Car object detection

Hubert Giza
Przemysław Roman
Mateusz Powęska

April 2022

1 Theoretical introduction

1.1 Problem definition

Our data set consists of 1000 images representing cars on the road or empty road and can be found here. There could be one or more cars present. Along with the images, we were provided with bounding boxes, which marked pixels, where cars were detected. Then our objective was defined as creation of masks of zeros and ones, where ones represents pixels marked as detected car object.

1.2 Model definition

Using standard approach, we have implemented neural network with U-net structure. Our building block consisted of 3 layers: Convolutionl layer, Relu activation layer and Convolutional layer. Next, we defined Encoder and Decoder. In Encoder, we have started with 3 channels and other dimensions as input width and height. We have defined 4 levels of Encoder and Decoder, where number of channels was being increased (or decreased respectively for Decoder) with each deeper level. Number of channels at the bottom level of Encoder was equal to 64. Also, we were storing information about each Encoder's level output to pass it forward to respective Decoder's level. This way, Decoder had it's own information received by transposing input and Encoder's output from given level. After the last block of Decoder, we had another Convolutional layer which at last calculated map of zeros and ones as output.

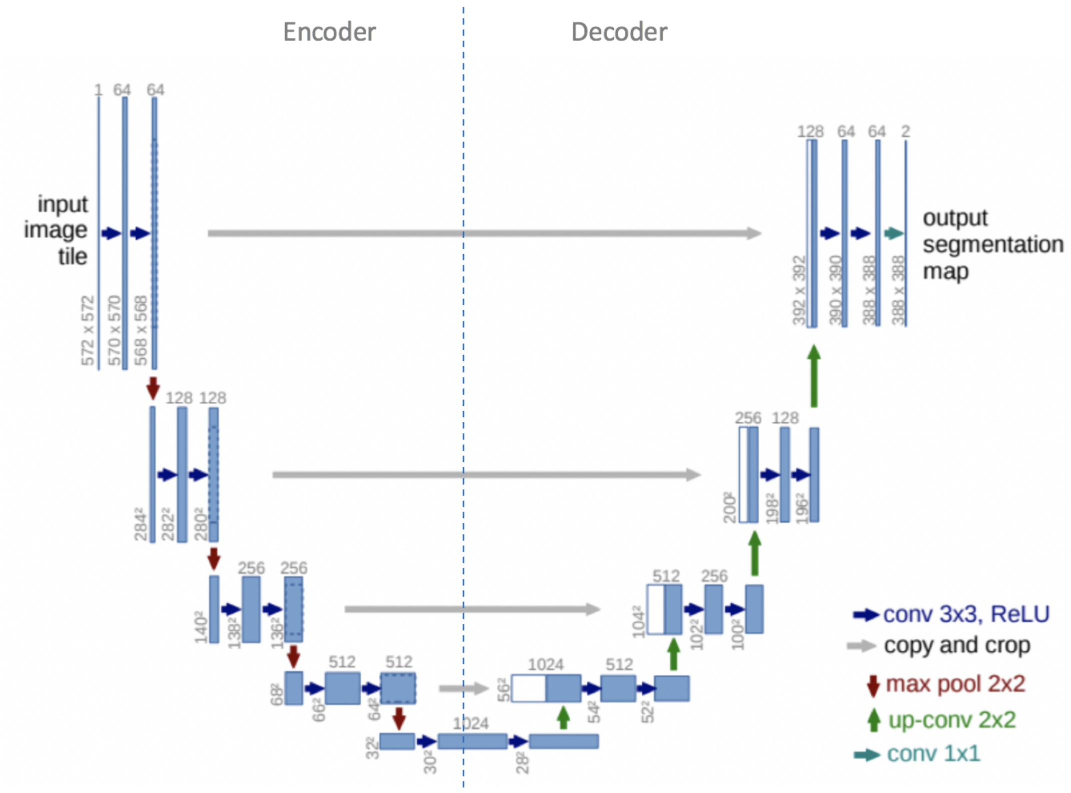

Figure 1: Image representing typical structure of U-net

1.3 Training

We have trained our model for 40 epochs using 85% of the data set. We did not base on the loss function graph, so we could check performance of our model on the same data set.

