The Hong Kong Polytechnic University

Department of Electronic and Information Engineering

Final Year Project Report:

Road Traffic Sign Classification Under Challenging Conditions

Supervisor	**: Prof. Kenneth Lam**
Student Name	**: FENG Weixi**
Student ID	**: 14109625D**
Submission Date	**: 28th Nov, 2017**

Contents

I. INTRODUCTION/BACKGROUND

Automatic Driving System(ADS) is a key component in the area of artificial intelligence applications. Traffic sign detection and recognition are two important tasks in this system, because traffic signs are visual symbols instructing how should the vehicle be driven. Therefore, knowledge in Computer Vision and Pattern Recognition(CVPR), Machine Learning and Image Processing is required.

There are many existing traffic sign datasets including but not limited to German Traffic Sign Detection Benchmark(GTSDB)[1], German Traffic Sign Recognition Benchmark(GTSRB) [2] and Belgium Traffic Sign Data Set(BelgiumTS). All these datasets are generated from real world scenes, where the environment variables are mutually different from each other.

A. Motivation

Although state-of-the-art algorithms have nearly perfect performance on these datasets, these works are still far away from real world application. First of all, these datasets are too small to represent the space of elementary events in reality. Hence, most of the mathematical models are not generalized enough to be incorporated into real-time system. Secondly, images in these datasets do not emphasize the relationship between challenging conditions and algorithm performance. The solution to the first problem is limited by the fact that large and comprehensive datasets are merely available for students and faculty in academic institutes, except for huge enterprise like Google and Facebook. Therefore, solution to the second problem becomes the focus of this project, as it is less restrained by the amount of data.

In this project, Challenging Conditions(CC) are defined as possible events that will negatively affect the extraction of useful information from images. Just like different noises in signals, Challenging Conditions may downgrade algorithmic performance to a certain extent. Based on the dataset we use in this project, Challenging Conditions are classified into 14 types, including fog, rain, snow etc.

B. Objective

This project is an international capstone group project of PolyU and University of Technology, Sydney (UTS), supervised by Prof. Kenneth Lam. Among the four group members, two are from PolyU EIE department, and the other two are from UTS. Originally, the target of the group project was to develop a surveillance system that could detect, track and recognize figure and face, where I was responsible for face recognition implementation. However, in around April this year, we noticed that there is a competition organized by the IEEE Signal Processing Society, named Video and Image Processing(VIP) Cup. After discussions and thorough consideration, our group decided to join the competition. Hence, our project topic was changed to Road Traffic Sign Detection System Under Challenging Conditions. All group members were assigned papers focusing on different methods ranging from Conventional Methods to Deep Learning. From May to July this year, I was trying to discover robust Deep Learning methods to classify traffic signs. This is the reason that my topic has been changed as Traffic Sign Classification Under Challenging Conditions.

II. Related Work

Before the widespread of Convolutional Neural Network(CNN), classification is completed based on conventional methods such as Support Vector Machine[3].

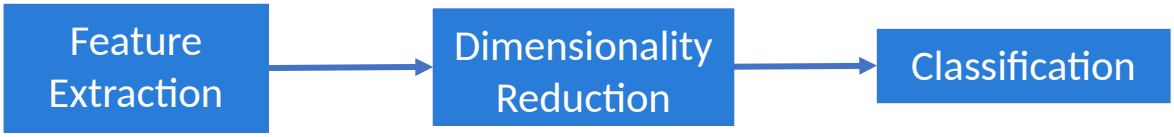

Figure 1 Traditional system

As is shown in fig. 1, the base pipeline consists of three stages: feature extraction, dimensionality reduction, and classification. The most commonly used hand-crafted feature in traffic sign classification is Histogram of Oriented Gradient(HoG)[4]. Feature extraction is to extract the informative data and eliminate redundant data from an initial set of measured data. The extracted feature should be representative to the original data and discriminative to data with different labels. Dimensionality reduction is to decrease the number of dimensions

of a feature, keeping as much useful information as possible. The computation complexity of a system can be largely alleviated thanks to dimension reduction.

In [5], a comprehensive comparison between different methods is conducted, including Nearest Neighbour, Spare Representation-based Classifier and Support Vector Machine. The best algorithm on GTSRB is HoG + Iterative Nearest Neighbors based Linear Project (INNLP) + Iterative Nearest Neighbors Classifier (INNC), which reached an accuracy of 98.53%. Some other classifiers have been considered, including Sparse Representation-based Classifier (SRC) and multiple types of SVM.

