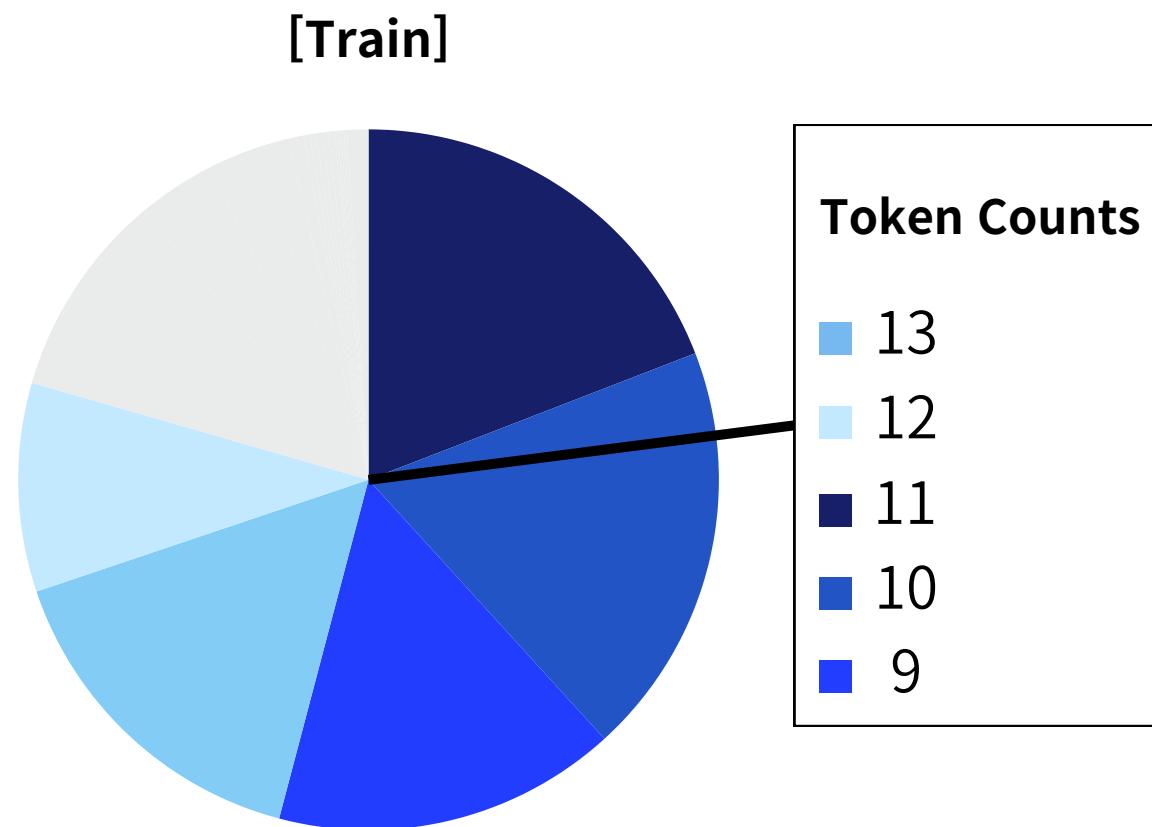
최종 15개 클래스

EDA #1

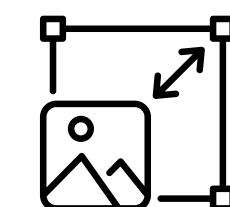
clip-coco dataset



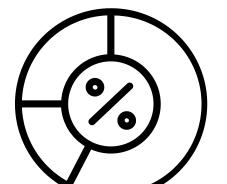
“a close up of a bear in a field of grass”



이미지 총 개수
700(train) + 286(val)



