Establishing a training set of San Francisco street view photos for the image segmentation model on biking environment

Yingjia Feng, Offer Grembek

Table of Content

Abstract

This research focuses on retraining the SegNet image segmentation model to make it more suitable for San Francisco street views by building a training dataset based on San Francisco Google Street View.

This research first tested the original SegNet image segmentation model on several street view photos of San Francisco, yet the segmentation result is not ideal. After that I figured out the most efficient way to improve the segmentation model is to train it on the local street view photos instead of the official training set given by Cambridge University. Therefore, I decided to build up my own training set by gathering 405 street view photos from Google Street View.

I labeled each photo into 9 classes by using LabelMe. The labelling process is to add polygons onto every photo and classify these polygons into different classes: lane divider(continuous), lane divider(discontinuous), sidewalk, vehicle, bus stop, yellow lane divider, bike mark, separate bike lane and green bike mark. Other parts of the street view photo are not considered and not labelled in this training set and thus all became background.

After labelling all the images, I transformed them into PNG files, which can be used by model training. In order to have uniform size, I resized the PNG files and the original images into 480*360 pixels and transform and normalize it into 1-dimensional grayscale images.

Finally, the combination of 405 original street view images and the corresponding label mask files become the training set of the new model. I use the training set to retrain the image segmentation model. The testing result of this new model is relatively better than the official model and prove the potential feasibility of using street view images to automatically evaluate the biking environment of the city.

Key words: Image segmentation, labeling, biking environment, SegNet

Introduction

Biking environment is the key to promote biking and reduce GHG emission. A precise and efficient evaluation system for biking environment can be very supportive to improve the biking safety. Efficiently recognize the accessibility of biking environment of the street can help the government precisely improve the weak point of the cities biking environment and make the cities more biking friendly.

Traditionally, the evaluation of the street biking facilities needs intense field observation and data collection, which is fairly time consuming and cost intensive. Recently, the widely collection of street views on Google inspired the researchers the tremendous information given by street views. Many researches focused on using the street views to analyzing the micro scale city environment, such as neighborhood or sidewalk qualities, are conducted and the exploration of automatic data collection and evaluation on cities environment and facilities are getting more and more interesting.

This research focus on exploring the feasibility of using Google Street View to evaluate the biking environment of the city of San Francisco. Image segmentation is the core method used in the research to get the facilities information from the street views. I chose SegNet[1] as the model for image segmentation for the reason that it is an image segmentation model developed by Cambridge University based on deep neural network and developed especially for road scene images. Yet the model trained by the official dataset performed pretty bad on the street views of San Francisco. The key reason caused the bad performance is most likely to be the dataset given by the developers are sampled in Britain, whose street view might be quite different from San Francisco. Therefore, this research intends to build up the specialized training set based on San Francisco to retrain the image segmentation model in order to make the model more suitable for San Francisco.

Literature review

- **How can street view images help the evaluation of various aspects of city environment?**

Some research focused on evaluating the efficiency and agreement of using street view “virtual audit” the road facilities compared with the field audit. Recently, Madeleine’s research [2] compared two ways of auditing on pedestrian streetscape. The research compared the virtual method and field method in time consuming and agreement. It turned out that percentage agreement between those two methods are 80%. Thus, the research concluded that virtual audit is indeed a reliable way of auditing the micro environment of a city.

Currently, the usage of street views to evaluate cities facilities are explored in various fields. Such as assessing the road accessibility for disabled people, or evaluate the safety level of the neighborhood. Hara and Froehlich [3] use a combination of crowdsourcing, computer vision and machine learning to develop a scalable data collection method on Google Street View.

The most classical usage of street view images is in the platform called “Streetscore” [4] built by MIT media lab. This research establishes a machine learning algorithm that can predict the safety score of a street by inputting the street view photo. Yet the core method used by this research is not image segmentation. They converted the images into multiple features that can capture the shape, color and texture feature of the image and then used machine learning to build up a model that can relate those features with the safety score of the streets. Also, their tremendous dataset of 3000 street views from New York and Boston contributed a lot to the precision of prediction.

Data Collection

There are two data sources for this research, the first one was the official dataset provided by Cambridge University. The model trained on this set turned out to be not good enough for image segmentation of San Francisco.

The images used for building up my own training set is from Google Street view in San Francisco area. I manually collected 405 images for the training set to prepare for the labelling and training.