However, none of the methods outperform the best Deep Learning based method, which won the first place on GTSRB competition several years ago. Recent CNN-based methods include creating ensemble of CNNs [6], multi-scale CNN [7] and much deeper networks.

This project is based on two papers about traffic sign classification on GTSRB dataset. These two papers proposed two different methods in doing classification. One is based on Convolutional Neural Network(ConvNet) and the other is based on Random Forest, a traditional but still powerful classifier. In [7], a Multi-Column DNN(MCDNN) formed by various DNN was proposed to increase the recognition performance and make the network robust to variation in contrast and illumination. Their work reached an accuracy of 99.46%, which was claimed to be the only one better than human recognition performance. In [8], the authors focused more on a type of conventional classifier, tree classifier. They compared K-d trees and Random Forests in traffic sign recognition tasks, using HoG features. The best performance was achieved by Random Forest with an accuracy a little bit higher than 96%.

III. METHODOLOGY

Given a new traffic sign image, the task of classification is to decide which particular class it belongs to. There are two basic characteristics of classification tasks. First, there is a large amount of data. Second, the inter-class variance can be very large. Thus, feasible methods must be robust and generalized to deal with all kinds of possible data. Machine learning is a very good option for classification. Before the computer could make prediction, it has to go through the process called supervised learning. Briefly speaking, supervised learning is to infer a function from labeled training data. The labels of the training data are very important, because they not only tell computer the number of classes, but also force computer to correct their mistakes during the training process. For instance, in training stage, if a data is misclassified, its correct label will make the classifier adjust weights and parameters in order to correct this misclassification. In analogy, the machine is trying to find the best decision boundary by knowing the “truth” of data.

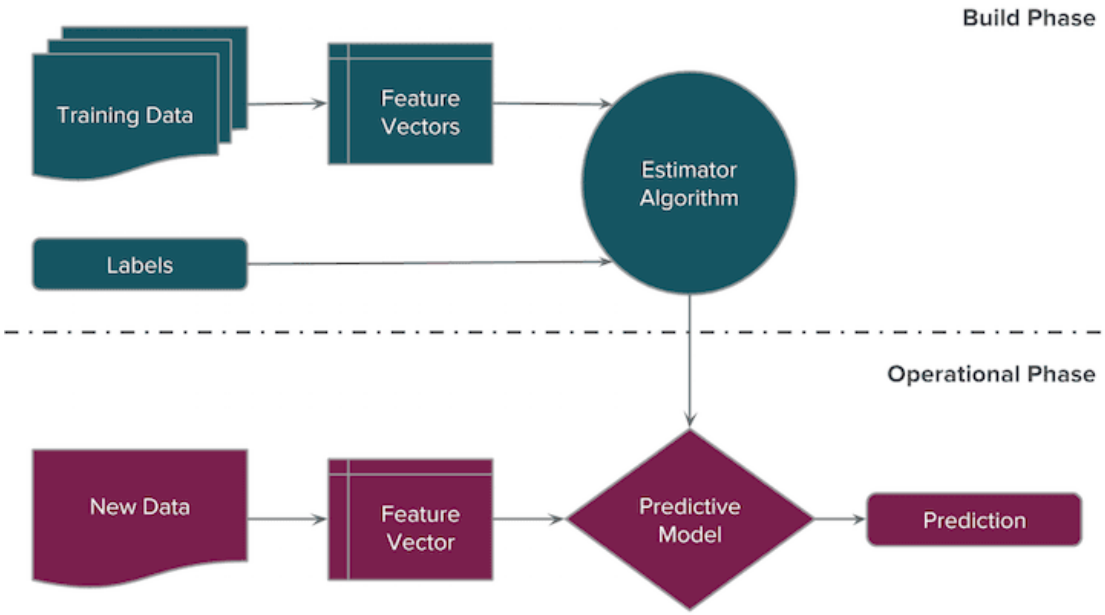

Figure 2 General procedures of machine learning

Most of the supervised machine learning methods could be illustrated by the fig. 2, doesn’t matter if it is deep learning or conventional methods. There are 2 phases in the classification task, the build phase and the operational phase. In Build Phase, feature vectors are extracted from raw data and fed to estimator algorithm with labels. The output of estimator algorithm is a predictive model being trained for a lot of times within the scope of training data. The

operational phase is relatively simple, testing the model using new data outside the scope of training data. The prediction results are analyzed in order to evaluate the performance of predictive model. Other factors, like computation cost, may also be considered.