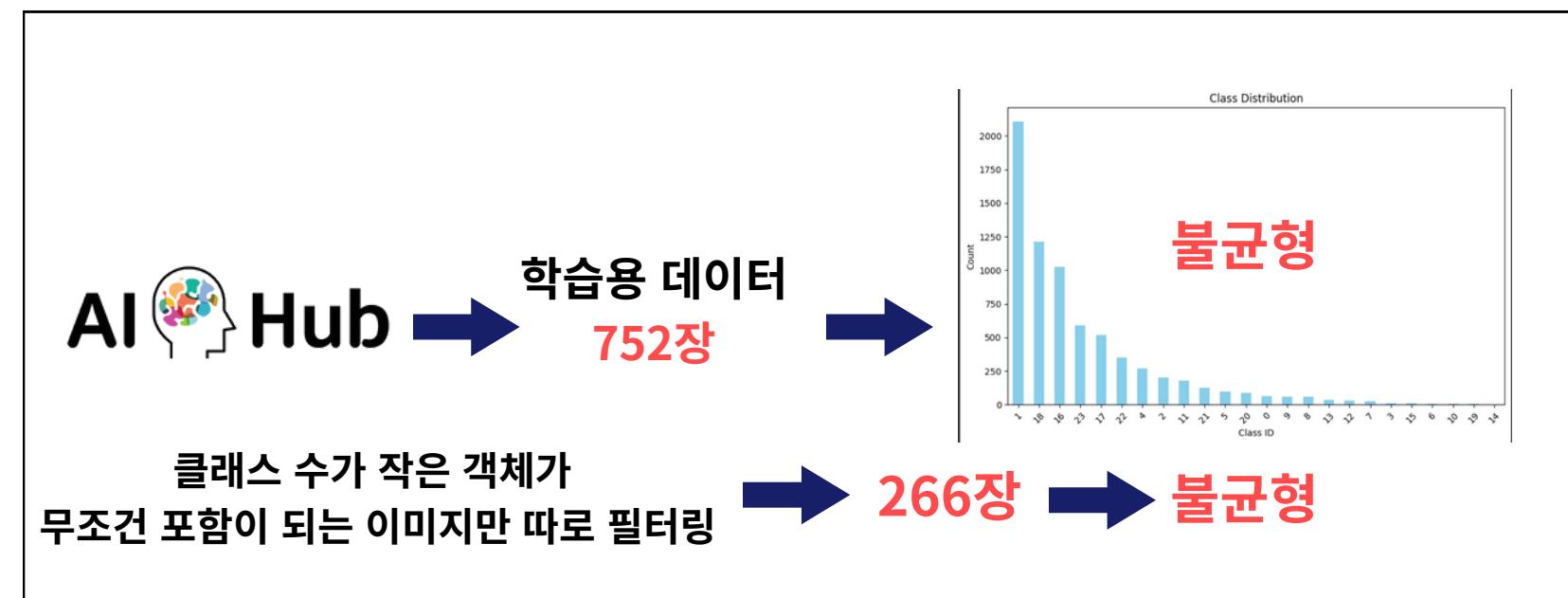
이미지
사이즈 동일



비율
7:3

EDA #2

YOLO-AI HUB dataset

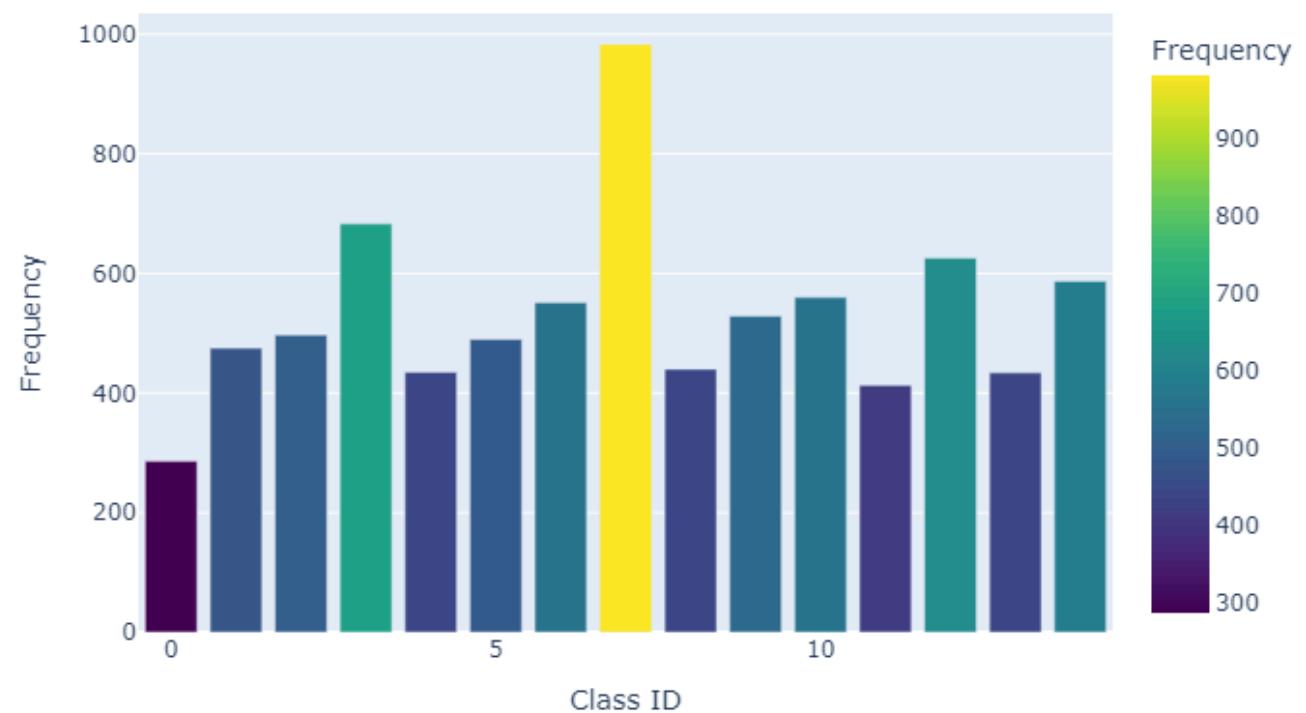


데이터 증강



데이터셋: 8125장
클래스 15개

Class Distribution



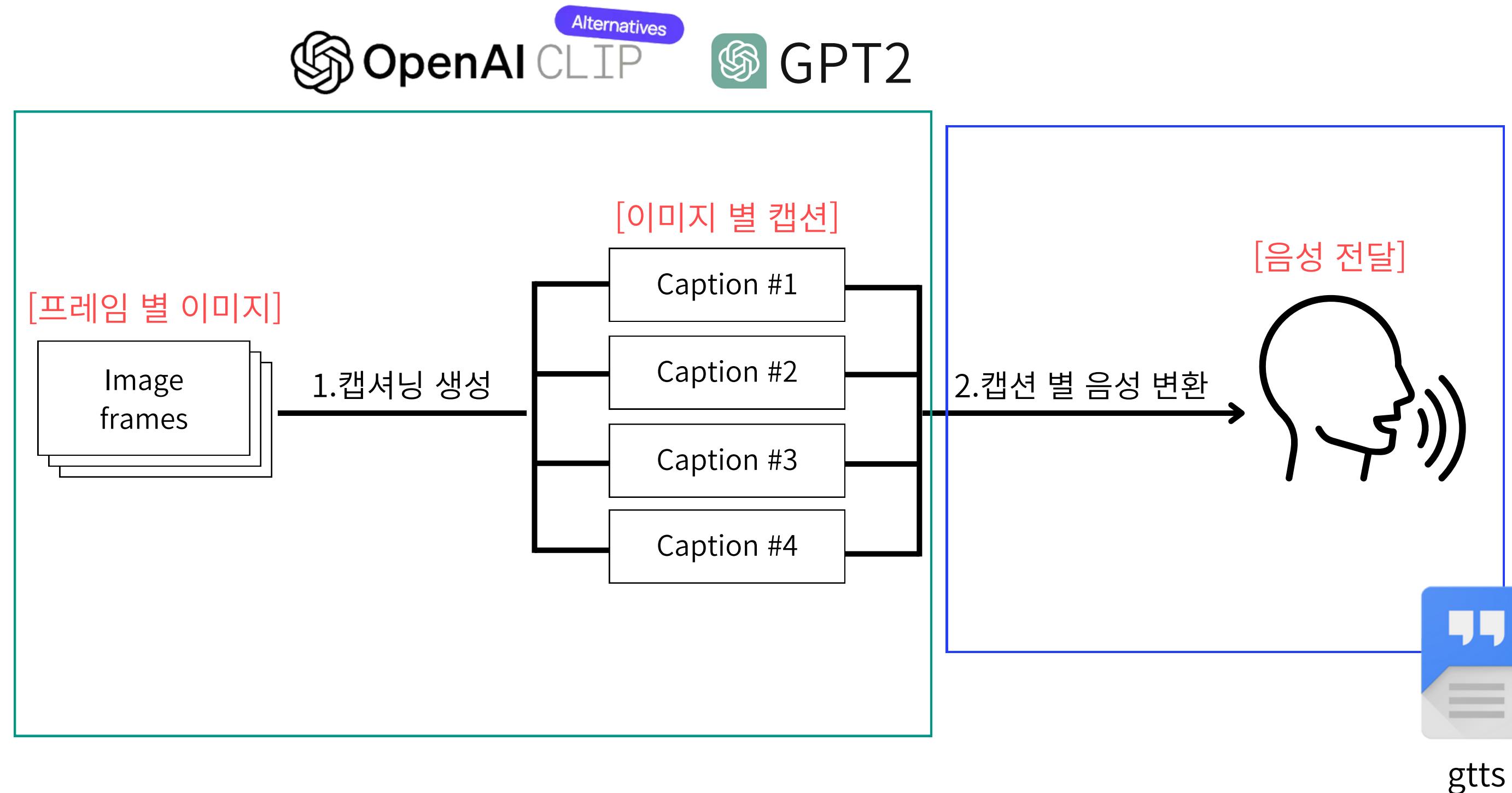
데이터 불균형 문제 해소

4. Model

-
- 1) 전체 모델 아키텍처
 - 2) YOLO, CLIP, CLIPCAP 모델 아키텍처
 - 3) 실험 결과

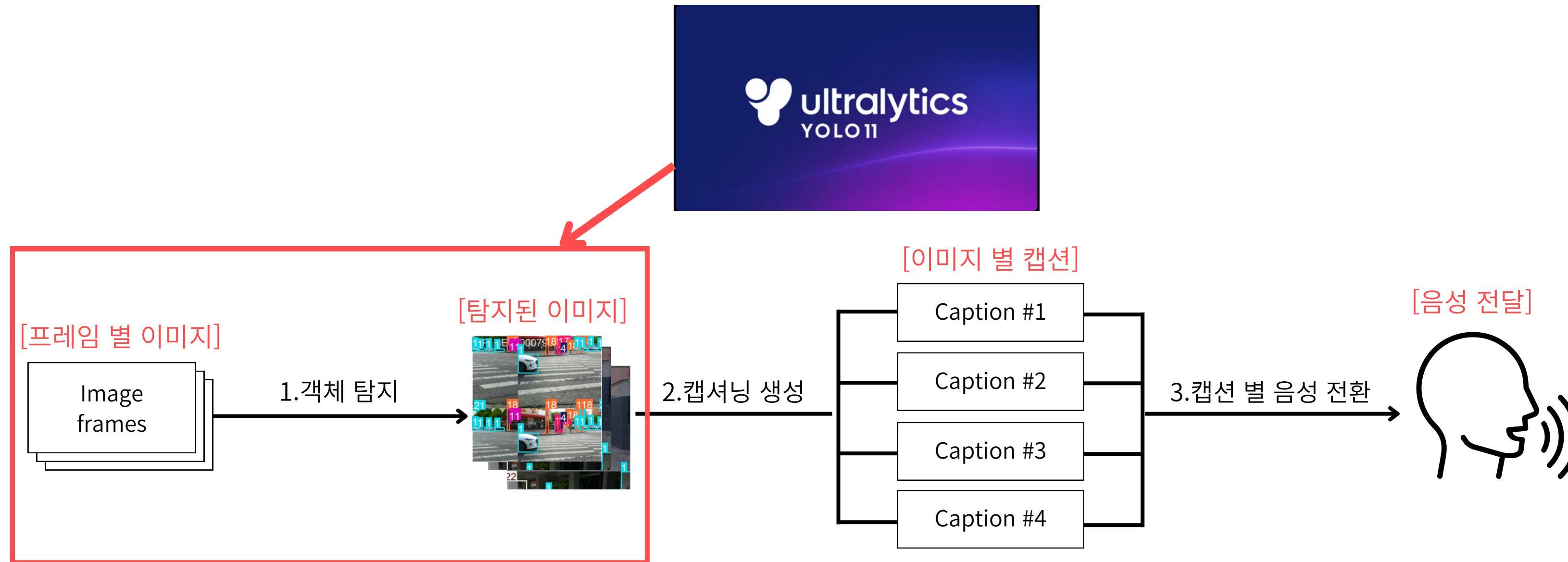