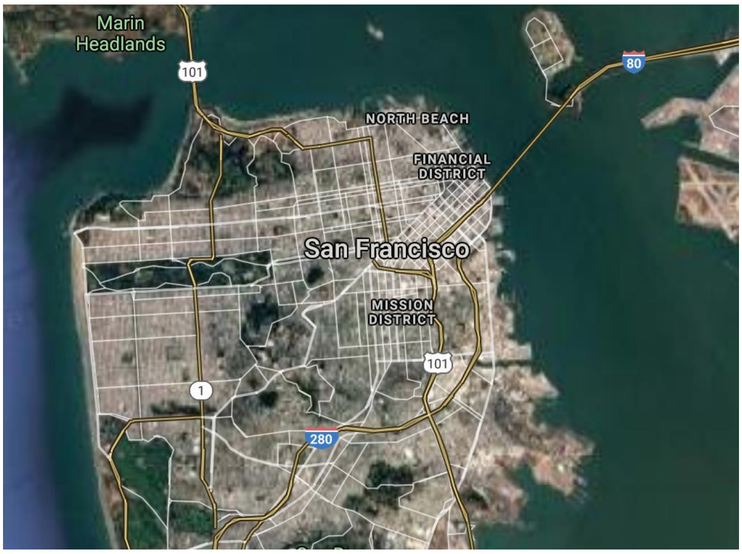


Figure 1: Street view collection area

The street views are manually collected on Google Map in the area of San Francisco. The baseline of collecting the street views are the streets should be clearly displayed on the image and the camera angle should head towards the street. And there is no extremely discontinuous object in the image.

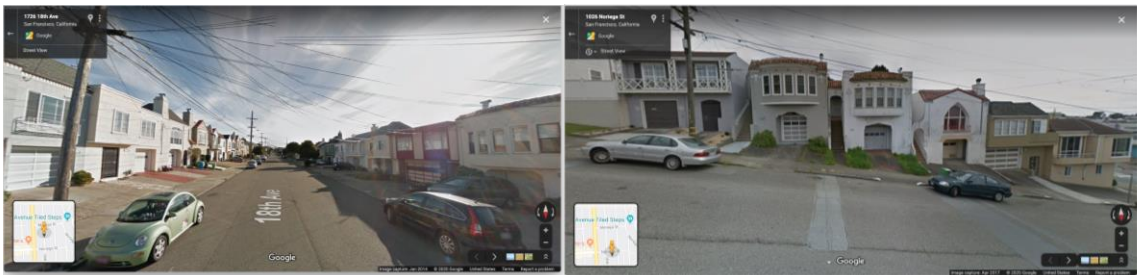


Figure 2: Street views collection example

All the street views are only collected in the approaches, not intersection. Because the facilities to be considered in the intersection area are very different from the approaches, this research only focuses on the approaches.

Labeling

The labelling is manually conducted by myself through the software called LabelMe. Figure 3 display the platform of LabelMe. Labeling is actually adding polygons on to the images and classifying the polygons into different classes. The file generated by LabelMe is a json file for each original image. To generate the mask file for training set, I still need to transform the json file into a certain form.