Raw data cannot be utilized directly because it contains noise and useless features. Therefore, it is a good idea to extract discriminative features and eliminate unwanted features. Discriminative features can be hand-crafted or highly abstract. The design of handcrafted features often involves trade-off between computational complexity and accuracy[9]. For example, Scale Invariant Feature Transform (SIFT)[10] is a handcrafted feature known for its robustness to rotation and scale variations, but it is the result of high computation cost. Normally, two different algorithms are implemented respectively for feature extraction and classification. However, things are different in deep learning. Deep learning simply combined the feature extraction and classification together as a whole model called neural network. The data from some layers of this network is called abstract features or non-handcrafted features. Therefore, neural networks are sometimes regarded as a black box where the mechanism is not fully understood. Even without a mathematical understanding of neural network, people could still manipulate it to finish classification tasks.

This project focuses on two kinds of classifiers and

A. Proposed Convolutional Neural Network

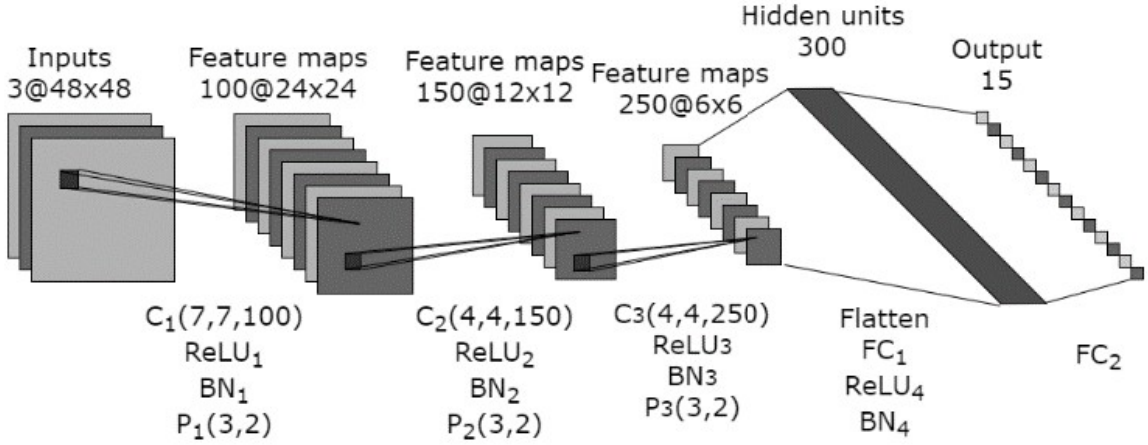

Figure 3 Proposed convolutional neural network

For road traffic sign recognition, a trade-off was made between accuracy and computation cost. Considering the driving speed of an automobile can be fast sometimes, the algorithm

should be able to classify the traffic sign faster than normal classification problem. Therefore, a simple and shallow convolutional neural network was proposed to classify the detected regions of interest (ROI) into 15 classes of road signs. The proposed network is shown in fig. 3. The architecture of our model is based on networks in [6] and [11].

There are 5 layers in the proposed network, three consecutive convolutional layers and two fully connected layers. The network takes 48 by 48 RGB images and has 15 output scores. Among the 15 output nodes, the first 14 nodes represent 14 classes of true traffic sign and the 15^{th} node represents negative samples, which are non-traffic sign images.

1) Convolutional Layer

The convolutional layer is denoted as $C_x(M, M, K)$ where M represents the width and heights of filters, K represents the number of filters, number of output channels. Each filter W^k slides across the input maps Y^{n-1} with a stride $S=1$, and produces an output map Y^n. If Y^{n-1} has K' channels, Y^n can be represented as

$$Y_j^n = \sum_{i=1}^{K'} Y_i^{n-1} * W_{ij}^n + b_j^n \; j = 1,2,\ldots,K$$

In order to keep the output width and height equal to those of input, we padded $\frac{M-1}{2}$ zeros to four sides of the input maps.

2) Max-pooling Layer

In this network, we applied max-pooling with 3x3 receptive fields and with a stride of 2. In order to keep the output size half of the input size, zeros are padded to four edges and corners of the input maps. It is called max-pooling layer because the receptive fields (or called non-linear filter) will return the maximum pixel value inside the field range. The following example shows how a max-pooling layer works.

