CS 446 Machine Learning Final Project

Road Boundary Detection in Image via Machine Learning
Final Project Report

Team Member:

Dongbo Wang (dwang49)

Bangqi Wang (bwang34)

Yibo Jiang (yjiang71)

1. Introduction

Road boundary detection in image is a fundamental topic in SLAM and other applications. The main goal is to detect the region in the image that corresponds to the road area. Good road boundary detection results could help construct create map applications and driverless cars. While this topic is often directly related to the field of computer vision, the current algorithms might not work well when the images are made complicated with the presence of irregular elements like the vehicles and pedestrians on the road, shadows of the trees, etc. However, machine learning, especially deep learning, is a great tool to tackle problems like this. After experiencing with enough samples, the deep learning algorithm should be able to learn the key features of the road and find out the road area under complex environments.

Our project managed to combine the recent research results in computer vision and machine learning, namely SLIC and deep learning. Our basic strategy is to segment the image into superpixels first using SLIC and perform classification on each superpixel into road / nonroad using deep learning. We are able to achieve an average of 95% accuracy on the KITTI dataset.

2. Problem Statements

2.1 Problem Definition

As previously defined, road boundary detection is find the image regions that corresponds to the road area. This is a rather general definition since there are plenty of road types, each has its unique characters and surroundings. To limit our problem to a reasonable scale, our project is focusing on typical urban roads in the US. We are using the "Road/Lane Detection Evaluation 2013" dataset provided by "The KITTI Vision Benchmark Suite", which is limited to urban roads as shown in Figure 1. Still, we believe that with enough examples and time, our project should be able to deal with any types of roads.

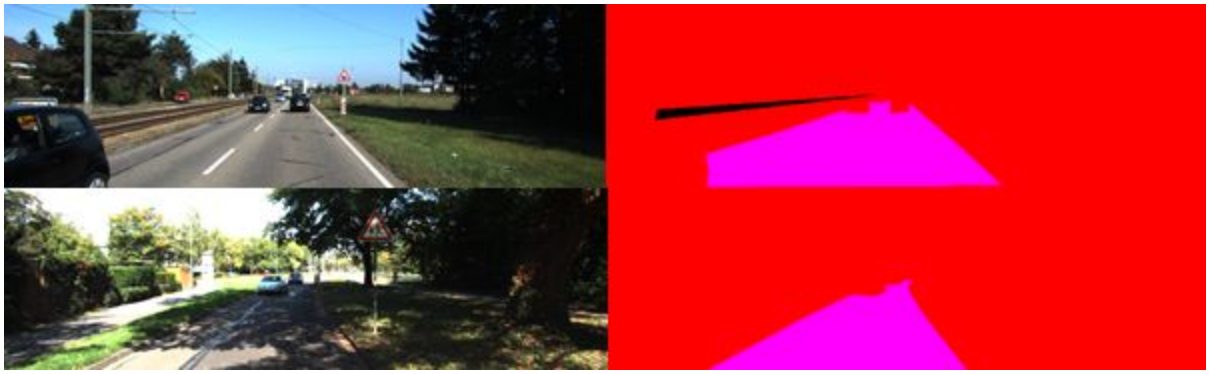

Figure 1. Urban Road Image and Labeled Ground Truth Image

This KITTI dataset, provides 289 urban road images paired with their labeled ground truth. Our goal is to learn a model that could take in an urban road image like the left ones and produce a labeled image like the right ones, the performance could be evaluated by comparing our resulting labeled image to the ground truth labeled image. It could be noticed that there is a black region in the top right ground truth image, that is the label for the purpose of other research and not part of our consideration. We only look at the purple region that is the road and consider all other regions as nonroad region.

2.2 Method Description

We consider our project as a common machine learning problem that consists of three major steps. First, we generate our dataset by extracting certain features from the image and assign a label to each example. Second, we apply the appropriate machine learning model, in our case, neural network and convolutional neural network from deep learning, and train the model with five fold validation. Finally, we applied the trained model to the test data and analyze its performance. The final stage of testing is straightforward and not much could be changed there, so we put lots of thoughts into the first two steps. For the first step of feature extraction, we implemented two different approaches, one is extract the numerical features into vectors and the other is to extract features as small images. For the second step, we applied different deep learning models to different feature types, neural network for numerical feature vectors and convolutional neural network for superpixel feature images. The general block diagram is shown in Figure 2. We will explain more about the two approaches in the following sections.