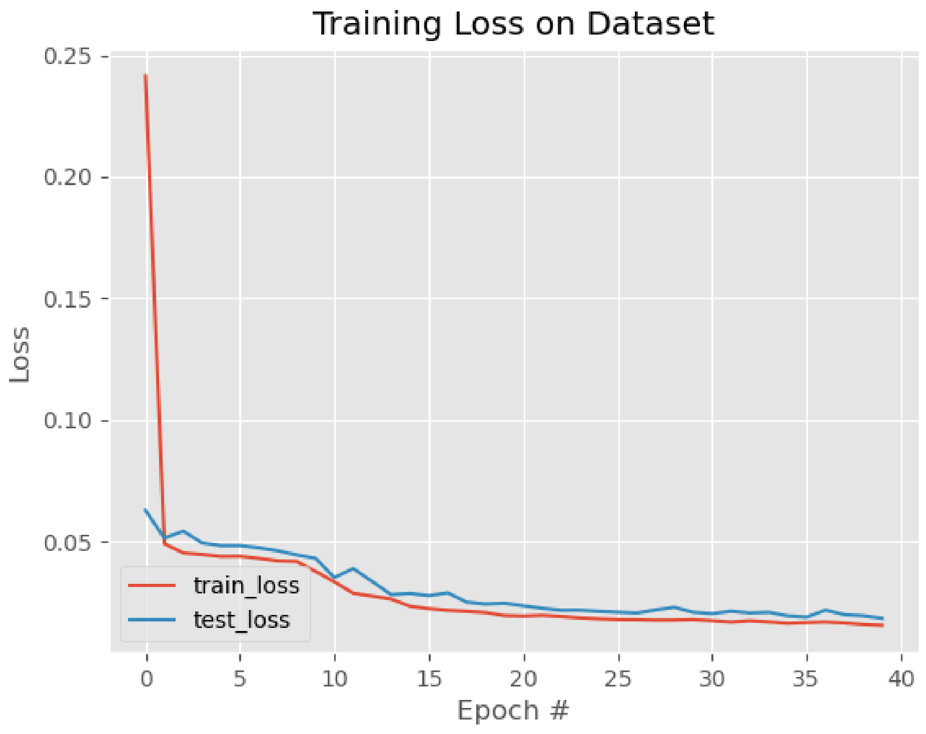

Figure 2: Loss function graph

1.4 Performance metric

To check performance of our model, we have used Jaccard Index, which is intersection over union. It is calculated with given equation:

$$IoU = \frac{Intersection}{Union}$$

Using thing metric, we have received 82% accuracy on our test data set.

2 Running the program

2.1 Install requirements

pip install -r requirements.txt

2.2 Download the dataset

Do it manually from the link

2.3 Adjust parameters in config.py

Make sure that DATASET_PATH points to the location containing the downloaded dataset
The default filestructure:

```
DATASET_PATH
├── test
│   ├── images
│   └── masks
├── train
│   ├── images
│   └── masks
└── train_bounding_boxes.csv
```