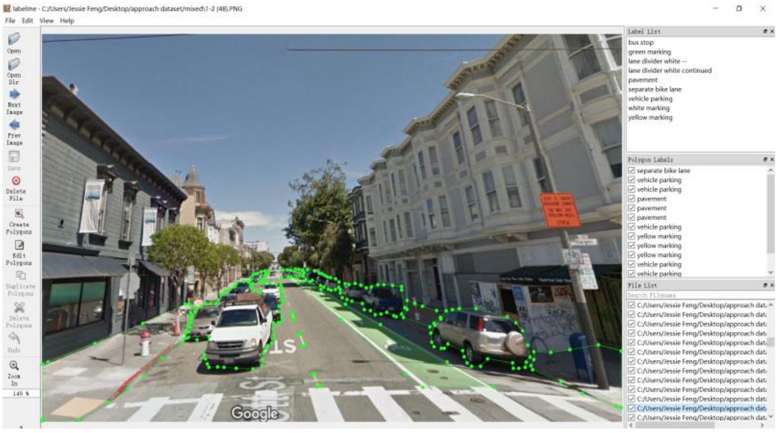

labelme - C:/Users/Jessie Feng/Desktop/approach dataset/mixed\1-2 (48).PNG
File
Edit
View
Help
Open
Open Dir
Next Image
Prev Image
Save
Delete File
Create Polygons
Edit Polygons
Duplicate Polygons
Delete Polygons
Undo
Zoom In
149 %
Label List
bus stop
green marking
lane divider white --
lane divider white continued
pavement
separate bike lane
vehicle parking
white marking
yellow marking
Polygon Labels
☑ separate bike lane
☑ vehicle parking
☑ vehicle parking
☑ pavement
☑ pavement
☑ pavement
☑ vehicle parking
☑ yellow marking
☑ yellow marking
☑ yellow marking
☑ vehicle parking
☑ vehicle parking
File List
Search Filename
☑ C:/Users/Jessie Feng/Desktop/approach dat
☑ C:/Users/Jessie Feng/Desktop/approach dat
☑ C:/Users/Jessie Feng/Desktop/approach dat
☑ C:/Users/Jessie Feng/Desktop/approach dat
☑ C:/Users/Jessie Feng/Desktop/approach dat
☑ C:/Users/Jessie Feng/Desktop/approach dat
☑ C:/Users/Jessie Feng/Desktop/approach dat
☑ C:/Users/Jessie Feng/Desktop/approach dat
☑ C:/Users/Jessie Feng/Desktop/approach dat
☑ C:/Users/Jessie Feng/Desktop/approach dat
☑ C:/Users/Jessie Feng/Desktop/approach dat
☑ C:/Users/Jessie Feng/Desktop/approach dat
☑ C:/Users/Jessie Feng/Desktop/approach dat
Google

Figure 3: LabelMe

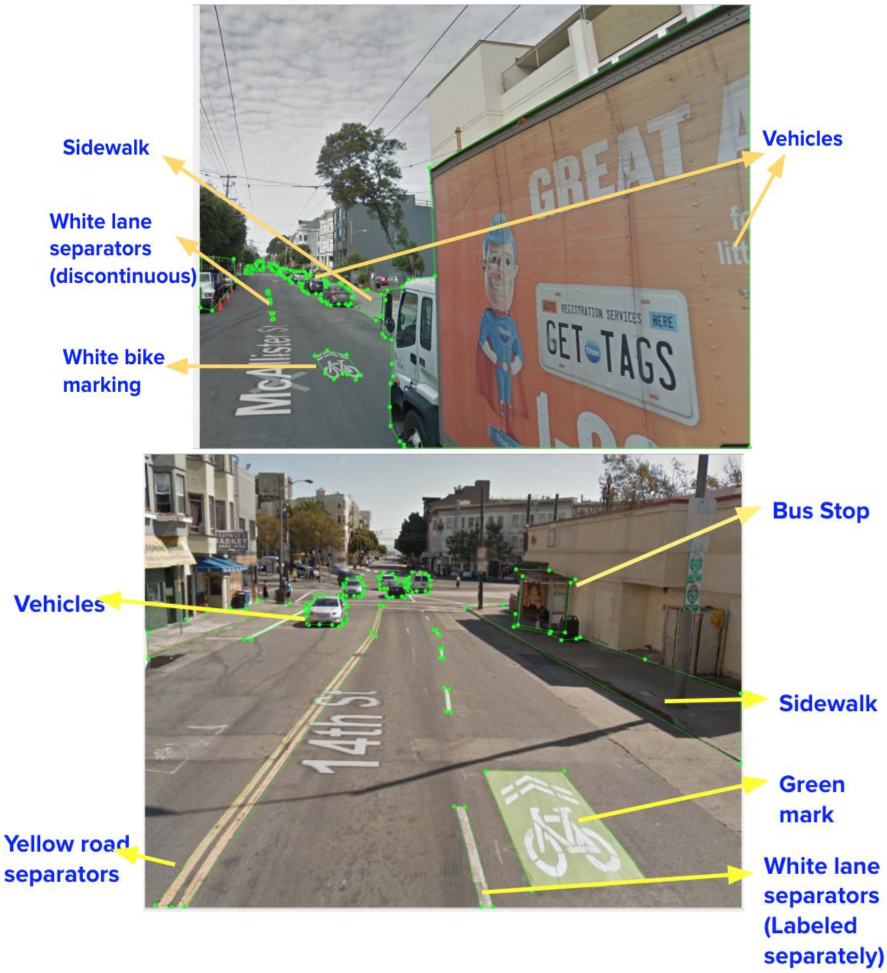


Figure 4: Labeling methodology

- **Vehicles**
  - Label all the vehicles on the street instead of only label the parking ones. (To avoid making confusion for training model)
  - Make the polygon as close to the vehicle as possible
- **Lane Dividers**
  - Label all the lane dividers into 3 categories: Yellow dividers / White continuous dividers/ white discontinuous dividers
  - Label the discontinuous dividers separately along with their shape
- **Bike Lane Markings**
  - Only label the bike lane markings that are not in bike lane (Only mark the shared lanes bike marking)
  - Label them into two categories: Green markings / White markings
- **Bus Stops**
  - Only be able to label the bus stops who has physical stations
- **Sidewalk**
  - Label as much as possible of sidewalk on each photo so that we can use it as a reference location to separate the street spatially.
- **Bike Lanes**
  - Label the bike lane as a whole rectangle with the bike marking within it.
  - Label the bike lane no matter it is painted green or not