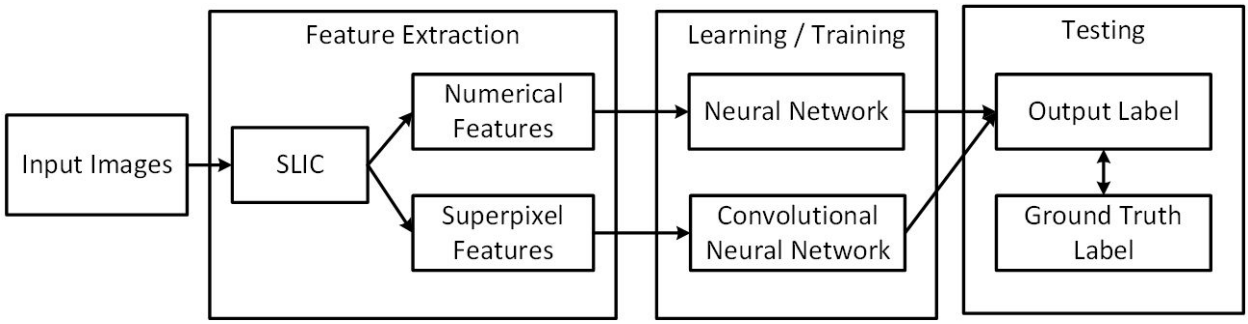

Figure 2. Block Diagram

2.3 Input Images

There are a total of 289 images and each has one ground truth labeled image. The images are RGB images of size 1242 X 375 and the ground truth images are also RGB images with the same size. In the ground truth images, the road areas are purple with (R, G, B) = (255, 0, 255) and nonroad areas are either red with (R, G, B) = (255, 0, 0) or black with (R, G, B) = (0, 0, 0). So in practice, we only check the blue channel of the ground truth images and consider pixels with value 0 as nonroad and pixels with value 255 as road. For the five fold validation, we shuffle the 289 images randomly and separate them into five groups and create the folds by take one group as testing set and the other four groups as training set.

2.4 Feature Extraction

Detect the boundary of the road is not an usual image learning problem because instead of doing classification of each image, we are trying to learn about how to segment the image into road and nonroad regions. Intuitively, this problem is similar to the image segmentation problem of separating foreground and backgrounds. There are algorithms in the field of computer vision such as graph-cut algorithm and so on. There are not much learning involved in these algorithms except for some EM algorithm and they may not work very well when the foreground is complex, like the 'noises' on the road area. Therefore, we need something that could transform this image segmentation problem into some classification learning problem. Classifying every single pixel is certainly a bad idea, because a

single pixel contains far less information than a group of pixels. So we think it would be better if we could cut the image into small regions and perform classification on these small regions. That's why we used SLIC (Simple Linear Iterative Clustering) algorithm that could segment an image into small segments first without requiring any further knowledge about the image. SLIC is a widely used algorithm in computer vision for it is simple to implement, fast to run and easy to adapt to different images. The algorithm of SLIC is shown in Algorithm 1 below [2].

Algorithm 1 SLIC superpixel segmentation

/ Initialization */*
Initialize cluster centers $C_k = [l_k, a_k, b_k, x_k, y_k]^T$ by sampling pixels at regular grid steps S.
Move cluster centers to the lowest gradient position in a 3×3 neighborhood.
Set label $l(i) = -1$ for each pixel i.
Set distance $d(i) = \infty$ for each pixel i.

repeat
  / Assignment */*
  for each cluster center C_k **do**
    for each pixel i in a $2S \times 2S$ region around C_k **do**
      Compute the distance D between C_k and i.
      if $D < d(i)$ **then**
        set $d(i) = D$
        set $l(i) = k$
      end if
    end for
  end for
  / Update */*
  Compute new cluster centers.
  Compute residual error E.
until $E \leq$ threshold