전체 모델 아키텍처 버전 1

Overall Model Architecture



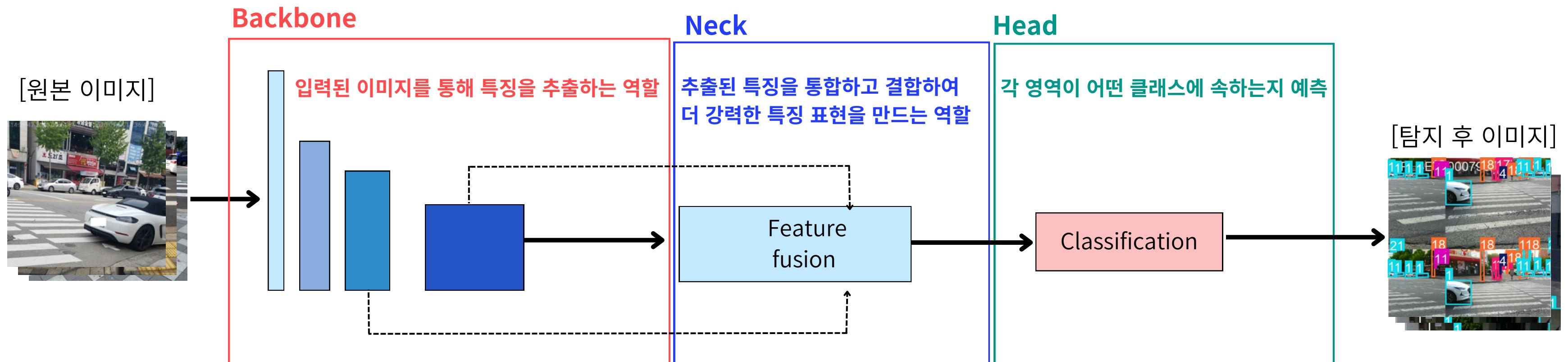
전체 모델 아키텍처 버전 2

Overall Model Architecture



YOLO 모델 아키텍처

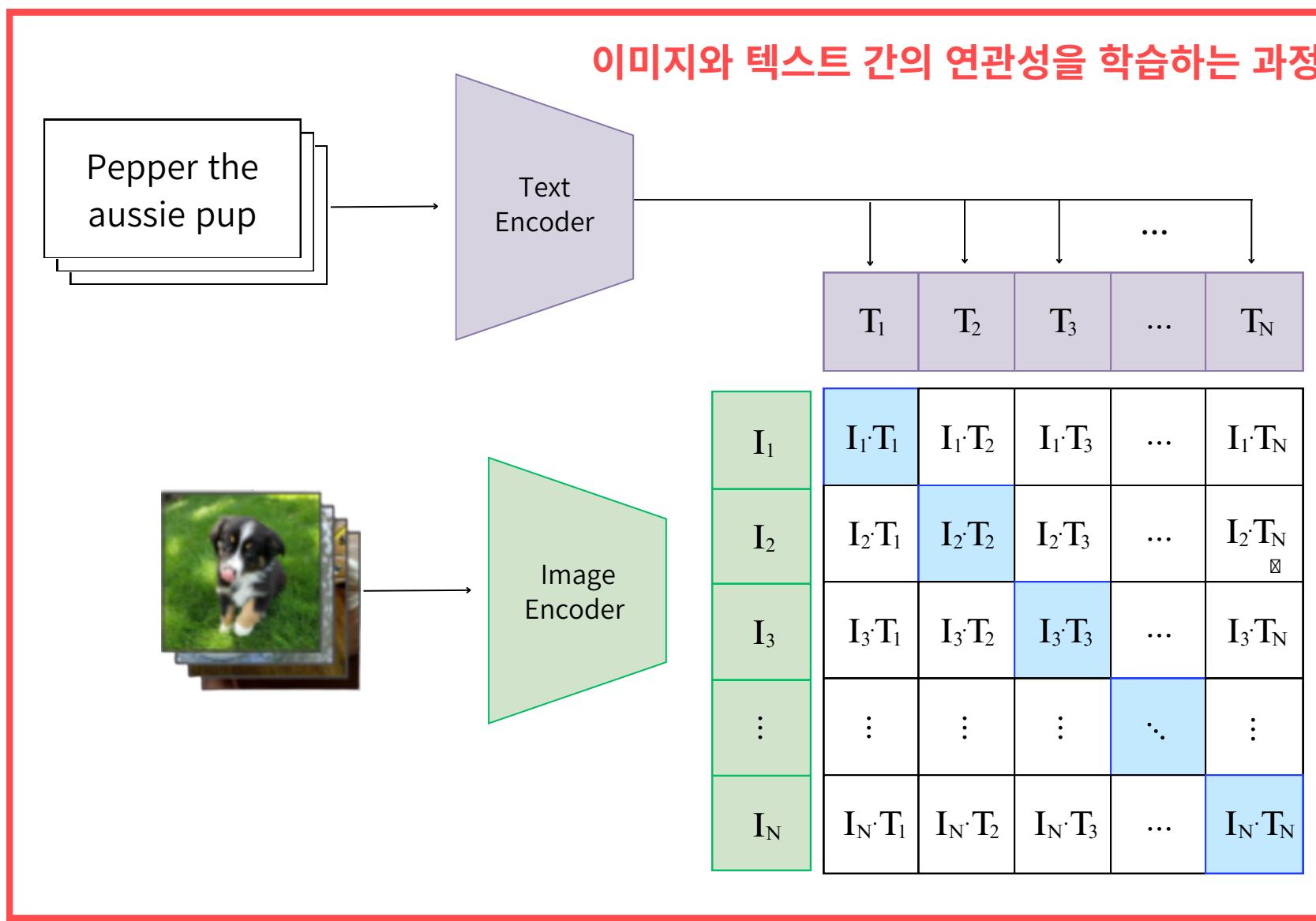
YOLO Model Architecture



CLIP 모델 아키텍처

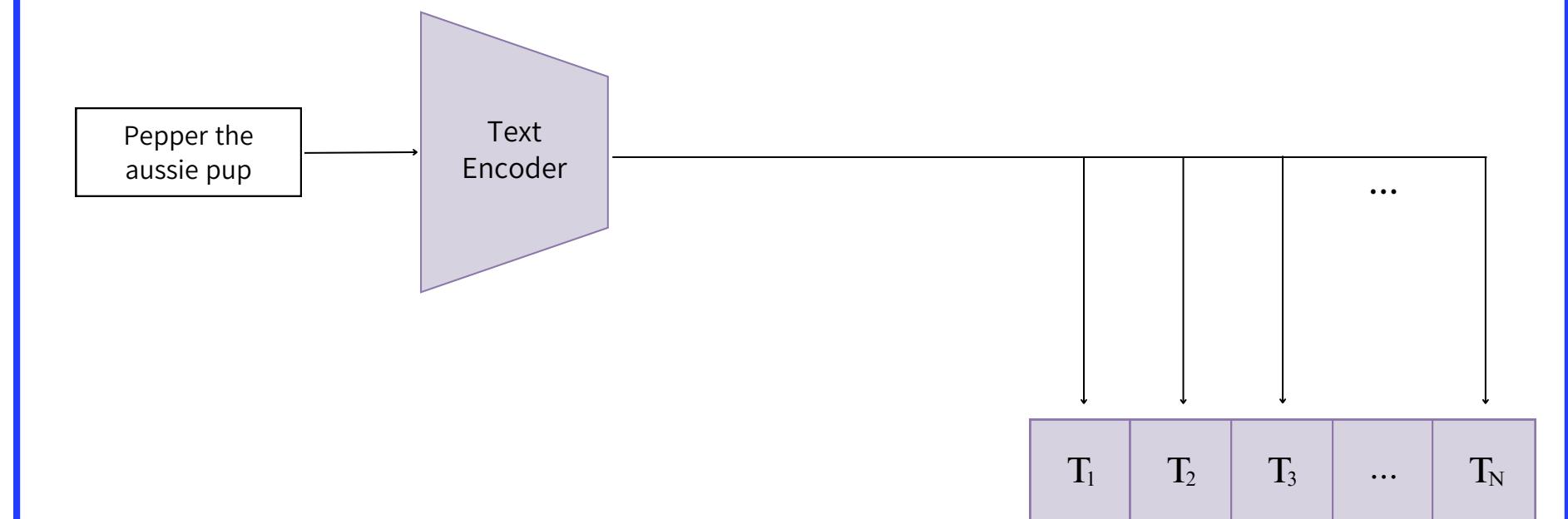
CLIP Model Architecture

1) 대조 학습(Contrastive pre-training)

