Contents

Raw Dataset

Segmentation Dataset

Images: 396개, Labels: 396개

구분	형식	비고
라벨 파일	[이미지 파일명].png	이미지 파일명(확장자 제외)과 동일한 이름의 png 파일로 작성됨.
클래스 ID	Integer	0: 배경(인도, 건물 등), 1: 도로, 2: 횡단보도, 3: 캡션

Image

Mask

Image

Mask

Raw Dataset

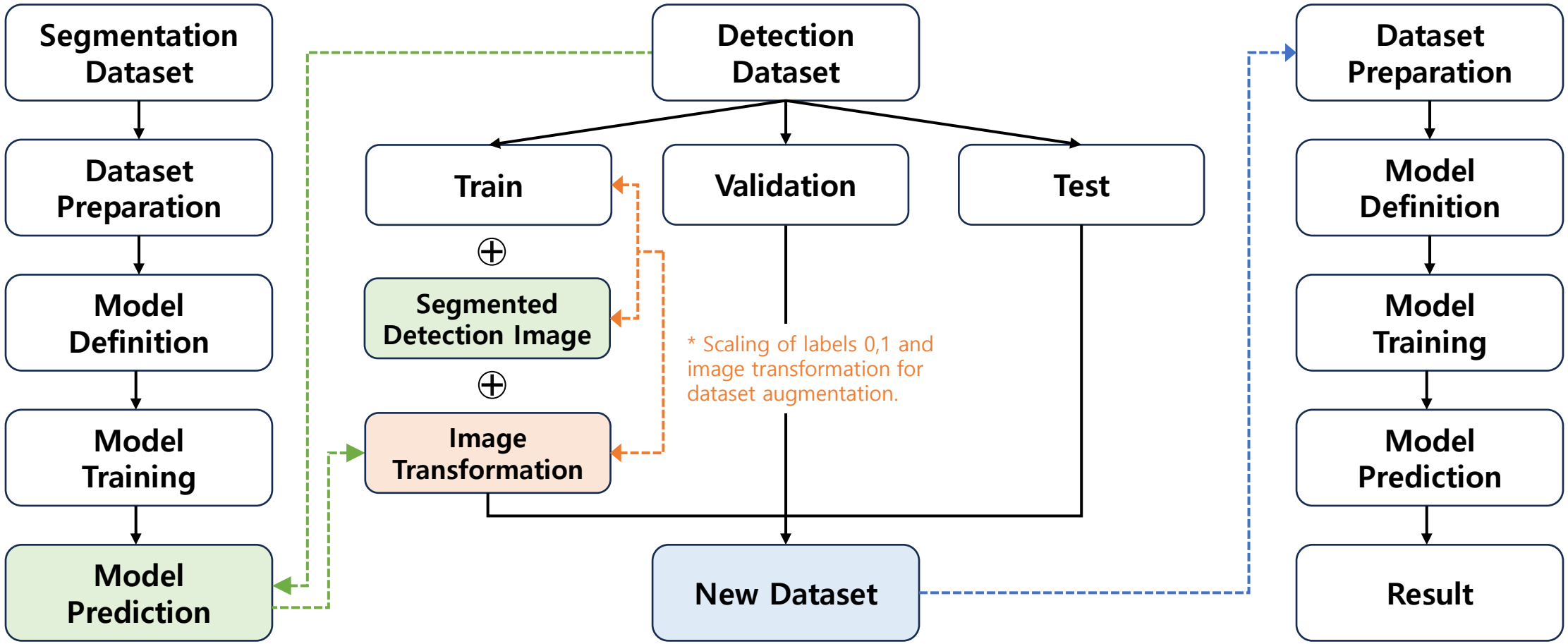
Detection Dataset

1차 데이터셋
Images: 794개, Labels: 794개

2차 데이터셋
Images: 3130개, Labels: 3130개

구분	형식	비고
라벨 파일	[이미지 파일명].txt	이미지 파일명(확장자 제외)과 동일한 이름의 txt 파일로 작성됨.
Bbox 표기	[클래스 ID] [중심 X] [중심 Y] [너비 W] [높이 H]	빈칸으로 구분하며 클래스 ID는 정수임. 바운딩 박스의 중심 좌표와 너비, 높이는 실수 형식임. (예시: 0 0.681862 0.124288 0.008299 0.032051)
클래스 ID	Integer	0: 무단횡단 보행자, 3: 우산을 쓴 무단횡단 보행자, 1: 횡단보도 보행자, 4: 우산을 쓴 횡단보도 보행자, 2: 인도 보행자, 5: 우산을 쓴 인도 보행자
X,Y 좌표	Float	이미지의 너비와 높이로 정규화 된 값으로 최소값은 0, 최대값은 1임. X=바운딩 박스의 중심 X좌표(픽셀)/이미지 너비(픽셀) Y=바운딩 박스의 중심 Y좌표(픽셀)/이미지 높이(픽셀)
너비 W, 높이 H	Float	이미지의 너비와 높이로 정규화 된 값으로 최소값은 0, 최대값은 1임. W=바운딩 박스의 너비(픽셀)/이미지 너비(픽셀) H=바운딩 박스의 높이(픽셀)/이미지 높이(픽셀)