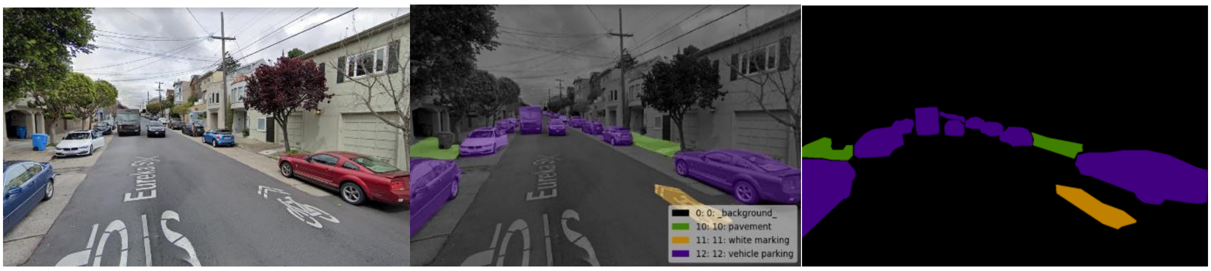


Figure 5: Labelled file created by “LabelMe”

Data cleaning and preparation

- Generate mask files from the json files and resize to 480*360

After labelling out all the polygons on LabelMe, I got 405 json files. Yet json file cannot be used to train the image segmentation model. I still need to transform them into PNG files. After transform the json files into PNG files, resizing them to 480*360 is a key step. Because the model has to take in a uniform size of images for training and testing.

- Generate the mask file

The mask file asked by SegNet model is a 1-dimensional grayscale file, in which the grayscale of each pixel indicate the class to which it belongs. Yet the mask file generated from the json file are in RGB 3-dimensional images. In our case, I first convert the label PNG files into grayscale PNG files. Then normalize the grayscale to the range of 0-9. Thus, each pixel has a grayscale of integer from 0 to 9 to indicate its class.