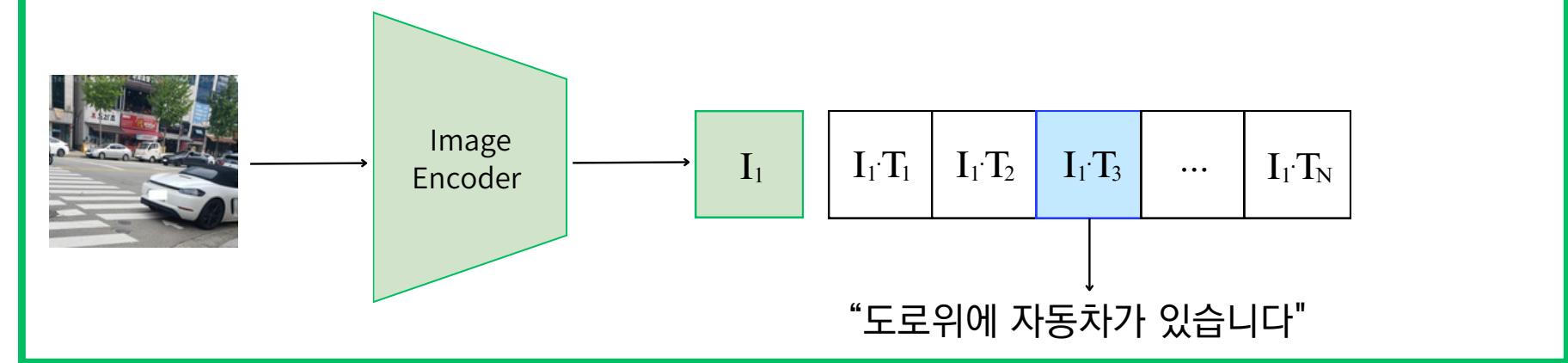


2) 텍스트로 데이터셋 분류기 생성

텍스트를 벡터화해 새로운 이미지와 가장 유사한 라벨을 예측하는 과정

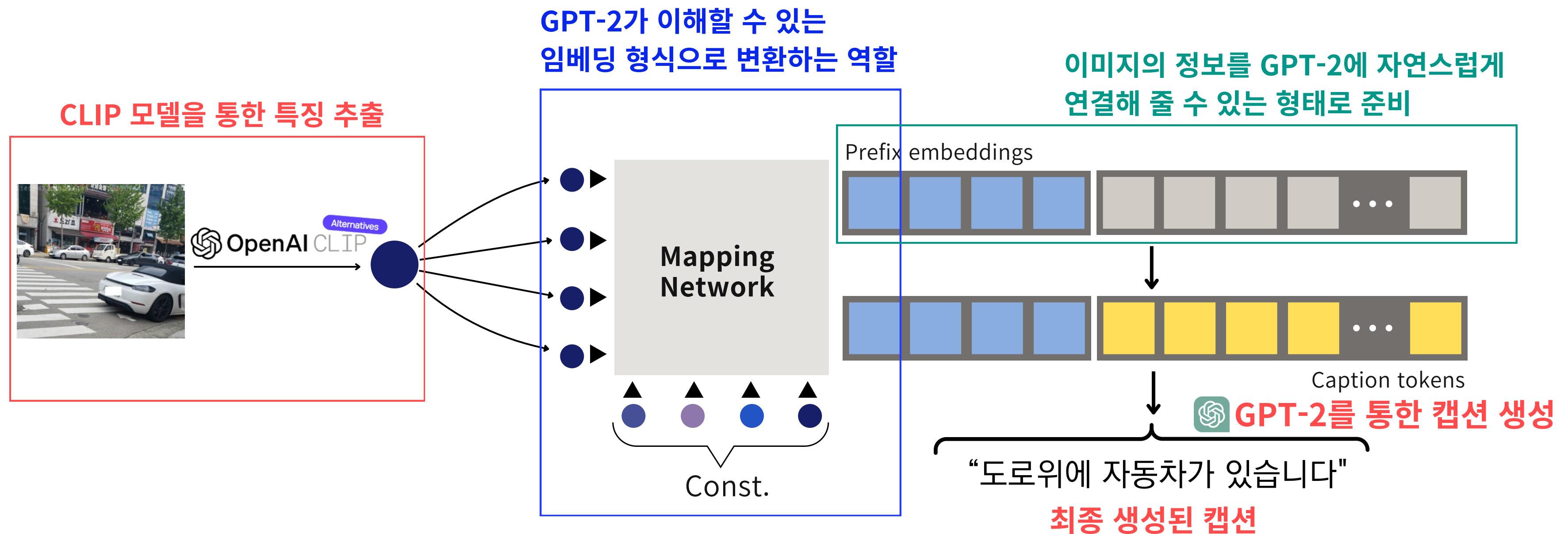


3) 제로샷 예측에 사용



CLIPCAP 모델 아키텍처

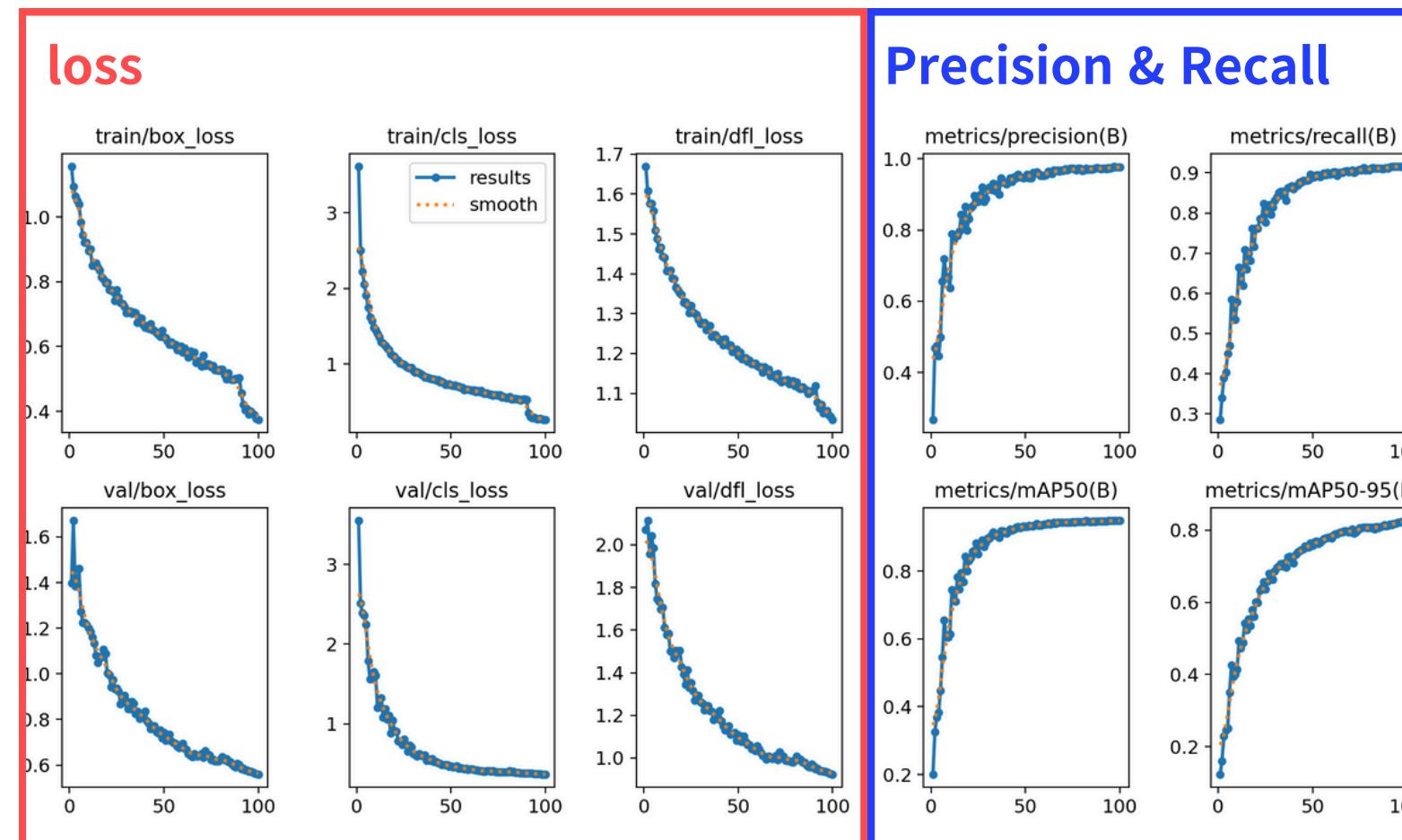
CLIPCAP Model Architecture



실험 결과 #1: YOLO

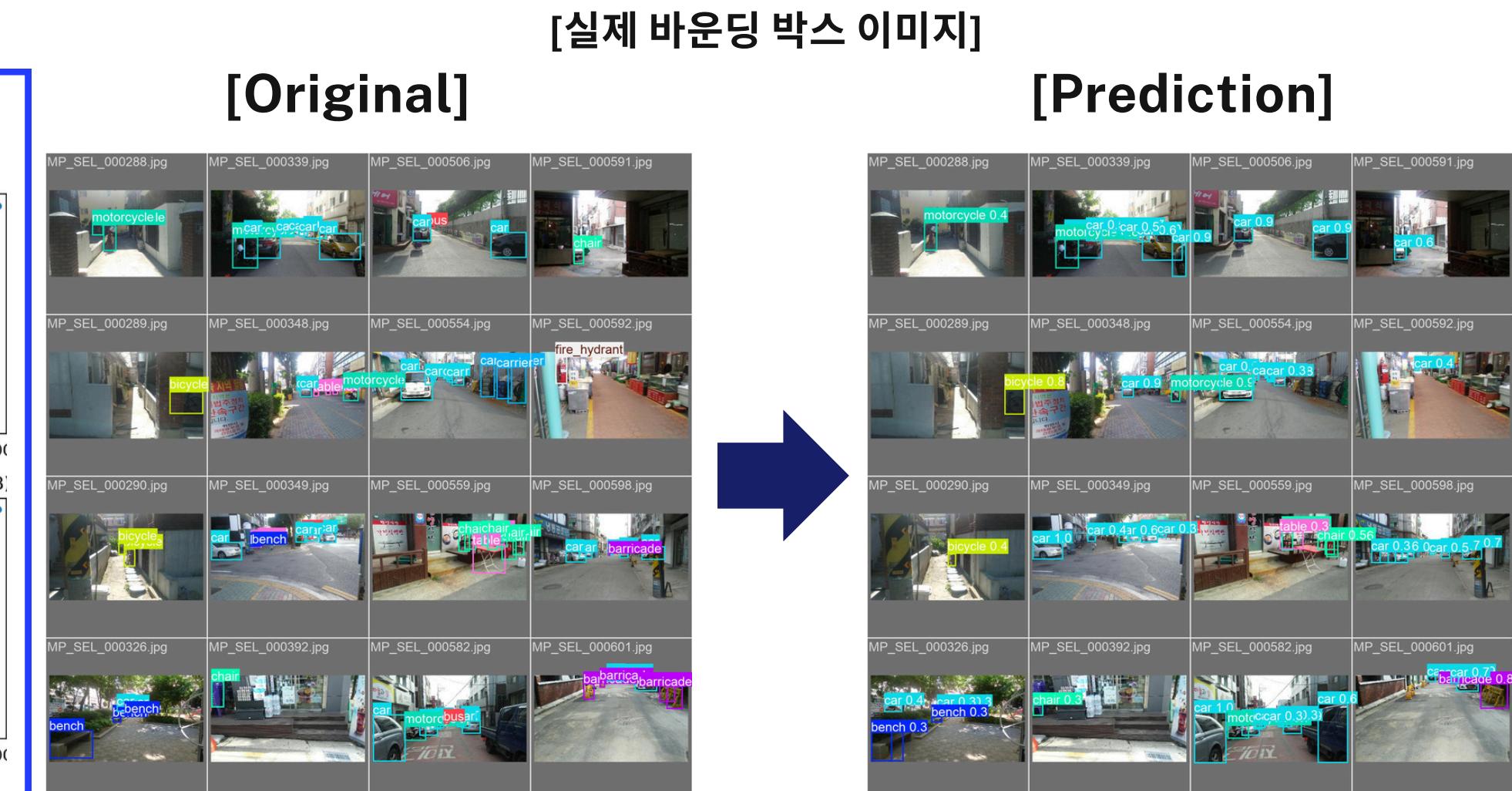
파인튜닝

[훈련 과정의 손실 함수와 정확도 지표]



[각 클래스에 대한 성능 지표]

Class	Images	Instances	Box(p)	R	mAP50	mAP50-95
ALL	2437	3476	0.975	0.913	0.949	0.872
Bench	79	160	0.942	0.888	0.887	0.688
Car	51	199	0.832	0.613	0.744	0.482
Stroller	172	172	0.991	0.994	0.995	0.928
...
Stop	170	170	0.998	0.994	0.995	0.97
Chair	176	221	0.972	0.873	0.919	0.816