The basic idea is to group pixels into superpixels by the difference of color and distance. For each image, we specify the rough number of superpixels we want to get and SLIC will initialize the cluster centers evenly throughout the image. Then the algorithm makes several iterations, in each iteration, it assign pixels to the cluster center that has the nearest distance where the distance is a combination of color and spatial distance. Then the cluster center is updated to the average position of the cluster and small clusters got merged into larger once. After enough iterations, when the cluster center stop moving, stop the loop and output the result. There are several parameters that could be tuned for the algorithm but there is no standard criteria on what is a good parameter setting. So we only modify the parameters slightly different from the standard settings. With our setting, each image is segmented into roughly 800 superpixels and the superpixels adheres to the boundaries in the image naturally as shown in Figure 3. We want to make the superpixel boundaries adhere to the road boundary as much as possible because in that case the road region could be defined by a set of superpixels. Once we learn how to classify the superpixels into road and nonroad, we could put together all the superpixels inside the road region and then segment the image easily.

Figure 3. SLIC Superpixel Segmentation Result

Once we have the superpixels, we only need to extract some feature from each superpixel to create a single example in our dataset. As mentioned before, we have two ways of extracting the features out of each superpixel as shown in figure 4.

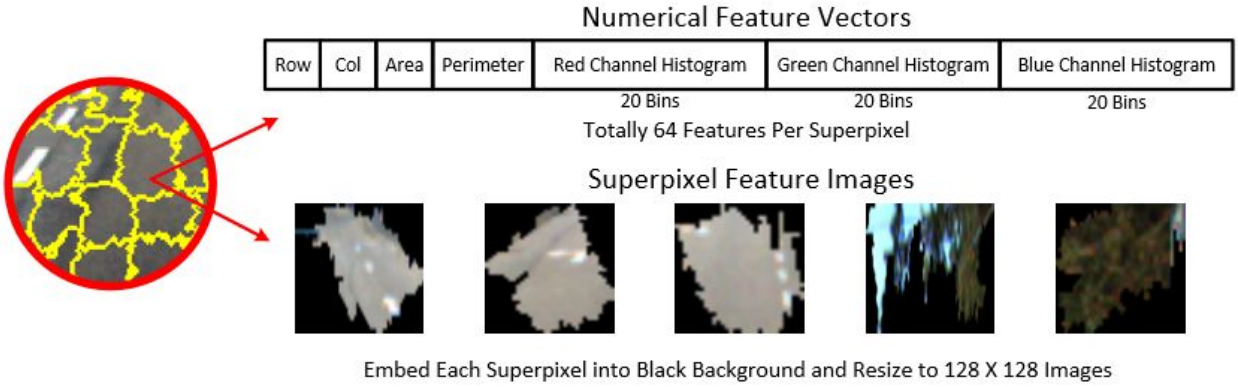

Figure 4. Feature Extraction

To extract the numerical feature vector, we look at each superpixel and compute the following statistic results of that single region, including the centroid row and col index, the area and perimeter of the superpixel, the 20 bins histogram for each of the RGB channels. So each example has 64 features, encoding the geometric and color information of that superpixel. The final dataset consists of 208693 numerical feature vectors, including 166031 nonroad vectors and 42662 road vectors.

To extract the superpixel feature image, we cut out each superpixel, put them into a black background and resize the patch into a 128 X 128 image. We implement this method to take advantage of the convolutional neural network, for we have changed the segmentation problem into an image classification problem of each superpixel that naturally falls into CNN's area of expertise. The final dataset consists of 208693 images. This size is too large so we use 2000 from each class to train.

To assign a label for each superpixel, we simply check whether the centroid of each superpixel is within the road region in the ground truth image.

However, there is some downside carried with this SLIC processing. For the superpixels might not adhere perfectly to the road boundaries. This will happen when the road boundary is ambiguous in the original

image meaning that the gradient is not high enough to be detected by the algorithm. This will be a common issue for all boundary detection problem because if you cannot found the boundaries at the first place, there is no place for further detection. To address this problem, we tested the the SLIC performance by checking the area error between the labeled training images and the ground truth images. It turns out that the SLIC will produce an average area error of 1.15%, meaning that there is a systematic error of around 1% carried to the final accuracy.