1	1	3	3
4	6	2	7
0	9	8	5
3	4	1	7

Max-pooling layer with 2x2 filter and stride = 2

6	7
9	8

Figure 4

The purpose of max-pooling layer is to do a lot of potential computation costs can be saved. Normally, the input to a ConvNet is

an image, a collection of pixels. As data flowing through several convolutional layers and pooling layers, the nodes in each layer represents different level of abstraction, from low-level features rising to high-level features. For example, edges and corners are low-level features. The second layer of a ConvNet may extract those features. High-level features are usually very abstract that we cannot describe it from visual observation.

B. Key features

1) Choice of Activation Function

In our model, instead of using hyperbolic tangent(tanh) as the activation function, the Rectified Linear Unit (ReLU) function is employed in the whole network. The advantages of using ReLU, compared to tanh, are that ReLU can increase the sparsity and solve the 'vanishing gradient' problem [12]. However, there is a potential disadvantage of ReLU. When a unit is not activated, the gradient will be zero and the weights will not be optimized. Therefore, if a unit is not activated initially, it may never be activated until the end of training. To solve this problem, Leaky ReLU with a small slope in the negative coordinates is proposed in [12]. Even if a unit falls into negative, the weights can still be optimized because of the small gradient(normally 0.01). Then the unit may have a chance to be reactivated in the following training steps. In [11], the authors applied Leaky ReLU and had promising results in traffic sign recognition. But experiments in this project showed that ReLU performed a little bit better than Leaky ReLU and the small gradient didn't help a lot. Explanations are given in the experiment section.

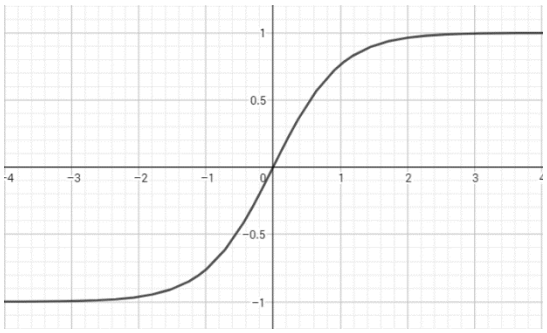

Figure 5 Hyperbolic tangent

Figure 6 ReLU and Leaky ReLU

2) Batch Normalization

Another difference between this model and the original one is that we apply batch normalization before max-pooling layer. Training of Neural Network is always restricted by the selection of learning rate. If it is too large, the loss function cannot converge due to large fluctuation of weights. On the contrary, if it is too small, the training process can be slow. This problem is referred as 'internal covariate shift' in [13], where batch normalization is proposed to solve it. Each mini-batch with value x and m number of values is denoted as

$$B=\{x_{1\ldots m}\}$$

Batch normalization transform is denoted as

$$y_i=BN_{\gamma,\beta}(x_i)$$

The transform can be written as the following equations:

$$\mu_B=\frac{1}{m}\sum_{i=1}^{m}x_i \quad minibatch\ mean$$

$$\sigma_B^2=\frac{1}{m}\sum_{i=1}^{m}(x_i-\mu_B)^2 \quad minibatch\ variance$$

$$\hat{x}_i=\frac{x_i-\mu_B}{\sqrt{\sigma_B^2+\epsilon}} \quad normalize$$

$$y_i=\gamma\hat{x}_i+\beta\equiv BN_{\gamma,\beta}(x_i) \quad scale\wedge shift$$

where γ, β are parameters to be learned by backpropagation[14].

Batch normalization is very helpful in this project because the interclass variance is quite large in the dataset. As there are 12 challenging conditions, any two of them can cause significant difference on the same image. An image can have extremely low values in darkening condition, while extremely high values in exposed condition. Therefore, when data flows into the network, the weights and biases can make the unit too high or too low. During the training of these parameters, it may be difficult for the loss function to converge if the learning rate is inappropriate. As explained above, this problem was solved by applying batch normalization after each convolutional layer. Tin each mini-batch, the mean is normalized to zero and variance is normalized to one. Experiments have shown that the performance of our model is much improved with batch normalization applied in each layer.