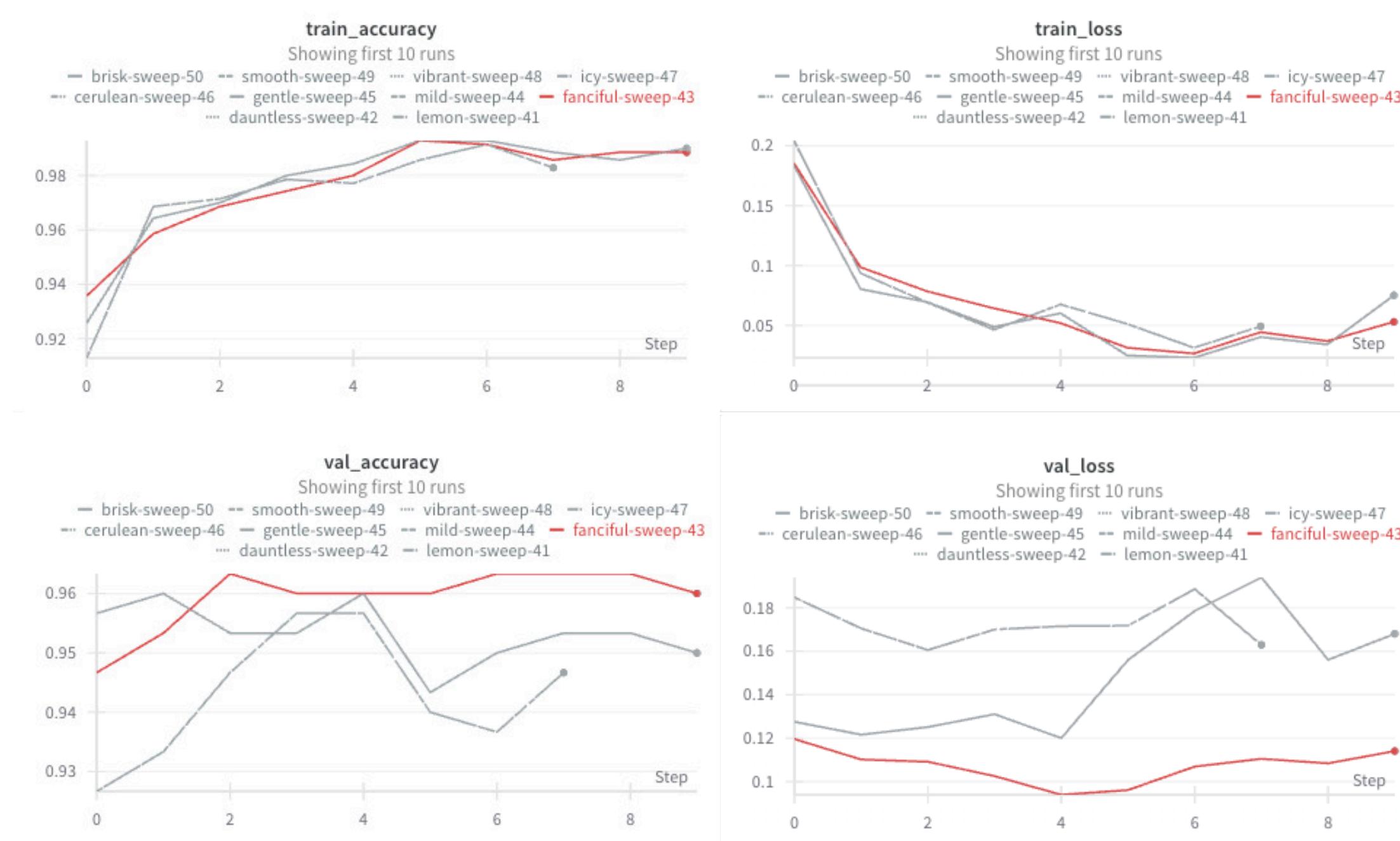


ALL 항목을 통해 YOLO 모델의 성능을 종합적으로 알 수 있었습니다.

실험 결과 #2: CLIP

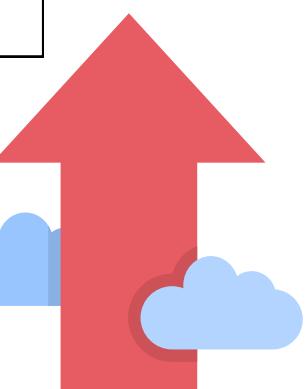
파인튜닝

[Train 및 Validation 성능 지표]



Best model	
Epoch (Early Stopping)	9
Train_accuracy	0.98857
Train_loss	0.053019
Val_accuracy	0.96
Val_loss	0.11406

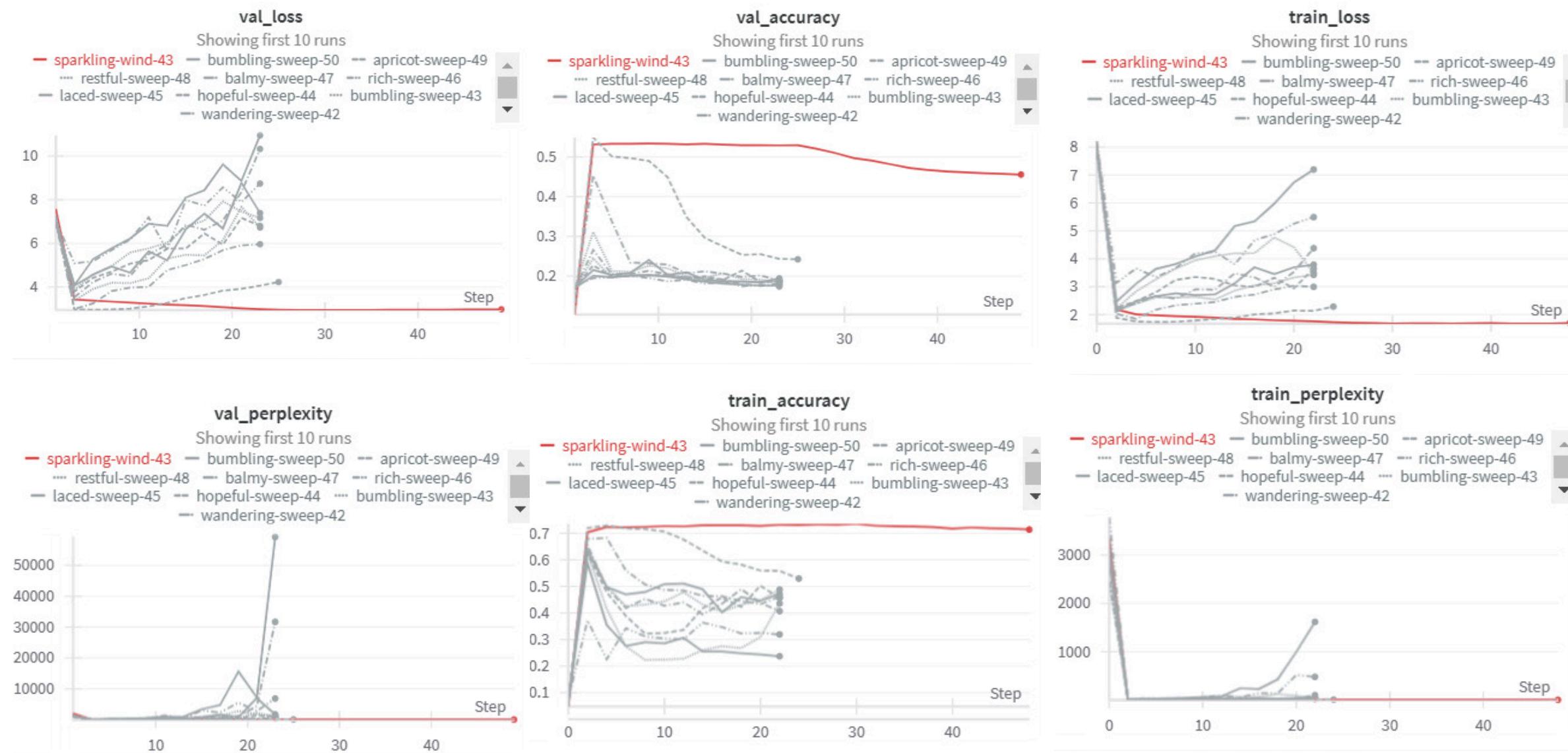
훈련 데이터와 검증 데이터
모두에 대해 **높은 성능** 나타낸다



실험 결과 #3: CLIPCAP

파인튜닝

[Train 및 Validation 성능 지표]



█ Best model	
Epoch	49
Train_accuracy	0.7140
Train_loss	1.6987
Train_perplexity	5.4742
Val_accuracy	0.4555
Val_loss	2.9744
Val_perplexity	19.6198