2.5 Training and Learning

For training and learning, we use several different methods for both numerical feature vectors and superpixel feature images. For each type of data, we construct a binary classifier to classify road and nonroad images. We tried Perceptron, Winnow, and Adagrad for numerical feature vectors. This section will only introduce methods with highest accuracy. We use neural network for numerical feature vectors and deep learning for superpixel feature images.

2.5.1 Neural Network for Numerical Feature Vectors

For numerical feature vector, the neural network is the best classifier with over 97% accuracy. The neural network contains two hidden layers. First hidden layer has 256 nodes and the second hidden layer has 128 nodes. Figure 4 shows the layout for neural network.

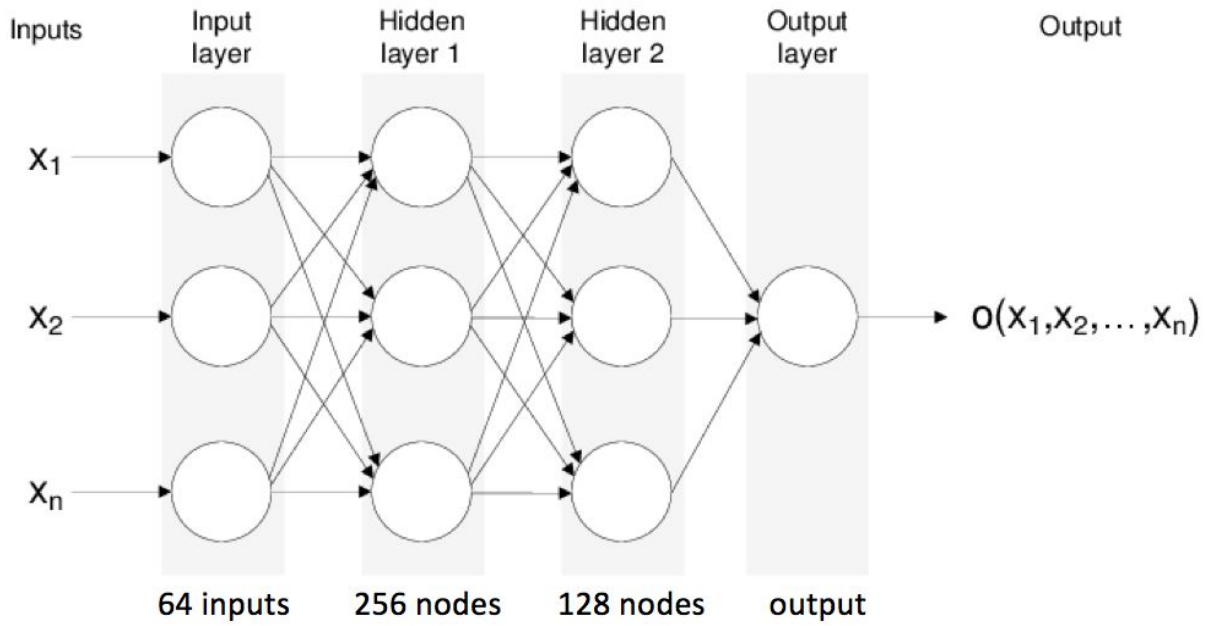

Figure 4. Neural Network Layout

The neural network will use backpropagation to update weights for features and improve performance during the training process. We tried several parameters such as 512, 256, 128, and 64 nodes for hidden layers. Hidden layer with 256 nodes gets the highest accuracy, over 97%.