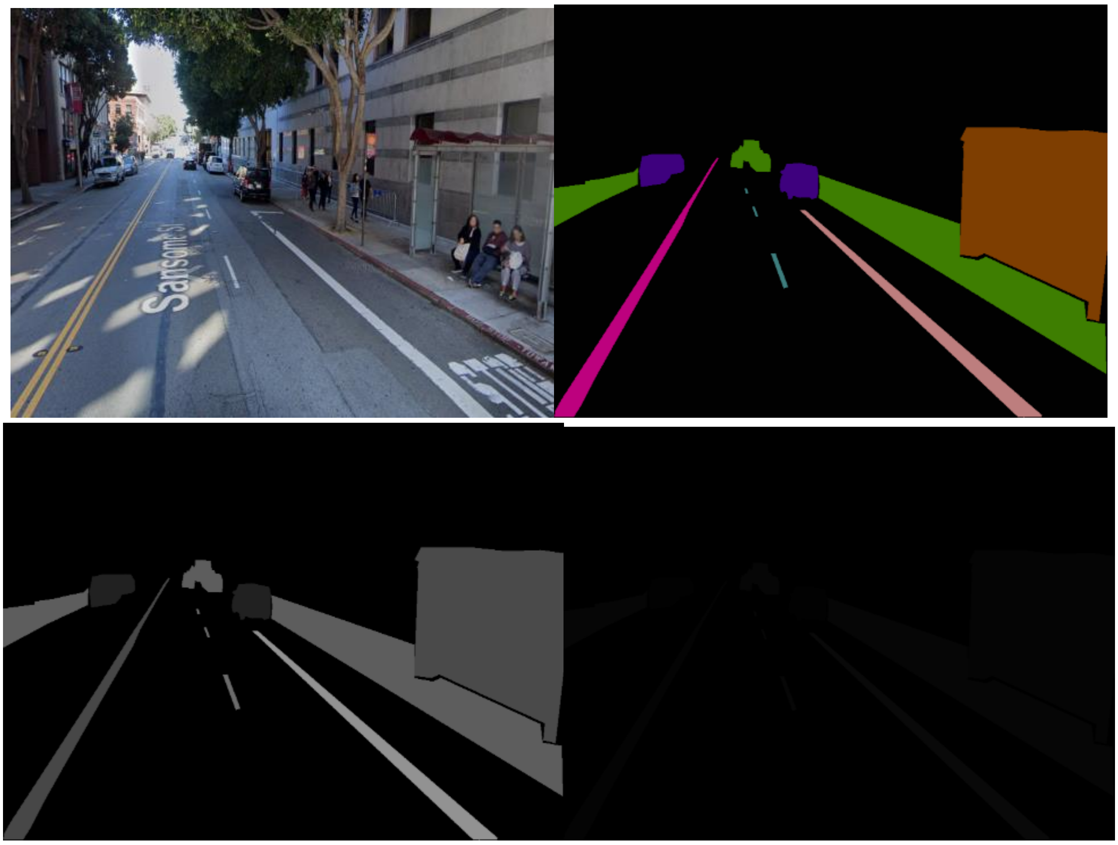


Figure 6: Generation process of mask files

The last image in the above figures is the final form of the mask file ready for model training. It seems like the image is all black, but in fact, if you look carefully, it is not all black, it actually has different shades of gray, it just getting darker than the left one because of the normalization changed the grayscale from 0-255 to 0-10. 0 is for all the background.

Model training and results

1. Modify the network parameters

- Change the classes number from 11 to 10 (9 labeled classes and background)

The original training model are designed to classify the images into 11 classes, yet in my own dataset, the images are labeled into 10 classes (9 labeled classes and background), therefore, in order to change the number of predicted classes, I changed the

- Model 1: Change the class weight in the loss function to the frequency of each class

In the original model, the class weight of the loss function is calculated based on the class frequency. Thus, I decided to first set the weight of each class on class frequency as well.

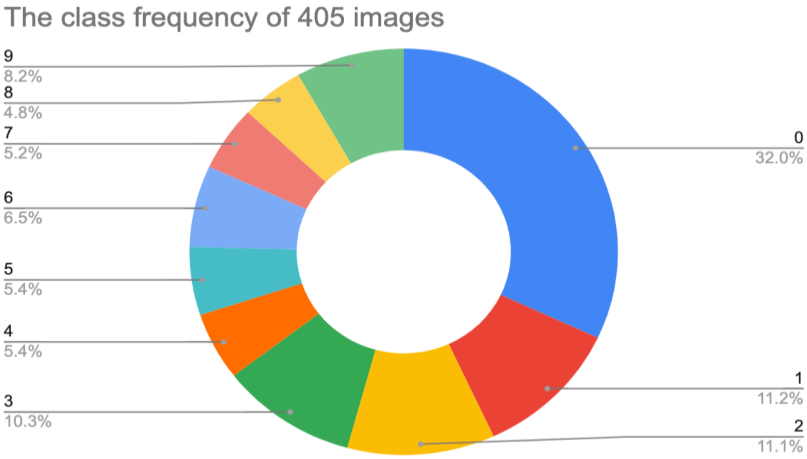


Figure 7: Class frequency

As we can see from the figure, 0: background has the highest-class frequency. The second and highest frequent classes are vehicle. Yet we can find out from the pie chart that the class frequency is not very balanced. Therefore, the model trained with class weight based on class frequency turned out to be extremely bad.

- Model 2: Arbitrarily using high class weight on the low frequency classes.

After getting an extremely bad model, I arbitrarily increase the class weight of the low-frequent classes. After 30000 iterations, the model turned out to be relatively good. Therefore, I chose this model to give to the final test on a set of new street views that are not in the training set.

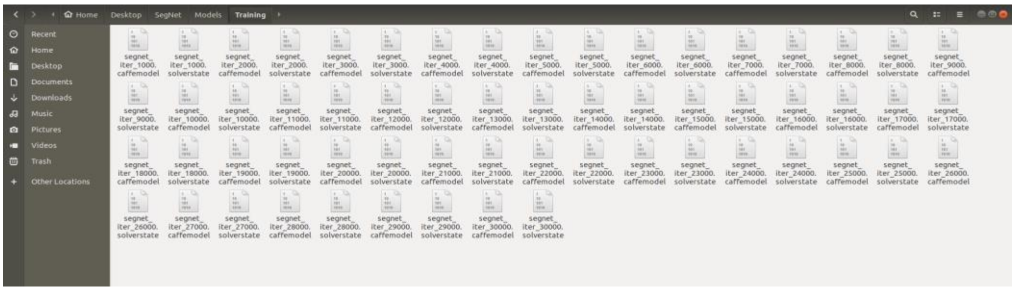
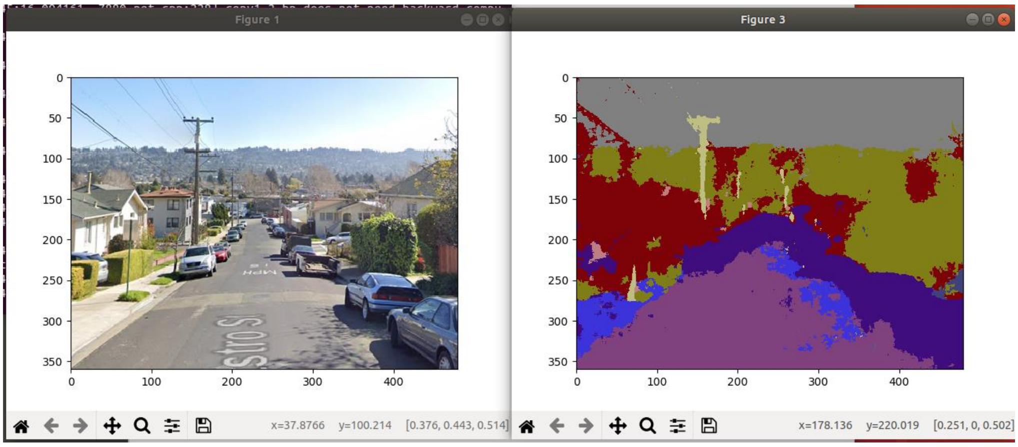