C. Ensemble of CNNs

1) Previous work

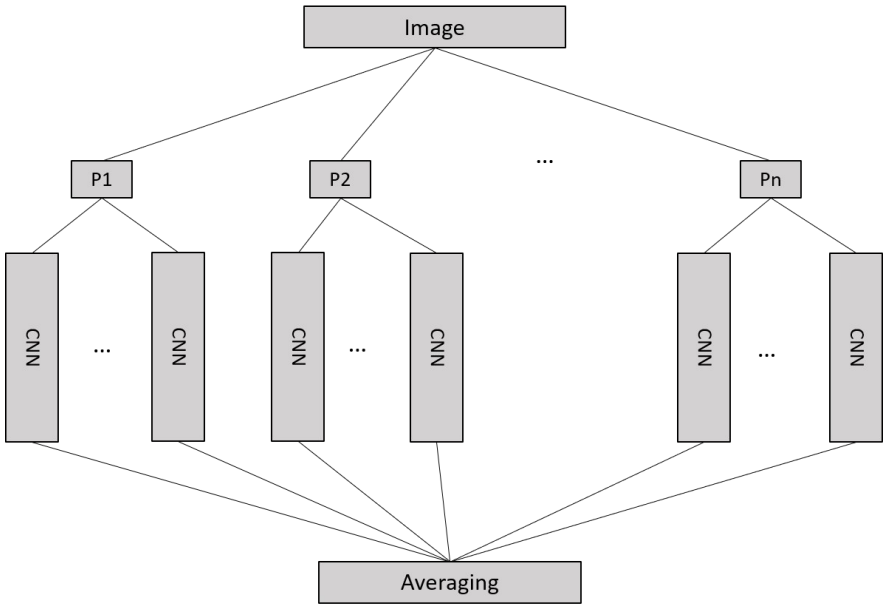

Figure 7 Multi-column CNN [6]

In [6], a multi-column CNNs were proposed to increase the accuracy of a single CNN. By creating this ensemble, they increased the accuracy of 98.52% into 99.46% on German Traffic Sign Recognition Dataset. This also helped them win the first place in the competition at that time. In their ensemble of CNNs, Px represent different pre-processor. As an image is input to the system, it will be pre-processed by different algorithms. Then different networks

take different images as input and generate an array of scores. Each entry in the score array represent the confidence that the image belongs to corresponding class. Normally, for single convolutional neural network, argmax will be used to get the prediction class.

$$n = argmax\left(\left[s_1, s_2, \ldots, s_N\right]\right)$$

However, in ensemble of CNNs, score arrays from different subnetworks are gathered and averaged into one final score array. Then apply the argmax to get the prediction class. It can be expressed as the following equation:

$$n = argmax\left(\sum_{i=1}^{M} w_i\left[s_{i,1}, s_{i,2}, \ldots, s_{i,N}\right]\right) w_i = \frac{1}{M}$$

where M represents the total amount of subnetworks, w_i represents the weight to i-th network. According to their paper, various subnetworks can be trained on inputs pre-processed in different ways. If the errors of M different models have zero mean and are uncorrelated, the average error might be reduced by averaging the M models[15].

The pre-processor step is quite important. This step is to increase the variance between any two networks and decrease the possibility of correlation between any two subnets. If all the subnetworks are nearly the same and highly-correlated, it would be meaningless to create this ensemble since the average results will be almost the same as the results of each subnetwork. Therefore, as for ensemble of CNNs, it would be better to increase the variance of subnetworks to a proper level.

In [10], they also created an ensemble of CNNs based on the following equations:

$$|E = \sum_{j=1}^{M} \left\| y_i - \sum_{i=1}^{N} a_i L_i^j \right\| - \lambda \sum_{i=1}^{N} |a_i|$$

$$\arg min_{I \subset \{1,\ldots,N\}} \left[\frac{1}{M} \sum_{j=1}^{M} \delta\left(y_j - \arg max \sum_{i \in I} L_i^j \right) \right] - \lambda |I|$$

where E is the error function of their ensemble, i represents the i-th subnetworks and j represents the j-th image from their training dataset. After minimizing the error function and get the trained a_i , subnetworks with a_i smaller than a threshold are eliminated. The rest of subnetworks will be optimally combined following the second equation. Generally, the second equation is to select a subset of networks where the classification accuracy is high, and the number of subnetworks are as low as possible. However, to create such highly optimized ensemble of CNNs, we need to sacrifice a lot of time during the training stage. Although each subnetwork is not deep, it is still computation complicated to train N number of such networks. To have a flexible selection of subnetworks, N cannot be too small. Otherwise, there may not be obvious improvement shown on the ensemble of networks.

