Introduction

Traffic sign classification is a multiclass classification task which build a classifier to recognize different traffic signs on the road. (e.g. "speed limit" or "stop" or "turn ahead"). This is a crucial for emerging self-driving cars to obey traffic law. In this project, we designed and trained a machine learning model to classify German traffic sign from the German Traffic Sign Benchmark.

This is a typical image classification task. To obtain a high accuracy classifier, we used convolutional neural network to build the model. Convolutional neural networks(CNN) are very similar to ordinary neural networks: they are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network still expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other. And they still have a loss function (e.g. Softmax) on the last fully-connected layer and all the tricks learning regular Neural Networks still apply. CNN has been the gold standard for computer vision problems since its first big success in ImageNet competition.

Dataset Exploration

The dataset we are using provided by The German Traffic Sign Benchmark. It is a multi-class, single-image classification challenge held at the International Joint Conference on Neural Networks (IJCNN) 2011. The data set has more than 40 classes. A subset of classes shown in Table 1. And more than 50000 RGB images in size of 32x32 with manual labels. In this section, we derived some statistic of the dataset such that we can recognize the potential challenges and pitfall of this task. Figure 1 shows the label count distribution. Figure 2 shows some sample traffic sign image in the data set.

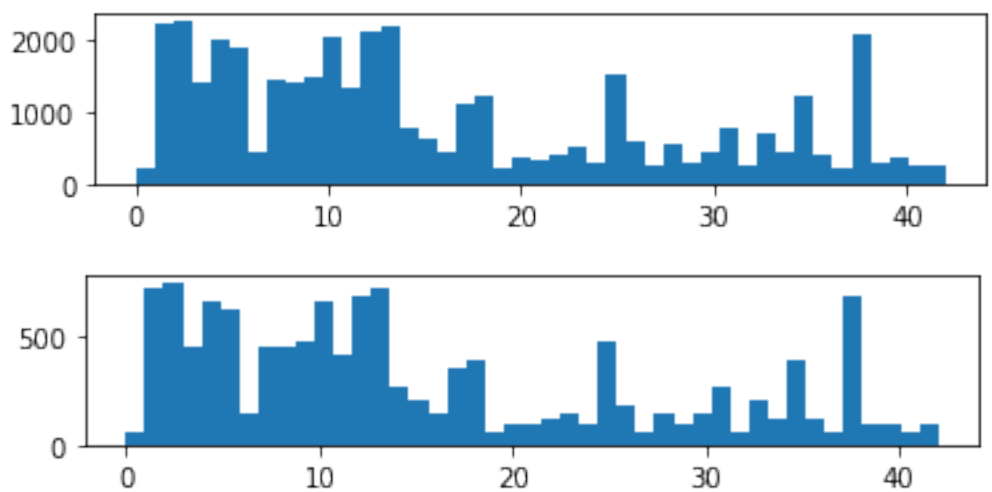
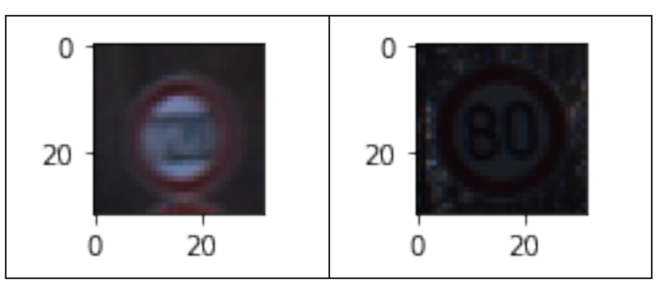


Figure 1: Figure on the top is label count distribution of train data. Figure at the bottom is label count distribution of test data.



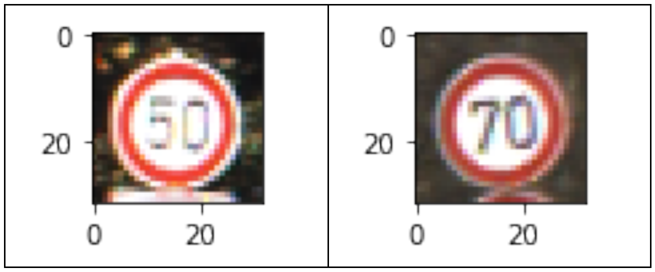


Figure 2: sample traffic sign images from training data

From above figures we found that the label distribution is rather unbalance. This could make the classifier overfit to some classes. Also, some classes in the data set are similar to each other, such as speed 50 and speed 70 in figure 2. Though it is easy for human to distinguish between them, the machine will have difficulty because of the similar feature. Moreover, figure 2 shows that images were taken under different light conditions and some image might be blur compare to others. These all pose challenges to build an accuracy classifier.

Class ID	Traffic Sign
0	Speed limit (20km/h)
1	Speed limit (30km/h)
2	Speed limit (50km/h)
19	Dangerous curve to the left
20	Dangerous curve to the right

Table 1: subset of traffic sign classes in the data set

Convolutional Neural Network

A convolutional neural network (CNN) is comprised of one or more convolutional layers (often with a subsampling step) and then followed by one or more fully connected layers as in a standard multilayer neural network. The architecture of a CNN is designed to take advantage of the 2D structure of an input image (or other 2D input such as a speech signal). This is achieved with local connections and tied weights followed by some form of pooling which results in translation invariant features. Another benefit of CNNs is that they are easier to train and have many fewer parameters than fully connected networks with the same number of hidden units.
Successful application of CNN in image classification task dates back to 1998 when Yann LeCun et al developed LeNet-5, a pioneering 7-level convolutional network that classifies digits, is applied by several banks to recognize hand-written numbers on checks digitized in 32x32 pixel images. Figure 3 shows the architecture of LeNet-5.