Algorithm


Segmentation Dataset
Dataset Preparation
Model Definition
Model Training
Model Prediction
Detection Dataset
Train
Validation
Test
Segmented Detection Image
Image Transformation
* Scaling of labels 0,1 and image transformation for dataset augmentation.
New Dataset
Dataset Preparation
Model Definition
Model Training
Model Prediction
Result

Segmentation

구분	U-Net	DeepLabv3+	FCB Former	DeepLabv3+ with FFT
Test Loss	0.920464	0.749418	0.588605	0.241387
IoU	0.614666	0.648695	0.703092	0.790671
Dice	0.726380	0.770655	0.814999	0.876615
FPS(gpu)	47.37	49.15	22.70	57.78

DeepLabv3+의 Backbone으로 사용한 ResNet101에 새로운 분기를 추가하여 segmentation 성능을 향상시켰다.

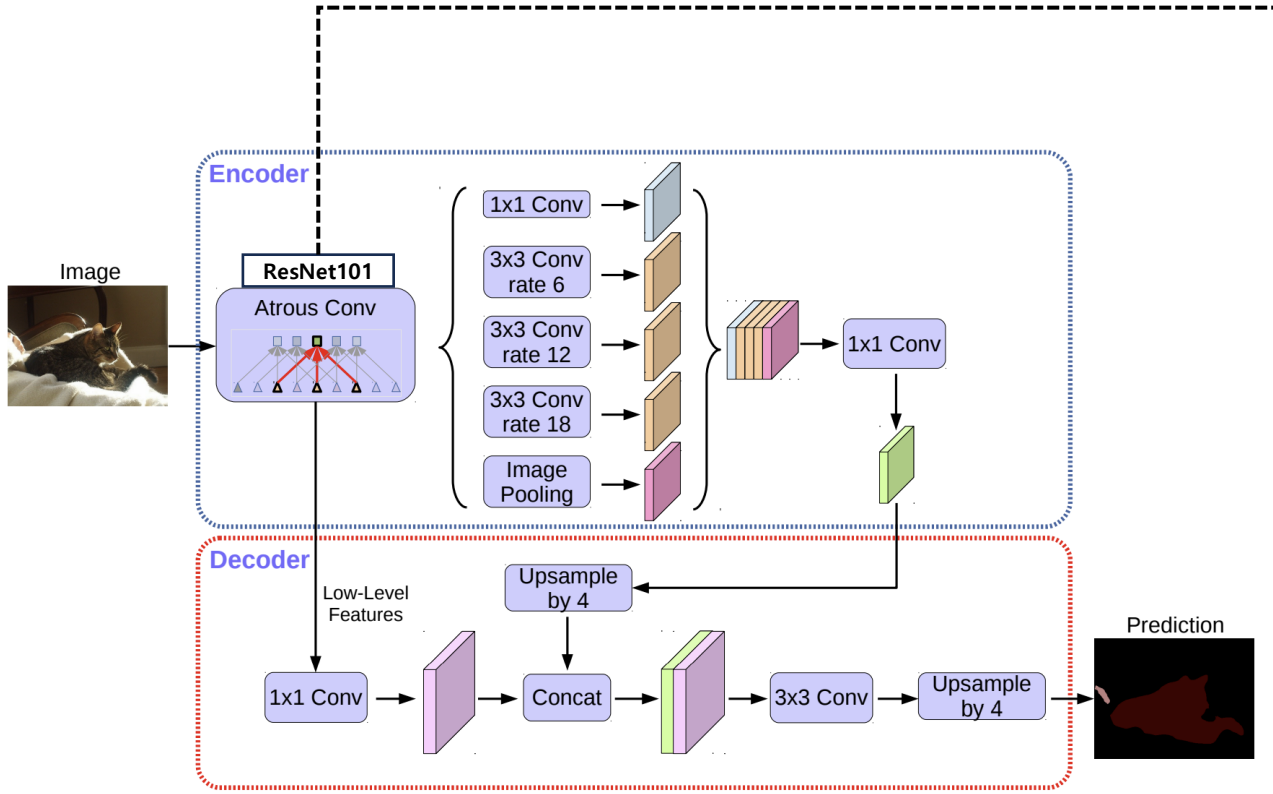
- 작은 객체 검출에 특화된 FAR 모델에서는 푸리에 변환을 이용한 FO, FA 분기를 backbone인 ResNet에 추가하였다.
- 이를 Image Segmentation task에 맞게 변환하여 ResNet101에 추가한 모델을 DeepLabv3+의 backbone으로 사용하였더니 위의 표와 같이 유의미한 성능 향상이 있었다.