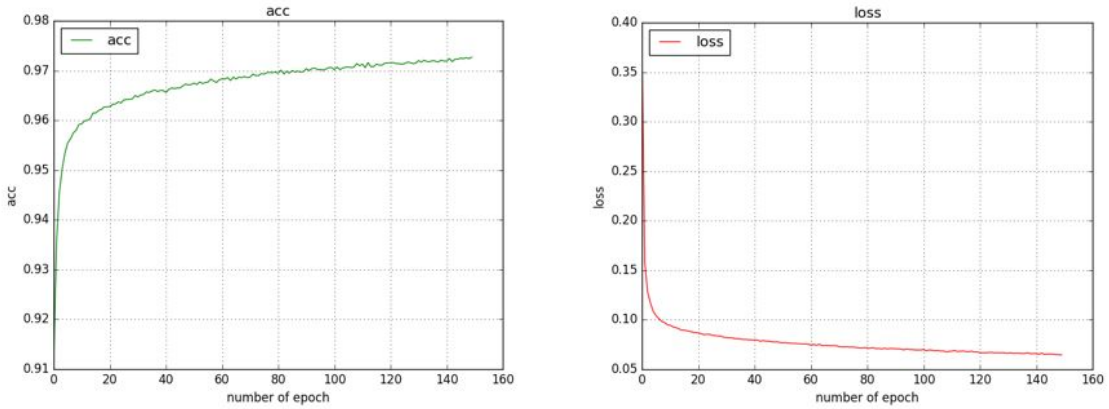

Figure 5. Training Accuracy & Loss for Neural Network

2.5.2 Superpixel Feature Images

We use deep learning convolutional neural network as a binary classifier for superpixel feature images. Due to the time constraints and computing power limitation, we use 2000 nonroad images and 2000 road images as training dataset. We design our deep learning neural network based on a deep learning neural network for breast cancer diagnosis [1]. Our deep learning neural network has four convolution layer, four max-pooling layers, and ReLu between convolutional layers and max-pooling layers. We test two different layouts with 2048 and 256 nodes in fully connected layers. Figure 6 shows the neural network layout.

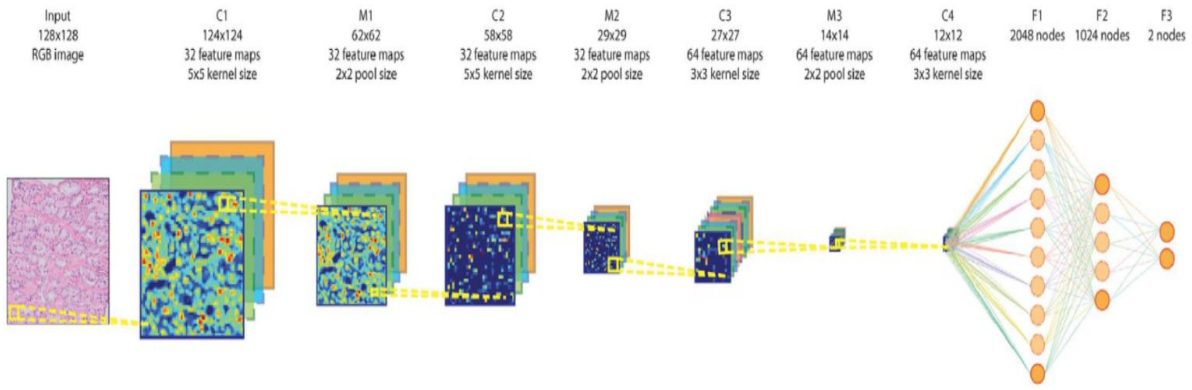

Figure 6. Deep Learning Neural Network Layout

For deep learning training, each epoch cost about 430 seconds. We terminate the training process when there is a harbinger of overfitting. According to the training and validation curves in figure 6, deep learning neural network can achieve accuracy over 90%. If we could have more time, we can train the neural network on larger dataset, adjust parameters for better performance, and modify neural network layouts.