2) My work

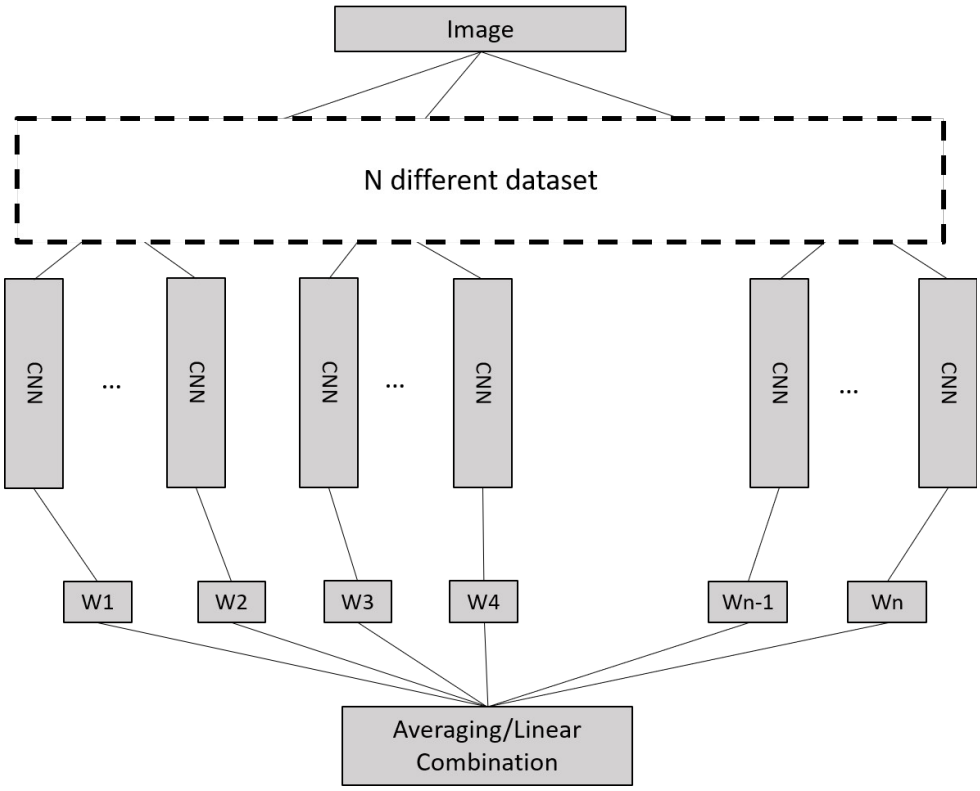

Figure 8 Proposed Multi-column CNN

In my work, I replaced the pre-processing steps with N different datasets. There were basically two reasons to do so. First of all, the dataset already contains various challenging conditions. The variance among the training images are large enough that we don't need to add any extra effect. Second, experiments on the proposed single convolutional neural

network showed that there was no significant increment in accuracy. Creating N different dataset could make sure that each single CNN was trained on different data.

Then I added a weight after each single network. The weights were used to find the most optimized linear combination of all the CNNs. It was proposed that linear combination is no better than simply averaging score vectors[16]. However, the conclusion was based on the MNIST digit dataset. For traffic sign classification, we might expect linear combination to have a better performance than simple averaging.

D. Random Forest

1) Decision Trees

Tree structure is a one kind of data structure that imitates the natural shape of a tree in real world. A tree consists of nodes and edges. In Figure 9, every circle represents a node while the lines between any pair of nodes are edges. Node A is called the **root** of the tree, while node E, F, C, D are **leaf** nodes. It is just like a tree in real world with bottom up.