도로 내의 무단횡단 보행자와 횡단보도 보행자를 탐지하는 데에 있어서 주변 배경을 제거함으로써 더욱 세밀한 탐지가 가능해질 것이라고 가정하였다. 이를 위해서는 좋은 성능의 Segmentation 모델을 사용하여 보다 정확한 도로 영역 분할이 요구되었다.

제공된 Segmentation Dataset에 transform을 적용한 다음 저장하여, train에 추가해주었다.

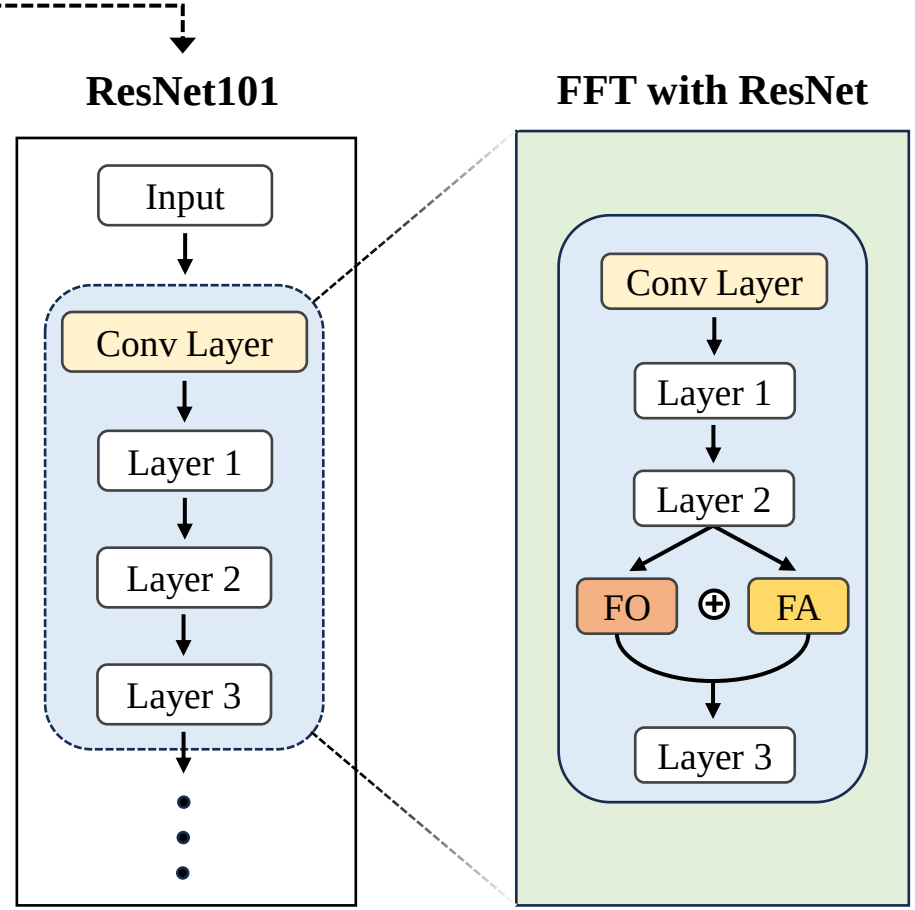
- Train: 316개(Original) + 3160개(Transform) = 3476개, Val: 40개, Test: 40개