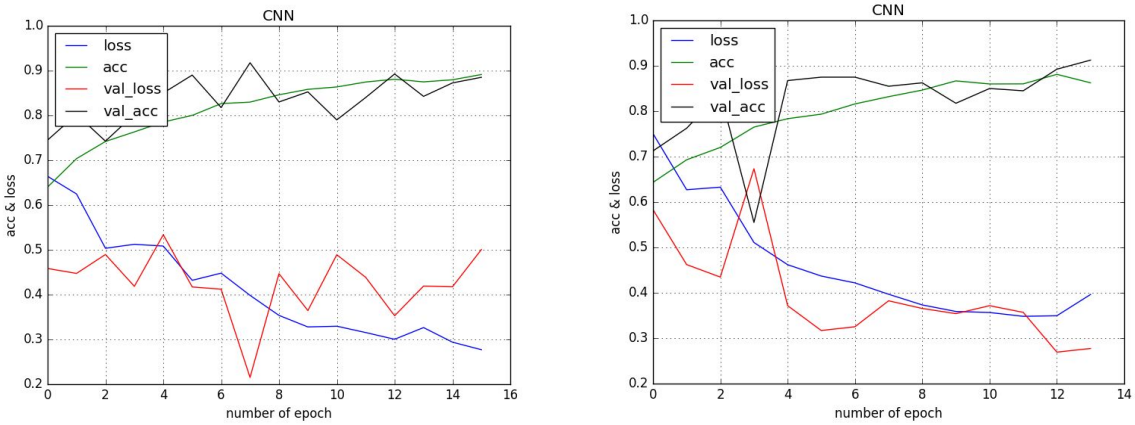

Figure 7. Accuracy & Loss for Deep Learning

2.6 Testing

For testing, we perform the similar procedure as before. First, we perform SLIC to each test image to segment the image into superpixels. Second, we extract the features as specified before, either into numerical feature vectors or superpixel feature images. Then, we use the trained model to classify each superpixel into road or nonroad category. Finally, we produce a labeled image and compare the labeled image with the ground truth label image for evaluation. We are using the area to measure the accuracy.

3. Results and Evaluation

3.1 Visualized Detection Result

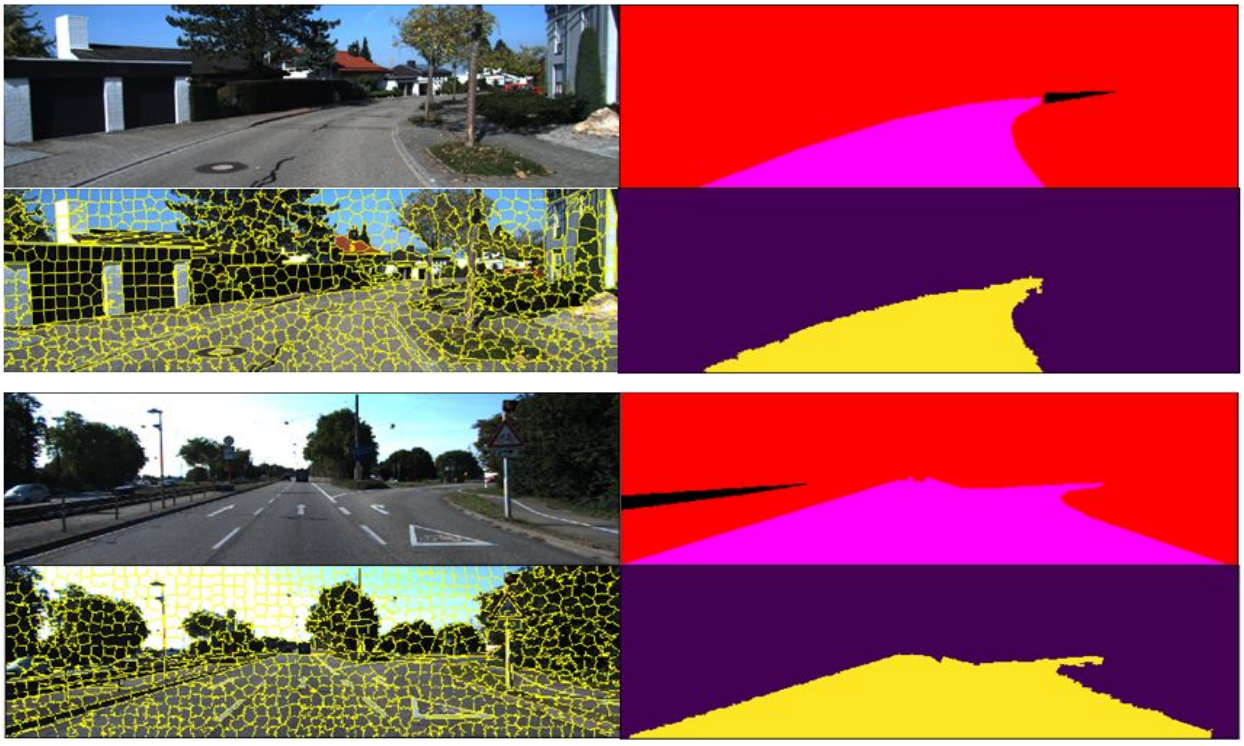

Figure 9. Exemplary Visualized Detection Results

3.2 Evaluation

The overall accuracy across the five fold validation is shown in Table 1 and Table 2. It also shows the false positive rate (FPR) where nonroad pixels detected as road, and false negative rate (FNR) where road pixels detected as nonroad.