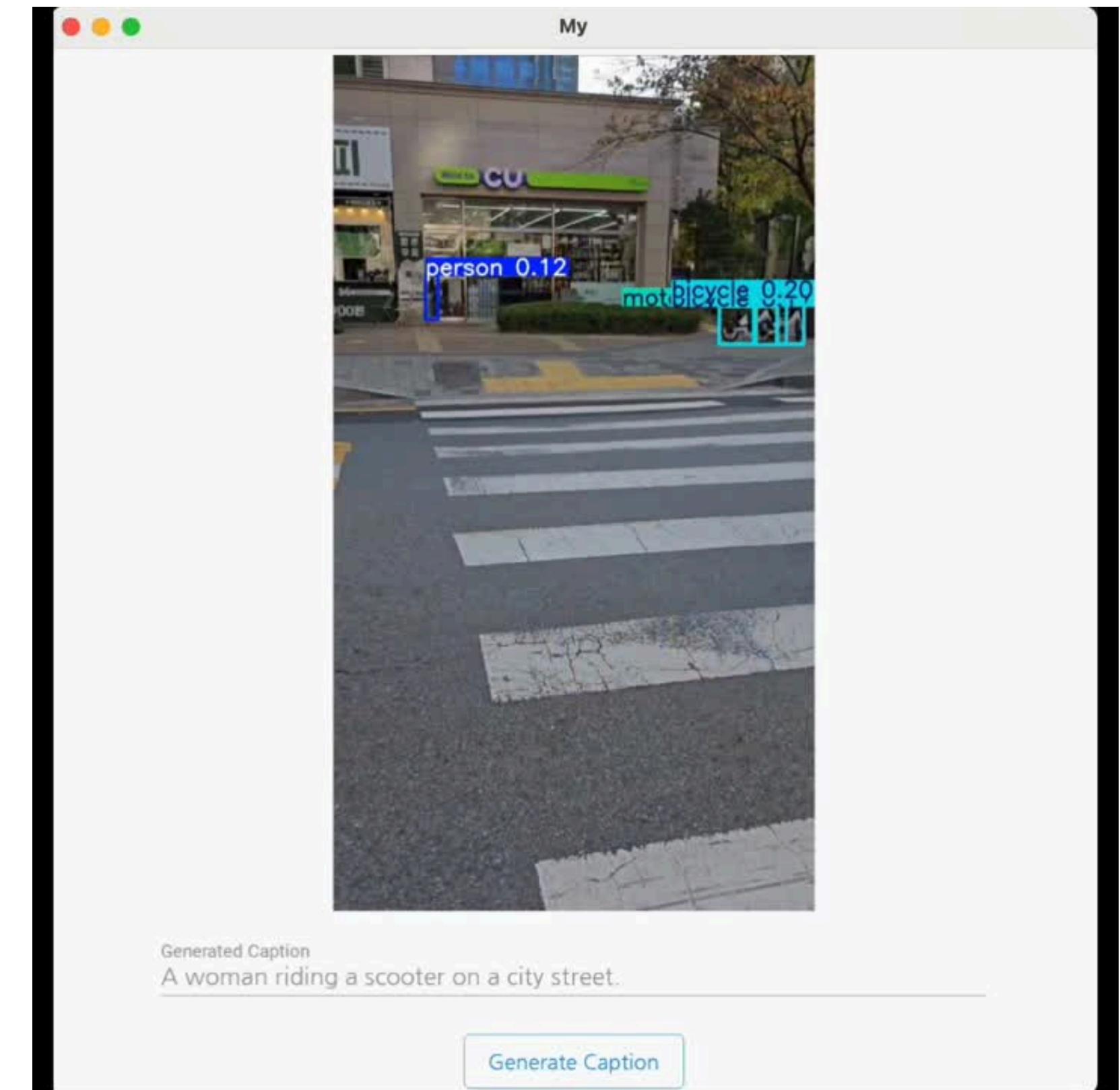
모델이 훈련 데이터에 비해 검증 데이터에 덜 일반화되어 있습니다
Train과 Val set의 정확도 차이로 인해 모델의 추후 성능 개선이 필요합니다

5.Result/Conclusion

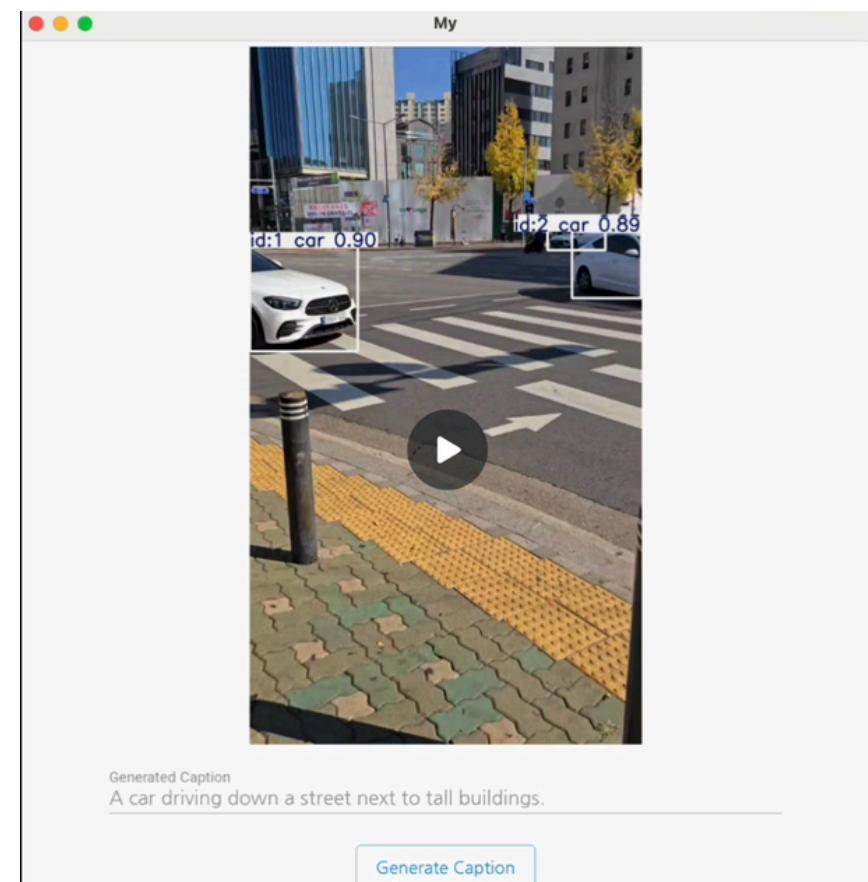
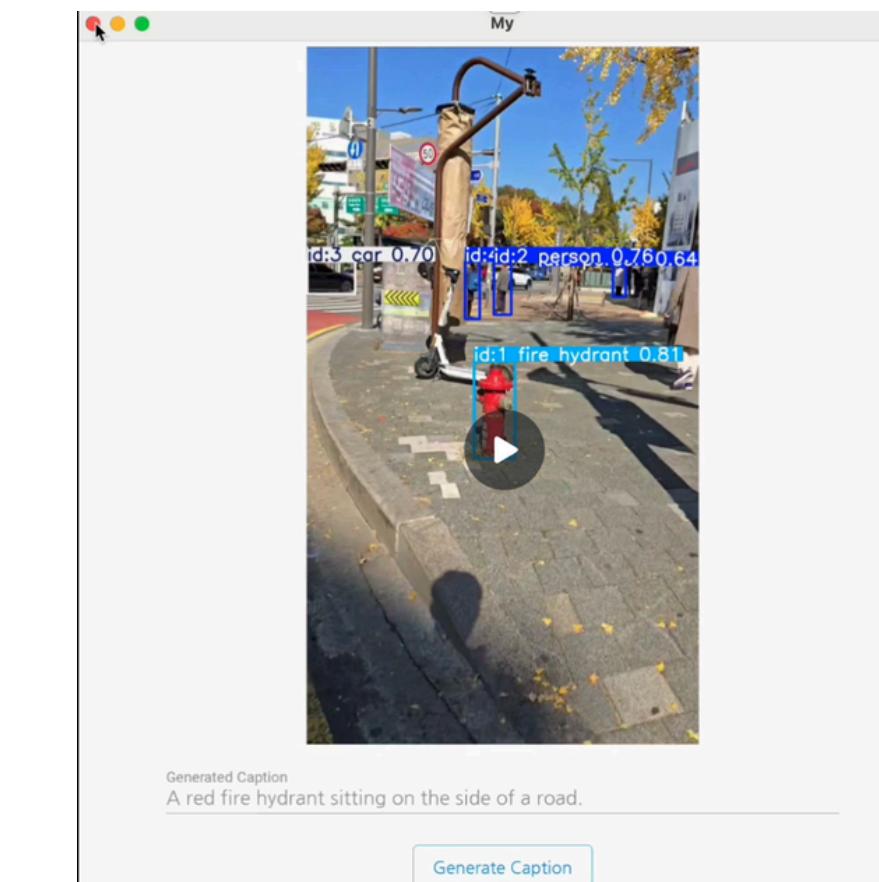
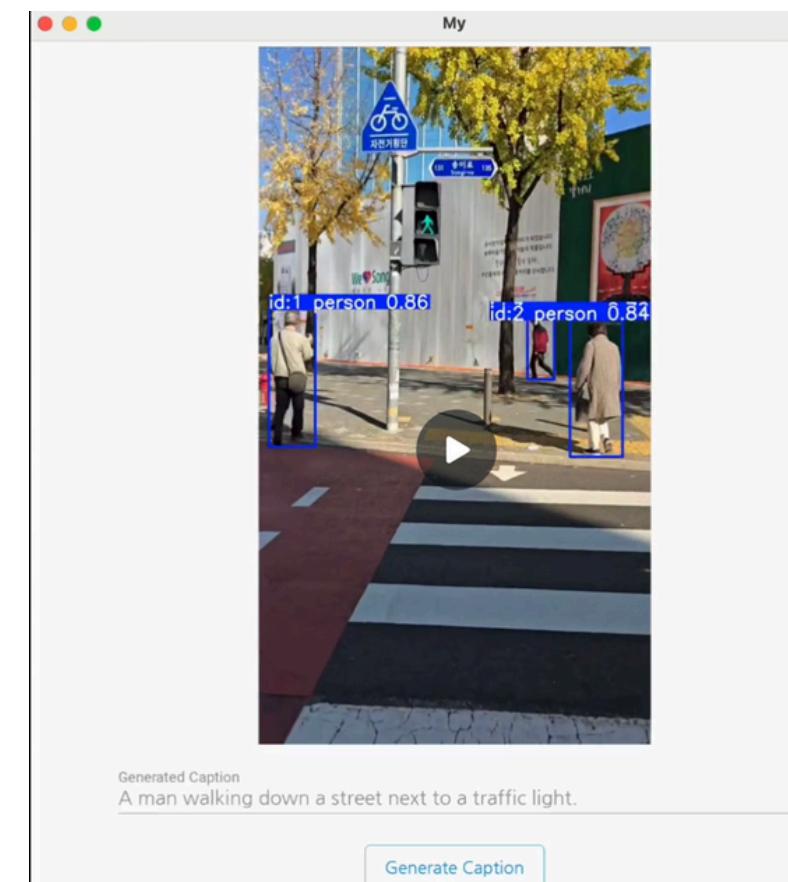
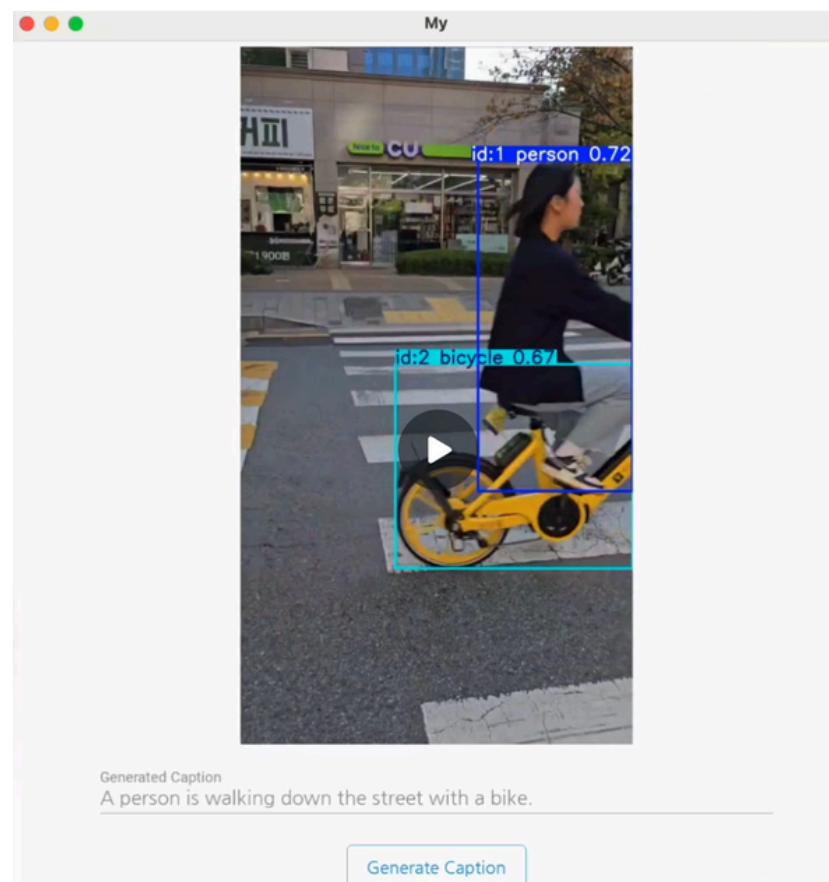
1) 프로젝트 수행 결과 - 시연