Segmentation



DeepLabv3+ Architecture

출처: https://arxiv.org/pdf/1802.02611



Dataset Preparation

구분	1차 Dataset	2차 Dataset	All Dataset
개수	Images: 794개, Labels: 794개	Images: 3130개, Labels: 3130개	Images: 3924개, Label: 3924개

기존의 6개의 클래스를 무단횡단자와 횡단보도 보행자에 대한 클래스 2개로 축소
· Class: 3 → Class: 0 (횡단보도 보행자 통일)
· Class: 4 → Class: 1 (무단횡단 보행자 통일)
· Class: 2 & Class: 5 삭제 (인도 보행자 삭제)

구분	Modified All Dataset
개수	Images: 2918개, Labels: 2918개
클래스 비율	Class 0: 891개, Class 1: 2027개

인도 보행자만 존재하는 이미지(background): 1006개
· Modified Dataset 개수: 3924개 → 2918개

이후에 Detection 이미지에 대해 도로 영역을 분할하여
위의 마지막 이미지를 Detection Dataset의 train에 추가

Dataset Preparation

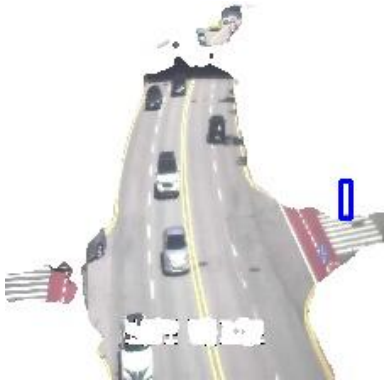
구분	Segmented Detection Dataset	Modified Segmented Detection Dataset
개수	Images: 2918개, Labels: 2918개	Images: 2422개, Labels: 2422개
클래스 비율	Class 0: 891개, Class 1: 2027개	Class 0: 813개, Class 1: 1608개

Detection 이미지에 대해 도로 영역 분할

· DeepLabv3+ with FFT 모델로 예측
· 도로 영역을 분할 하면서 바운딩 박스 내의 보행자가 사라지는 경우가 있어 새로 라벨링을 진행
· Modified Segmented Detection Dataset 개수: 2918개 → 2422개

Ex)

Original Image



Segmented Detection Image

예시와 같이 횡단보도 내에 사람이 있지만 평면적으로 보면 인도에 다수 넘어가있어 도로 영역을 분할했을 때, 객체가 사라지는 것을 볼 수 있다.
· 즉, background 이미지가 되어 버린다.

Dataset Preparation

구분	Train	Validation	Test
개수	Images: 2628개(All) + 2422개(Segmented) = 5050개 Labels: 5050개	Images: 232개, Labels: 232개	Images: 58개, Labels: 58개
클래스 비율	Class 0: 747 + 813 = 1560개, Class 1: 1882 + 1608 = 3490개	Class 0: 116개, Class 1: 116개	Class 0: 29개, Class 1: 29개

Modified All Dataset을 train/val/test로 분할

· 우선적으로, val과 test에 클래스 비율을 맞춰 분배하고 나머지를 train에 추가

· Modified Segmented Detection Dataset을 train에 추가

구분	Train
개수	Images: 5050개(Train) + 16360개(Transform) = 21390개 Labels: 21390개
클래스 비율	Class 0: 1560 + 9360(1560*6) = 10920개, Class 1: 3490 + 6980(3490*2) = 10470개

원활한 학습을 위해, 클래스 0과 1에 대해 각각 transform을 적용하여 augmentation을 해줌으로써 비율을 맞춘다.