Fold #	1	2	3	4	5	Average
Accuracy	97.45%	98.33%	98.21%	97.12%	97.81%	97.78%
FPR	1.02%	0.78%	0.68%	1.25%	1.07%	0.94%
FNR	1.53%	0.89%	1.11%	1.63%	1.12%	1.28%

Table 1. Numerical Feature Vectors Detection Statistics

Fold #	1	2	3	4	5	Average
Accuracy	95.37%	96.24%	95.15%	95.81%	96.01%	95.72%
FPR	1.54%	1.77%	1.66%	1.32%	1.22%	1.50%
FNR	3.09%	1.99%	3.19%	2.87%	2.77%	2.77%

Table 2. Superpixel Feature Images Detection Statistics

The overall performance of the two models are all very good. The numerical feature vectors has an accuracy slightly higher than the superpixel feature images. This unexpected because the superpixels contain more information than the vectors. However, the superpixels does not contain the geometric information about where the patch is inside the image and we are not sure how the black background will effect on the overall accuracy. Also, due to time limitations, we weren't able to try more CNN structures and parameter settings, so we cannot conclude that the CNN performance could not be enhanced.

There are other things that worth mentioning here. The unbalanced dataset with about 70% of non-road regions and 30% of road regions may bias the accuracy higher than expected. Also, the dataset is relatively small including several pictures taken consecutively on the same road, which is not a problem for computer vision when the dataset was created, but might be a big issue for machine learning because it is basically including some training examples in the testing examples which will definitely increase the overall accuracy.

4. Related Work

There are a bunch of related work for this problem listed under the KITTI dataset, the top accuracies is around 97%, but they seems to be using a different criteria evaluating the accuracy and they used the information from camera calibration to transform to bird view, so we don't think our result is comparable to their results.

5. Future Work

We have recognized that the SLIC will produce a systematic error around 1%, so we should find a better way to perform image segmentation that will produce a smaller systematic error. Also, we noticed that the boundaries we find may not be smooth and continuous due to poor segmentation. However, it is reasonable to assume that in the real world, the road boundaries are smooth and continuous. To improve the final result, we could apply some post-processing to smooth the boundaries of the road regions.

6. Conclusion

In conclusion, the two methods we developed, neural network for numerical feature vectors and convolutional neural network for superpixel feature images, are able to detect the road boundary with high accuracy around 97%. There is a systematic error produced by the SLIC algorithm. Also, the high accuracy might be biased by the dataset that is not big enough and unbalanced. In general, we believe our model should be able to detect the urban road boundaries in images with high accuracy.

7. Reference

[1] G. Litjens, C. Sanchez, N. Timofeeva, M. Hermsen, I. Nagtegaal, I. Kovacs, C. H. van de Kaa, P. Bult, B. van Ginneken, and J. van der Laak, Deep learning as a tool for increased accuracy and efficiency of histopathological diagnosis," Scientic Reports, vol. 6, no. 1, 2016.

[2] Achanta, Radhakrishna, et al. "SLIC superpixels compared to state-of-the-art superpixel methods." IEEE transactions on pattern analysis and machine intelligence 34.11 (2012): 2274-2282.

[3] Joze, Hamid Reza Vaezi, and Mark S. Drew. "Improved machine learning for image category recognition by local color constancy." Image Processing (ICIP), 2010 17th IEEE International Conference on. IEEE, 2010.

[4] Alhussein, Musaed. "Image Tampering Detection Based on Local Texture Descriptor and Extreme Learning Machine." Computer Modelling and Simulation (UKSim), 2016 UKSim-AMSS 18th International Conference on. IEEE, 2016.

[5] Li, Yuanzhong, Shoji Hara, and Kazuo Shimura. "A machine learning approach for locating boundaries of liver tumors in ct images." Pattern Recognition, 2006. ICPR 2006. 18th International Conference on. Vol. 1. IEEE, 2006.