Home
Desktop
SegNet
Models
Training
Recent
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
Other Locations
segnet_iter_1000.caffemodel
segnet_iter_1000.solverstate
segnet_iter_2000.caffemodel
segnet_iter_2000.solverstate
segnet_iter_3000.caffemodel
segnet_iter_3000.solverstate
segnet_iter_4000.caffemodel
segnet_iter_4000.solverstate
segnet_iter_5000.caffemodel
segnet_iter_5000.solverstate
segnet_iter_6000.caffemodel
segnet_iter_6000.solverstate
segnet_iter_7000.caffemodel
segnet_iter_7000.solverstate
segnet_iter_8000.caffemodel
segnet_iter_8000.solverstate
segnet_iter_9000.caffemodel
segnet_iter_9000.solverstate
segnet_iter_10000.caffemodel
segnet_iter_10000.solverstate
segnet_iter_11000.caffemodel
segnet_iter_11000.solverstate
segnet_iter_12000.caffemodel
segnet_iter_12000.solverstate
segnet_iter_13000.caffemodel
segnet_iter_13000.solverstate
segnet_iter_14000.caffemodel
segnet_iter_14000.solverstate
segnet_iter_15000.caffemodel
segnet_iter_15000.solverstate
segnet_iter_16000.caffemodel
segnet_iter_16000.solverstate
segnet_iter_17000.caffemodel
segnet_iter_17000.solverstate
segnet_iter_18000.caffemodel
segnet_iter_18000.solverstate
segnet_iter_19000.caffemodel
segnet_iter_19000.solverstate
segnet_iter_20000.caffemodel
segnet_iter_20000.solverstate
segnet_iter_21000.caffemodel
segnet_iter_21000.solverstate
segnet_iter_22000.caffemodel
segnet_iter_22000.solverstate
segnet_iter_23000.caffemodel
segnet_iter_23000.solverstate
segnet_iter_24000.caffemodel
segnet_iter_24000.solverstate
segnet_iter_25000.caffemodel
segnet_iter_25000.solverstate
segnet_iter_26000.caffemodel
segnet_iter_26000.solverstate
segnet_iter_27000.caffemodel
segnet_iter_27000.solverstate
segnet_iter_28000.caffemodel
segnet_iter_28000.solverstate
segnet_iter_29000.caffemodel
segnet_iter_29000.solverstate
segnet_iter_30000.caffemodel
segnet_iter_30000.solverstate

Figure 8 : Model training snapshots



x=37.8766 y=100.214 [0.376, 0.443, 0.514]

x=178.136 y=220.019 [0.251, 0, 0.502]