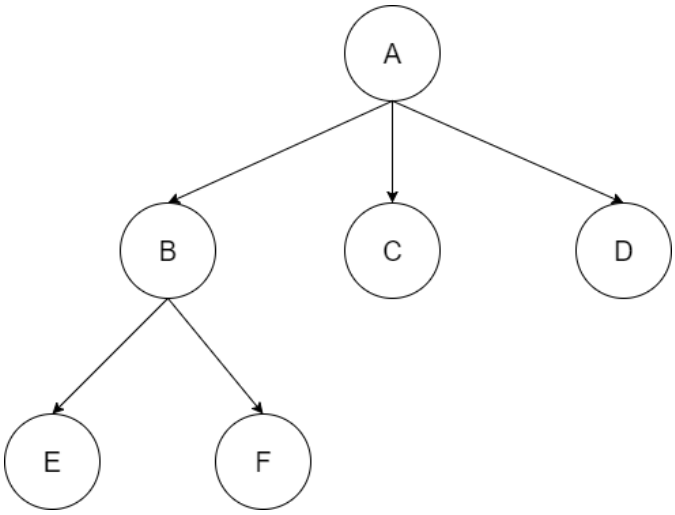

Figure 9 A sample tree diagram

There are many kinds of tree in graph theory, including but not limited to binary tree, B-tree, decision tree[17]. Some of the tree structures are applied for better management of data. However, in the tasks of classification, we are using decision tree that will help us to make decision instead of managing data. When a data enters the root of a decision tree, it will be passed to one of the child nodes based on some kind of criteria, until it reached a leaf node, where there is no more child node. The leaf node will return single decision or a series of probabilities, indicating the class of the input data.

One of the biggest question to concern when building a decision tree is “how do we split the data at a particular node”. The method to split nodes are called **splitting rules**. We may split the data evenly using a single threshold. We can also split the data based on some features. There are certainly many ways to build a tree and different ways generate different tree structures. Some of them may be effective while others cannot make accurate decision. Therefore, we want the splitting rule to generate the most effective decision tree. Based on this consideration, there are 3 major kinds of algorithms, ID3[19], C4.5[20] and Classification And Regression Tree(CART)[21].

Before introducing different algorithms, we define a set of training data S at every node. For every sample in S, there are attributes(features) denoted as A. There are totally X classes in the training data.

ID3 algorithm

ID3 algorithm is a kind of greedy algorithm that select the best attribute to split every node. It uses entropy $H(S)$ and information gain $IG(A_i)$ as evaluation tools.

$$H(S)=-\sum_{x\in X} p(x)\log_2 p(x)$$

$$InfoGain(S,A)=H(S)-\sum_{S_i\subseteq S} p(S_i)H(S_i)$$

where S_i denotes a subset of S after splitting based on attribute A. The entropy tells us the “purity” of information in this node. If all the samples belong to the same class, H(S) will be minimized as zero.

At every node, the algorithm evaluates all attributes and find one that maximize $IG(S,A)$, which means that the information gain after splitting is guaranteed to be max. After an attribute is chosen, it will no longer be considered in the following split. In a word, the ID3 algorithm minimizes the entropy as quickly as possible by using the best possible attribute to split the node.

C4.5 Algorithm

C4.5 algorithm was proposed based on the work of ID3 algorithm. The splitting criterion is normalized information entropy, which is a little bit different from ID3.

$$SplitInfo(S, A) = -\sum_{S_i \subseteq S} p(S_i) \log_2 p(S_i)$$

The information gain in C4.5 is defined as:

$$GainRatio(S, A) = \frac{InfoGain(S, A)}{SplitInfo(S, A)}$$

Then the attribute that generates the largest gain ratio will be selected.

Classification And Regression Tree(CART) Algorithm

Instead of using information entropy as splitting criterion, CART algorithm uses Gini to split nodes.

$$Gini(S) = 1 - \sum_{x \in X} p^2(x)$$

Gini also evaluates the “purity” of information in a node. If all the data belong to one class, Gini is minimized as zero. However, if the data are uniformly distributed in X classes, Gini is maximized.

There is another major difference between CART and ID3, C4.5 algorithms. While ID3 and C4.5 may build non-binary trees, CART will only build binary tree during training. A binary tree means that every node either has two child nodes or no child nodes. The training process is as follows.

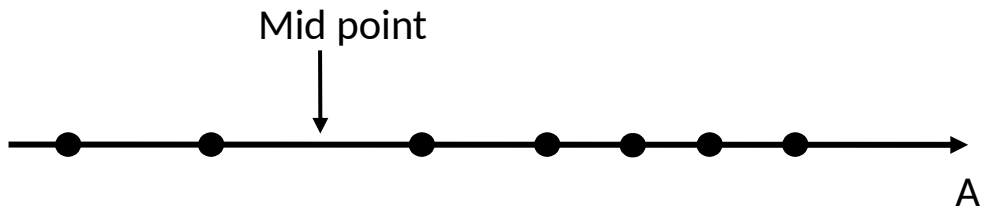

Figure 10 How to split the nodes

Assume we have X samples distributed along the attribute A dimension. The program will consider every middle point between two neighboring samples and then calculate the Gini index. If using one particular middle point as threshold can generate the maximum Gini decrement, this middle point is adopted as the threshold. The node is split into two child nodes. Then the same process is applied to each child node, which is a recursive procedure. At the end, the splitting will stop when Gini can decrease no more or certain pre-set rules are satisfied.