프로젝트 수행 결과 - 시연

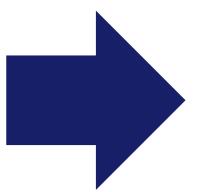
```
choo@DESKTOP-03BCRPD:/n + ^  
  
INFO: 172.17.0.1:59280 - "POST /generate-caption/ HTTP/1.1" 200 OK  
캡션 생성 소요 시간: 0.96 seconds  
INFO: 172.17.0.1:44344 - "POST /generate-caption/ HTTP/1.1" 200 OK  
캡션 생성 소요 시간: 0.98 seconds  
INFO: 172.17.0.1:44348 - "POST /generate-caption/ HTTP/1.1" 200 OK  
캡션 생성 소요 시간: 0.97 seconds  
INFO: 172.17.0.1:56124 - "POST /generate-caption/ HTTP/1.1" 200 OK  
캡션 생성 소요 시간: 0.95 seconds  
INFO: 172.17.0.1:56130 - "POST /generate-caption/ HTTP/1.1" 200 OK  
캡션 생성 소요 시간: 0.98 seconds  
INFO: 172.17.0.1:43146 - "POST /generate-caption/ HTTP/1.1" 200 OK  
캡션 생성 소요 시간: 0.96 seconds  
INFO: 172.17.0.1:43162 - "POST /generate-caption/ HTTP/1.1" 200 OK  
캡션 생성 소요 시간: 0.90 seconds  
INFO: 172.17.0.1:47540 - "POST /generate-caption/ HTTP/1.1" 200 OK  
캡션 생성 소요 시간: 0.97 seconds  
INFO: 172.17.0.1:47556 - "POST /generate-caption/ HTTP/1.1" 200 OK  
캡션 생성 소요 시간: 1.35 seconds  
INFO: 172.17.0.1:58954 - "POST /generate-caption/ HTTP/1.1" 200 OK  
캡션 생성 소요 시간: 0.97 seconds  
INFO: 172.17.0.1:58966 - "POST /generate-caption/ HTTP/1.1" 200 OK
```