Figure 9: Testing result given by official model

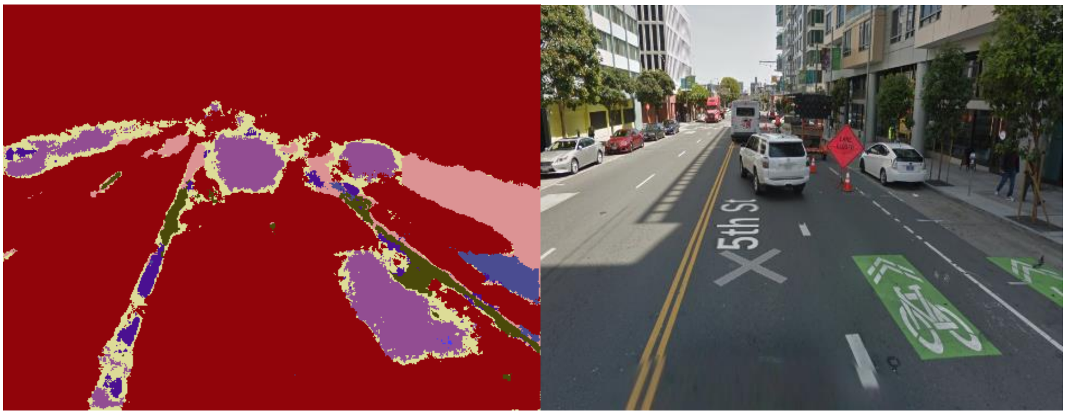
2. Training and results

In my own model, I first use the class weight based on the class frequency to train the model, yet because the two most frequent classes of all the classes are background and vehicle. Therefore, they have the largest class weights. This caused the final model turned out to have the largest class accuracy on recognizing background and vehicle yet has almost no ability of recognizing other classes.

The final model turned out to be able to classify the following classes:

Table 1: Color reference

🟪	Vehicle
🟪	Sidewalk
🟥	Yellow lane divider
🟦	Separate bike lane
🟩	White lane divider
🟥	Background



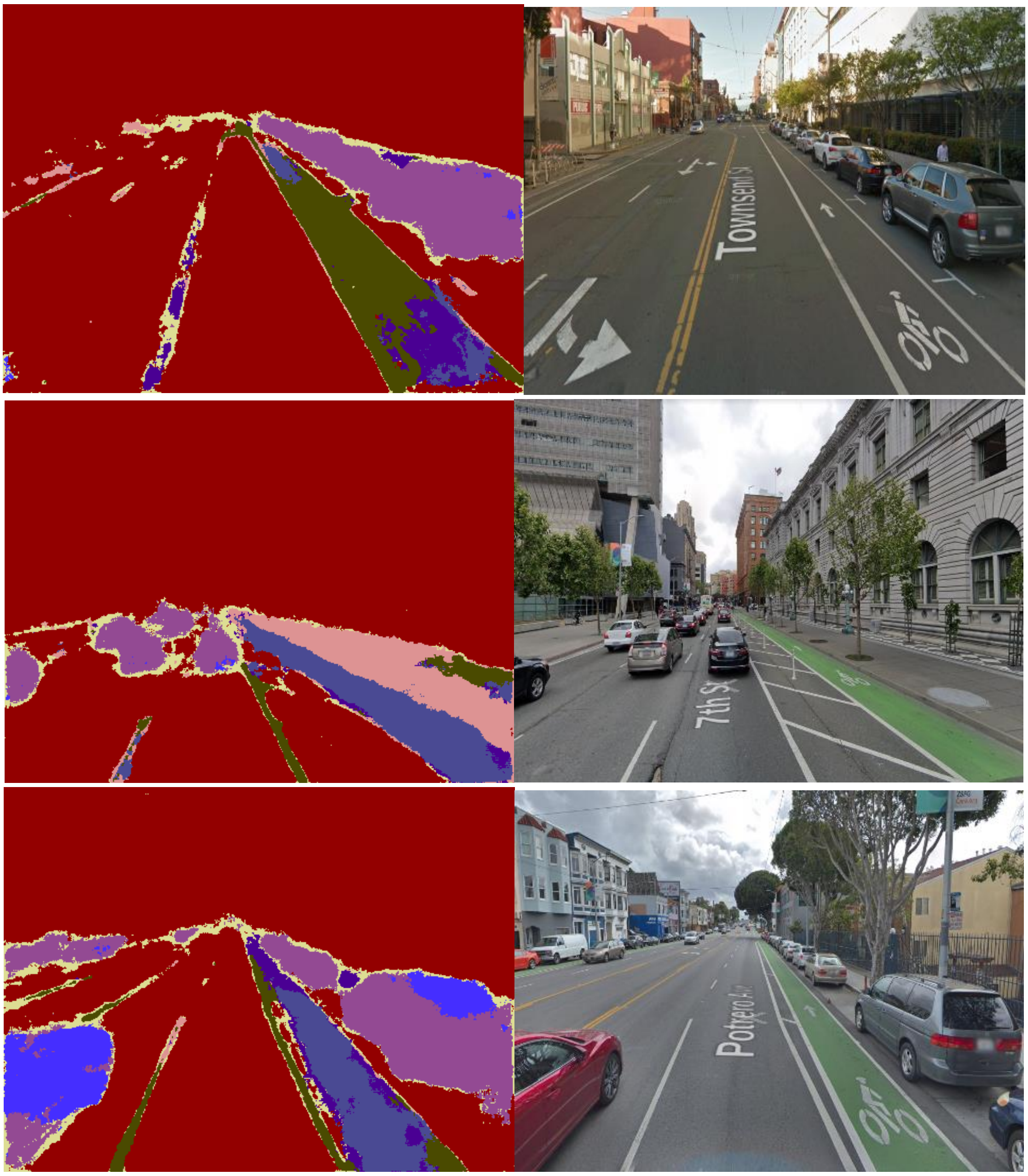


Figure 10: Testing images

We can find out from several key points from the testing result given above:

- The model can precisely recognize the boundaries between objects compared to the original model. And can precisely classify the background. That is to say, although the classification of objects may not be very precise, it can at least tell the important object we care about from the background.
- The recognition of vehicle is relatively high in all the classes. Almost all the vehicles can be precisely classified from the street views.
- The model can't tell the difference between the yellow lane divider and white divider. It ambiguously classified them all into white dividers. Yet although it cannot tell the difference

between different colored lane dividers, it can recognize the lane dividers relatively precise compared to the official model.

- Most importantly, the model can precisely recognize the separate bike lane when the bike lane is colored in green. Yet when the lane is not colored in green, it will recognize the whole lane as a lane divider.

Conclusion

General speaking, the modified model can detect the important object from the background better than the official model. The modified model is more sensitive to the lane divider and has better ability to detect vehicles. The better sensitivity of lane dividers is extremely important in the future evaluation of the street views. Because the lane divider can help divide the images into different lanes. Thus, the parking lane and driving lane can be detected from the street views.

The detection of separate bike lane still need improvement. The modified model can successfully detect the bike lane when they are colored in green. Yet when the bike lane has no coloring, the model tends to confuse it with the lane dividers. It can still recognize the object from the background, but will put it into lane dividers class.

The classification of sidewalk is not accurate. This may because the labeling methodology has labelled all the objects on the sidewalk (including the trash can, pedestrian, trees, etc.) all into sidewalk. And the variety of the object on the sidewalk may confuse the model and make it hard to generate a general pattern from the images. Therefore, the methodology on labeling the sidewalk may need to be improved.

In general, the modified model tends to be more suitable for the street views evaluation of San Francisco and we can believe that in the future through enlarging the training set and revising the label methodology can make the evaluation of street biking environment from street views feasible and efficient.

Limitations

In the process of manually collecting street views from Google Map, the baseline of choosing the appropriate street views are ambiguous. There is no clear definition of the relative heading angle according to the approaches' heading. And in the future, if the methodology is going to be promoted to a larger scale, the collection of sample street view for evaluation are difficult to conduct to make sure they have the same quality as the dataset.

Due to the limitation of time, the training set built up by myself is not large enough to train a precise image segmentation model. Compared to the official model trained by SegNet team and Streetscore platform by MIT media lab, 405 images of training set are much smaller than 3000 images.

In the image segmentation model, the class weight is a key parameter that should be fine-tuned to make sure the important class can be precisely classified by the model. Yet this research only tried two possible combinations of the class weight. If the class weight can be fine-tuned in the future, the segmentation model may become more accurate on those important by relatively low frequent classes.

Acknowledgement

Funding for this project was provided by the UC Berkeley Safe Transportation and Research Education Center (SafeTREC) and the Collaborative Sciences Center for Road Safety (CSCRS), a U.S. Department of Transportation-funded National University Transportation Center led by the University of North Carolina at Chapel Hill’s Highway Safety Research Center.

References

[1]. V. Badrinarayanan, A. Kendall and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," in IEEE Transactions on Pattern *Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481-2495, 1 Dec. 2017.

[2]. Steinmetz-Wood, M., Velauthapillai, K., O’Brien, G. et al. Assessing the micro-scale environment using Google Street View: The Virtual Systematic Tool for Evaluating Pedestrian Streetscapes (Virtual-STEPS). BMC Public Health 19, 1246 (2019) doi:10.1186/s12889-019-7460-3

[3]. Hara, Kotaro & Froehlich, Jon. (2015). Characterizing and visualizing physical world accessibility at scale using crowdsourcing, computer vision, and machine learning. ACM SIGACCESS Accessibility and Computing. 13-21. 10.1145/2850440.2850442.

[4]. Raskar, Ramesh & Naik, Nikhil & Philipoom, Jade & Hidalgo, Cesar. (2015). Streetscore -- Predicting the Perceived Safety of One Million Streetscapes. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops. 10.1109/CVPRW.2014.121.

[5]. Najafizadeh, Ladan & Froehlich, Jon. (2018). A Feasibility Study of Using Google Street View and Computer Vision to Track the Evolution of Urban Accessibility. 340-342. 10.1145/3234695.3240999.