Ex)

Original Image

Image Transformation

Object Detection

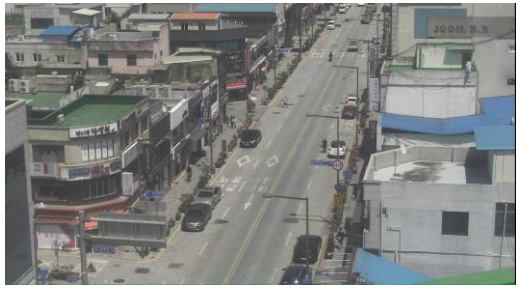
구분	YOLO v8-n	YOLO v8-s	YOLO v8-m	YOLO v11–s	YOLO v11-m
Precision	0.715	0.765	0.818	0.722	0.802
Recall	0.659	0.741	0.785	0.395	0.377
mAP50	0.76	0.827	0.826	0.549	0.629
mAP50-95	0.468	0.518	0.528	0.328	0.423

현재 Object Detection task에서 가장 많이 사용되고 있는 YOLO 모델을 사용

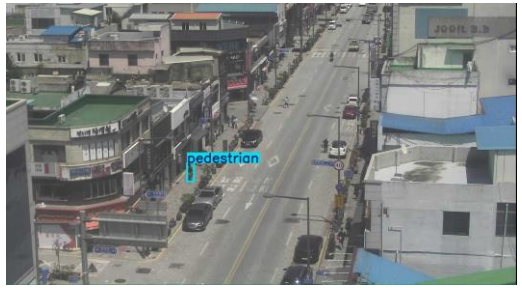
· 빠른 비교를 위해 augmentation을 통해 class 비율을 맞춘 Original Image + Segmented Image 데이터셋 개수를 3000개로 통일하여 학습에 사용

위의 표를 보면, mAP50을 제외한 모든 경우에서 YOLO v8-m이 성능이 가장 좋은 것을 확인할 수 있다.

Ex)



Test Image



YOLO v8-s

YOLO v8-m

Dataset 변형에 따른 성능 변화

구분	Original Image	Original Image + Crop Image	Original Image + Segmented Image	Original Image + Crop Image + Segmented Image
Precision	0.779	0.561	<u>0.818</u>	0.708
Recall	0.436	0.502	<u>0.785</u>	0.472
mAP50	0.586	0.566	<u>0.826</u>	0.626
mAP50-95	0.367	0.334	<u>0.528</u>	0.474

빠른 비교를 위해 augmentation을 통해 class 비율을 맞춘 데이터셋의 개수를 3000개로 통일하여 학습에 사용
YOLO v8-m 모델을 사용하여 학습을 진행하였다.
· 위의 표를 보면, Original Image 와 Segmented Image를 같이 사용한 경우에 성능이 가장 좋은 것을 확인할 수 있다.

Ex)

Original Image

Crop Image

작은 객체(사람)를 더욱 잘 감지할 수 있도록 객체 부분만 확대하여 crop한 이미지를 사용한다면 성능향상이 있을 것이라 가정하였다.
· <u>결국에는 성능향상에 큰 도움이 되지 않아 사용하지 않았다.</u>

Conclusion

Model	Dataset	Precision	Recall	mAP50	mAP50-95
YOLO v8-m	Original Image + Segmented Image	0.973	0.851	0.907	0.618

위의 표는 비교 실험을 통해 최종적으로 채택한 Original Image + Segmented Image 데이터셋을 class 비율을 맞추기 위해 Transformed Image를 추가하여 개수를 21390개로 augmentation한 뒤, YOLO v8-m으로 학습한 결과이다.

· 리소스 문제로 데이터셋을 더 늘리기에는 한계가 있다는 문제점이 있었다.

Test Image

Predict Image