프로젝트 수행 결과 - 시연 단계 별 설명



프로젝트 개선사항



현재 YOLO모델은 다른 클래스에 비해
유난히 Car 클래스를 잘 감지합니다

이는 추후에 개선해 나갈 것입니다

6.Appendix

-
- 1) 자체 평가 의견 또는 프로젝트 후기
 - 2) 프로젝트 팀 구성 및 역할
 - 3) 협업 전략
 - 4) References

프로젝트 팀 구성 및 역할



박성재

Modeling

- YOLO
- Clip
- ClipCap



추희정

Backend

- YOLO
- FastAPI



이승준

Project Manager

- ALL



백승제

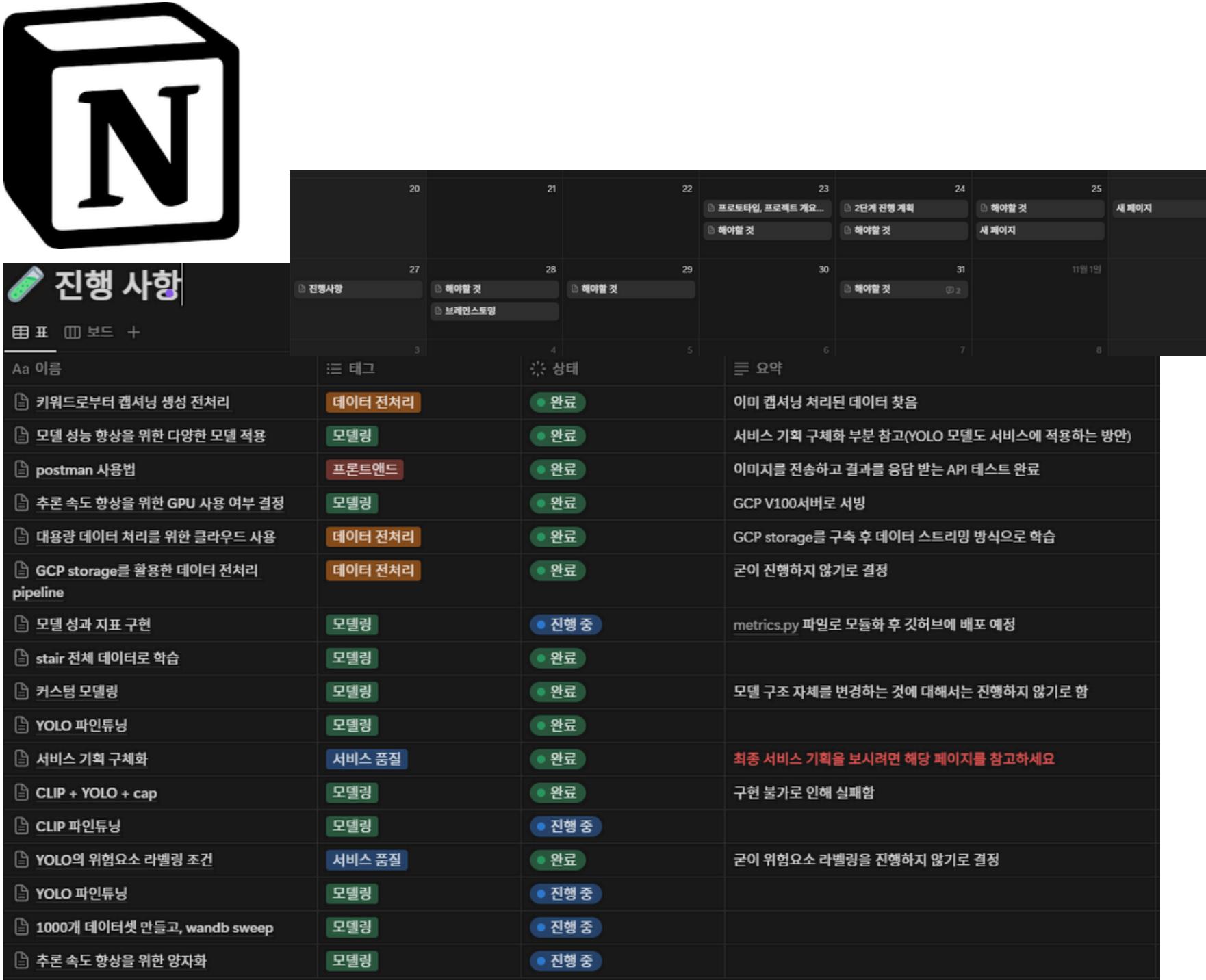
Frontend

- YOLO
- KIVY
- KIVYMD

자체 평가 의견 또는 프로젝트 후기

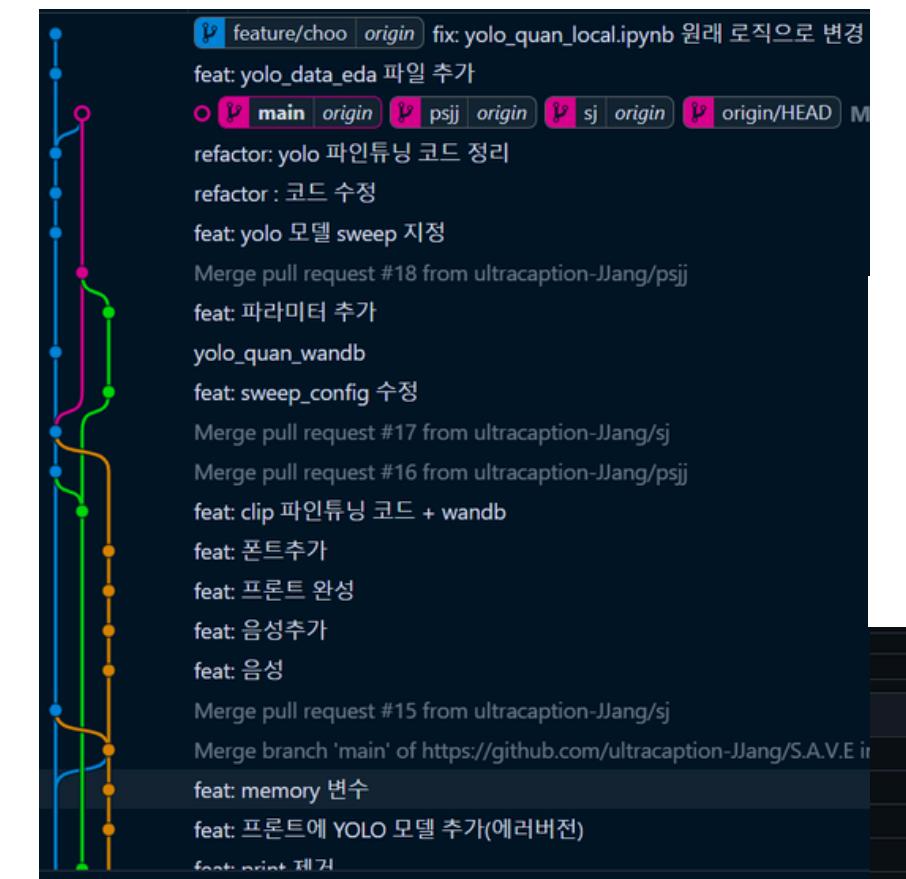
팀원	역할
박성재	모델링과 데이터 전처리 역량을 발전시킬 수 있었던 프로젝트였습니다.
백승제	이번엔 Python를 사용해서 프론트를 구축했지만, 다음에는 Javascript를 사용해서 구축해보고 싶다. 그리고 성능이 아쉬운 YOLO모델을 추후에 좀 더 확인해보고 싶다
이승준	프로젝트의 성공적인 진행을 위해 전체적인 부분을 파악하고 관리하는 것이 매우 중요하다는 것을 느꼈습니다. 각 파트가 어떻게 진행되고 있는지를 명확히 이해해야 프로젝트를 성공적으로 이끌 수 있다는 점을 깨달았습니다. 또한, 프로젝트 매니저로서 발생하는 문제들을 즉각 파악하고 해결하려면 소프트웨어 엔지니어링 역량이 필수적이라는 점을 절감했습니다.
추희정	이번에 맡은 Docker를 이용한 서버 구성은 매우 유익한 경험이었다. 실제 배포 환경에서의 문제를 미리 발견하고 수정할 수 있었으며, 가상 환경과 종속성을 효율적으로 관리하는 방법을 익혔다. 특히, FastAPI 서버를 Docker로 패키징하고, RESTful API로 기능을 제공함으로써 서버의 이식성과 유지보수를 높일 수 있었다. 잘 알지 못했던 서버를 맡게 되었지만 많이 도움을 주신 **승준님**께 무한한 감사를 드린다. YOLO 객체 탐지 모델과 ClipCap 이미지 캡셔닝 모델을 공부하는 부분부터 엄청난 압박감이 느껴졌지만 **성재님**덕에 많이 이해하고 구조를 볼 수 있었다. 그리고 의외의 디자인 능력자 **승제님**께는 실력에 정말 감탄만 나온다고 말씀드리고 싶다.

협업 전략



The Notion workspace displays a '진행 사항' board with several columns (20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31) and rows. Each row represents a task, and each column represents a stage or status. The tasks include:

- 20: 키워드로부터 캡셔닝 생성 전처리 (데이터 전처리, 완료)
- 21: 모델 성능 향상을 위한 다양한 모델 적용 (모델링, 완료)
- 22: postman 사용법 (프론트엔드, 완료)
- 23: 추론 속도 향상을 위한 GPU 사용 여부 결정 (모델링, 완료)
- 24: 대용량 데이터 처리를 위한 클라우드 사용 (데이터 전처리, 완료)
- 25: GCP storage를 활용한 데이터 전처리 pipeline (데이터 전처리, 완료)
- 26: 모델 성과 지표 구현 (모델링, 진행 중)
- 27: stair 전체 데이터로 학습 (모델링, 완료)
- 28: 커스텀 모델링 (모델링, 완료)
- 29: YOLO 파인튜닝 (모델링, 완료)
- 30: 서비스 기획 구체화 (서비스 품질, 완료)
- 31: CLIP + YOLO + cap (모델링, 완료)
- 32: CLIP 파인튜닝 (모델링, 진행 중)
- 33: YOLO의 위험요소 라벨링 조건 (서비스 품질, 완료)
- 34: YOLO 파인튜닝 (모델링, 진행 중)
- 35: 1000개 데이터셋 만들고, wandb sweep (모델링, 진행 중)
- 36: 추론 속도 향상을 위한 양자화 (모델링, 진행 중)



A vertical timeline of GitHub commits and pull requests, illustrating the GitHub flow process:

- feat: yolo_data_eda 파일 추가 (Merge pull request #18 from ultracaption-Jjang/psjj)
- refactor: yolo 파인튜닝 코드 정리 (refactor: 코드 수정)
- feat: yolo 모델 sweep 지정 (feat: yolo 모델 sweep 설정)
- Merge pull request #17 from ultracaption-Jjang/psjj
- feat: 파라미터 추가 (yolo_quan_wandb)
- feat: sweep_config 설정 (feat: sweep_config 설정)
- Merge pull request #16 from ultracaption-Jjang/psjj
- feat: clip 파인튜닝 코드 + wandb (feat: clip 파인튜닝 코드 + wandb)
- feat: 폰트추가 (feat: 폰트추가)
- feat: 프론트 완성 (feat: 프론트 완성)
- feat: 음성추가 (feat: 음성추가)
- feat: 음성 (feat: 음성)
- Merge pull request #15 from ultracaption-Jjang/sj
- Merge branch 'main' of https://github.com/ultracaption-Jjang/S.A.V.E into main
- feat: memory 변수 (feat: memory 변수)
- feat: 프론트에 YOLO 모델 추가(에러버전) (feat: 프론트에 YOLO 모델 추가(에러버전))
- feat: print 제거 (feat: print 제거)

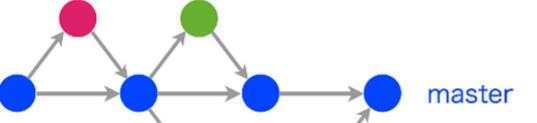


GitHub repository details:

- About**: 실시간 이미지 캡셔닝을 활용한 시작장애인 응성 안내 서비스
- Code**: 40576cb - yesterday (80 Commits)
- Activity**: last week
- Custom properties**: 0 stars, 0 watching, 0 forks
- Releases**: No releases published. Create a new release
- Packages**: No packages published. Publish your first package
- Contributors**: 4
 - SeongjaeP SeongjaePark
 - lseungjun Seungjun Lee
 - bj0530 bj0530
 - heejungdev00 choohjeung



GitHub flow



References

- Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., & Sutskever, I. (2021). Learning transferable visual models from natural language supervision. arXiv preprint arXiv:2103.00020.
<https://doi.org/10.48550/arXiv.2103.00020>
- Mokady, R., Hertz, A., & Bermano, A. H. (2021). CLIPCap: CLIP prefix for image captioning. arXiv preprint arXiv:2111.09734.
<https://doi.org/10.48550/arXiv.2111.09734>
- <https://wikidocs.net/169362>
- <https://github.com/ultralytics/ultralytics>
- <https://github.com/openai/CLIP>
- https://github.com/rmokady/CLIP_prefix_caption
- <https://roboflow.com/>
- <https://www.marqo.ai/course/fine-tuning-clip-models>
- <https://www.kaggle.com/datasets/awsaf49/coco-2017-dataset>
- <https://mlops-for-mle.github.io/tutorial/docs/intro>
- https://github.com/rmokady/CLIP_prefix_caption
- <https://github.com/openai/CLIP>

S.A.V.E

End of Document
Thank You