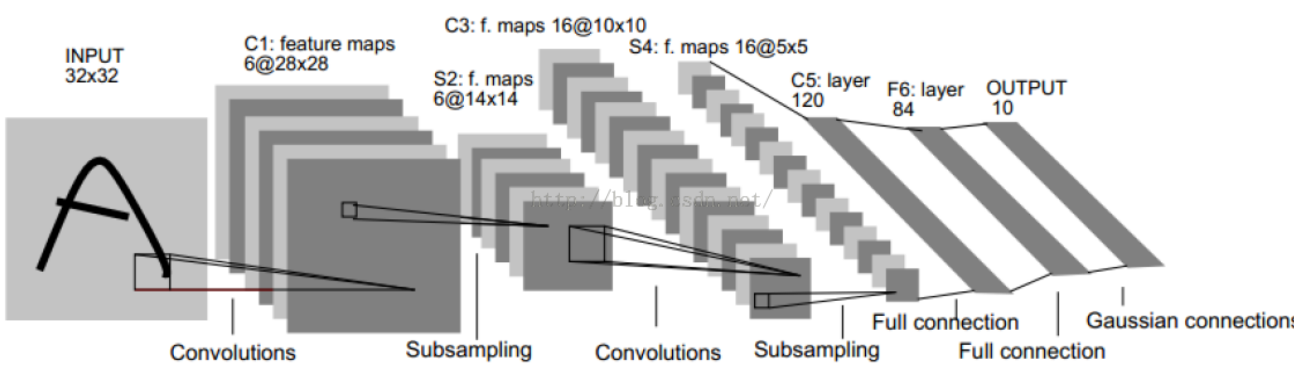


Figure 3. Architecture of LeNet-5

Experiment

Although LeNet-5 has a simple architecture, it can achieve an accuracy higher than 99% in recognizing hand-written digits. Since the input size is the same as LeNet-5 (i.e. 32x32 images), in this project we will adopt this model to our traffic sign classification task. Before we train the network, we must address the data unbalance problem as well as potential overfitting. To overcome these challenges, we use some data augmentation technique to generate some additional data from original data set. For each class the generate fake images until the number of image of this class reach 2200. To generate fake images, we take an image from original dataset, then randomly change its brightness, and randomly blur the image using Gaussian kernel. In such way we can generate nearly infinite amount of data. After derived the dataset, the normalize the dataset to -0.5 and 0.5. This normalization preprocess will make the model converge faster and generalize better. To adopt LeNet-5 to our task, we add drop-out layers to combat the overfitting problem. We used stochastic gradient descent with initial learning rate 0.01 and rate decay 0.0005 to optimize the model. The final model reach accuracy higher than 96% on test data. Figure 4 shows the training statistics.

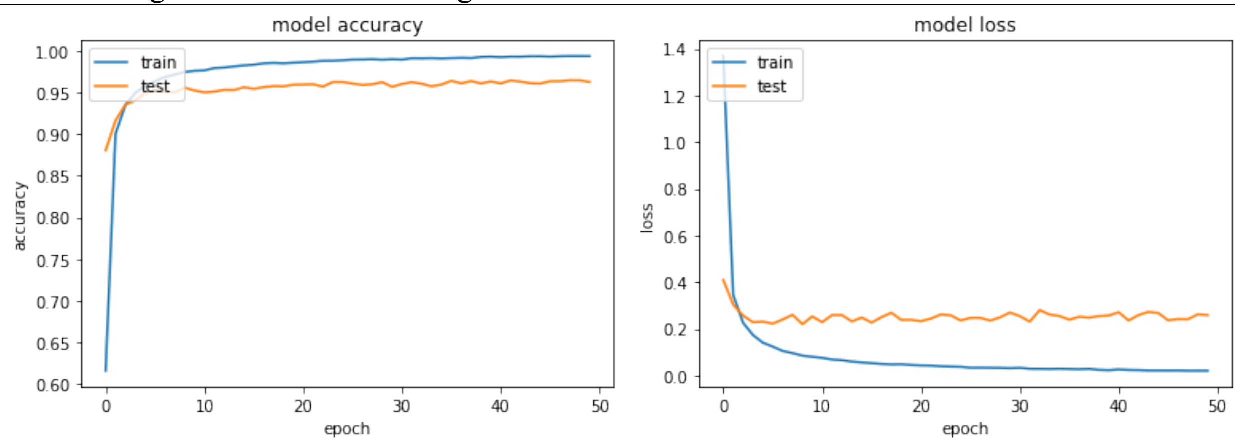


Figure 4: Left figure shows model accuracy. The final accuracy on test data is 96.26%. Right figure shows training loss during the training process.

Conclusion

As the experiment result shows, convolutional neural network is a powerful technique for image recognition. With such a simple architecture, we can obtain a high-performance classifier for a complicated task such as traffic sign classification. Though the model is powerful, it has some inherent disadvantage such as easily overfitting. To overcome such problem, data augmentation is the most straightforward solution.