2.4 Run data_preparation.py

python data_preparation.py

2.5 Run train.py

python train.py

2.6 Run predict.py

python predict.py

3 Demonstration

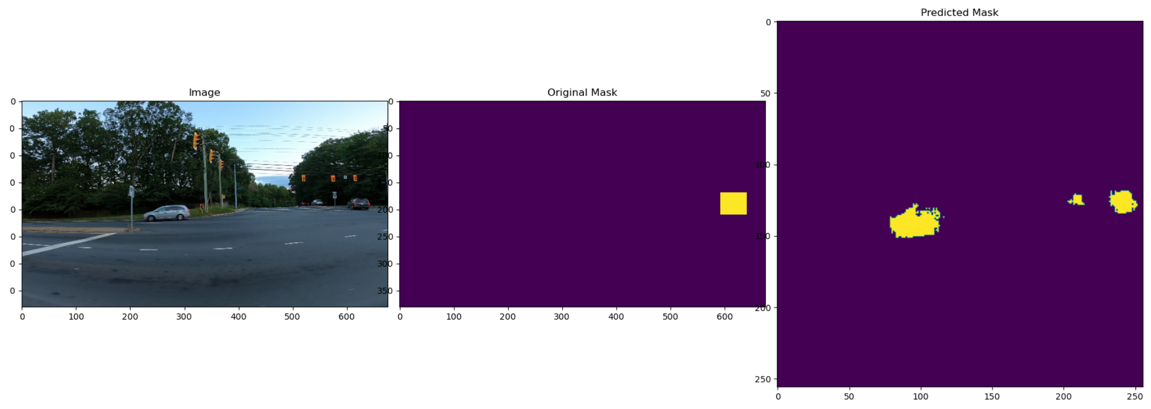

Figure 3: Correct detection of more car objects than provided

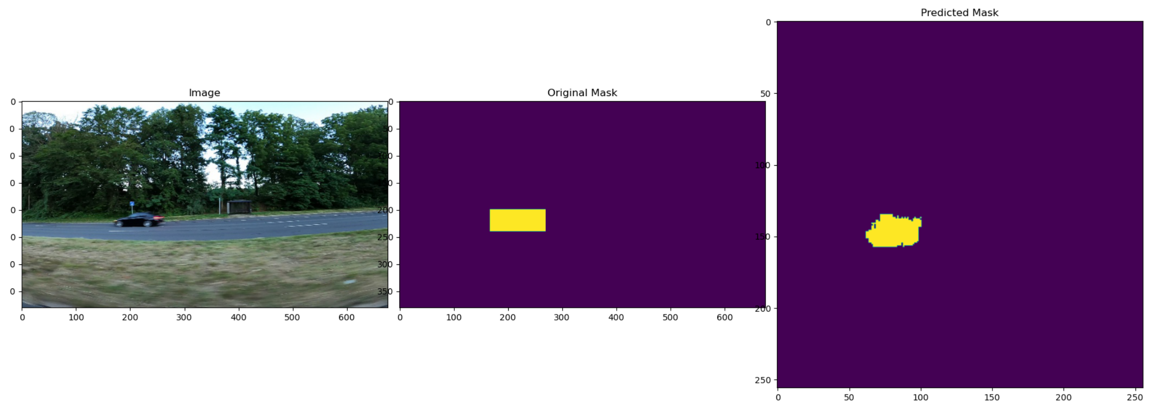

Figure 4: Correct detection of one car object

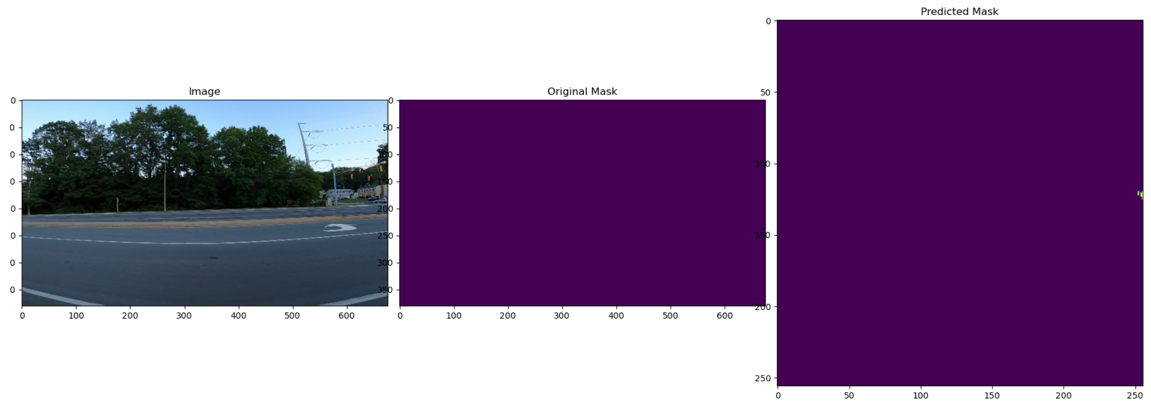

Figure 5: Correct detection of a small car object

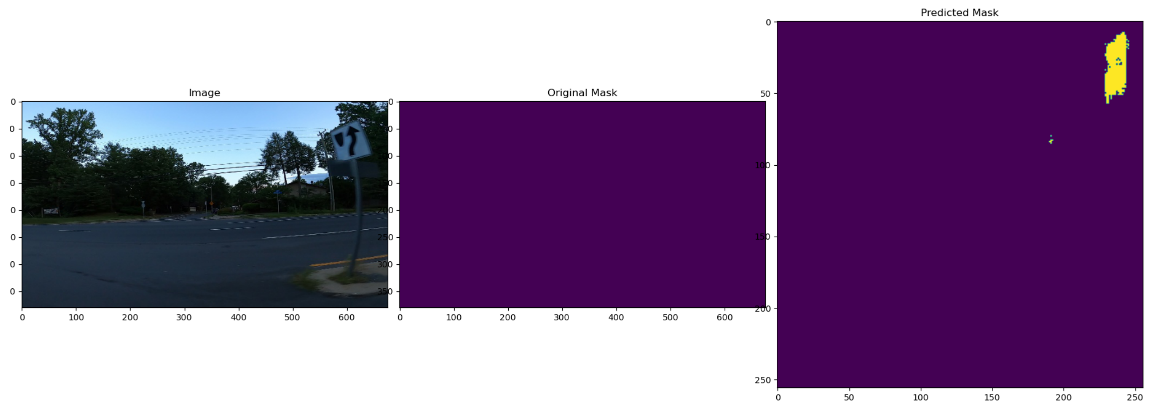

Figure 6: Mistaken a road sign for a car object

4 Conclusion

Our data set was not perfect. It did not have ideal label boxes (in some cases it didn't cover a car at all, even though it was present on the image) and there wasn't so many training samples provided. Still, it was possible to achieve 82% of accuracy without any additional bells and whistles, using only basic U-net model. It is still possible to achieve higher performance using additional training techniques and heuristics.