CART is one of the most popular algorithm in building a tree for classification and regression. In the Random Forest that we later built, we also applied CART algorithm in seek of the best performance.

2) Random Forests

Random Forest was first proposed in paper by Leo Breiman[22], where he built uncorrelated decision trees and combined them together to form a forest. Random Forests adopted CART like algorithm, which is slightly different from the original version.

Algorithm

The algorithm of training Random Forests started from a bootstrap aggregation[23]. When we only have a number of weak classifiers, we will consider forming a strong classifier by combining these weak classifiers. There are many ways to train and combine weak classifiers. For example, AdaBoost is an algorithm that applies boosting[24] to combine weak classifiers into a strong one. Bootstrap is another way to generate different training set for single decision tree.

Assume we have training set $S = \begin{matrix} s \\ (¿¿1, y_1), (s_2, y_2), \cdots, (s_N, y_N) \\ ¿ \end{matrix}$ where y represents a label.

Assume we have *K* number of decision trees in the Random Forests. Assume there are *M* attributes(feature dimensions) in each training sample.

1. Randomly select N samples with replacement from the training set.
2. Use the selected N samples to build a decision tree. At each node, randomly select *m* attributes from *M* total attributes. Only consider these m attributes to make splitting. Split the node when Gini Gain is maximized.

3. Repeat step 1 & 2 for *K* times until all decision trees are built.
4. Combine the *K* decision trees into Random Forests, where the final prediction is decided by voting from all trees.

The difference of Random Forests and CART algorithm is that the former does not consider all attributes at each node. It randomly considers a subset of *M* attributes, which means that there is a possibility that certain attribute will be selected more than once.

The advantage of using bootstrap to form Random Forests is that this algorithm can reduce the variance between decision trees without increasing the bias[25]. As *K* different training sets are generated for training, we are trying to make sure that the correlation between two tree models are low. If we use exactly the same training set for *K* decision trees, they may be highly correlated to each other. As will be explained, if the sub-classifiers are highly correlated, combining these weak classifiers may not generate satisfying increase in overall performance. Therefore, by applying bootstrap algorithm, we de-correlate the tree models in order to get powerful Random Forests.

Technical consideration

There are a lot of parameters when training Random Forests. Some key parameters include number of trees, number of features, maximum depth etc.

The number of attributes to consider *m* is one of the most important parameters. However, the size of *m* is an empirical choice[26]. Basically, the selection of *m* does not have a rule to follow. In many classification problems, the number of attributes to consider is set to $\sqrt{M}$ where *M* is the total number of attributes. Sometimes, setting *m* to other values like $\log_2 M$ gives better results. Although training will be faster when *m* is smaller, the risk of underfitting increases as well. Therefore, we need to do a number of experiments to get the best number of attributes.

The number of decision trees also matters a lot. To form effective Random Forests, the number of trees can range from a hundred to nearly a thousand. Nevertheless, it is not the truth that the more trees there are, the better the performance will be. On the one hand, having more trees continuously sacrifices computation costs in both training and testing stages. On the other hand, the increase in classification accuracy can be trivial or even worse. This is

because overfitting and highly correlated trees may be more likely to appear during this process. In this project, we have referred to several papers about traffic sign classification using Random Forests and adopted some of the parameter settings[8][25][26].

Other parameters to be considered includes maximum depth, class weights etc. Based on experiences, these parameters have relative less significant impact on the performance. The maximum depth has impact on whether the decision trees overfit or not. Class weights shall be carefully designed when the training data is highly imbalanced. By default, every sample has equal possibility to be drawn in the bootstrap algorithm. But we could manually adjust it according to the distribution of samples.

IV. DATASET

A. IEEE Video and Image Processing Cup Dataset

The IEEE Video and Image Processing Cup 2017 provided a large road traffic sign dataset. This dataset consists of videos whose sequences include processed traffic videos and synthesized videos. There are 12 types of simulated environmental conditions, including decolorization, lens blur, codec error, darkening, dirty lens, exposure, gaussian blur, noise, rain, shadow, snow and haze. Examples are shown in fig. 8. These challenging conditions span a wide range of challenging levels from mild to severe. There are 5 levels of severity in total.

Figure 11 All challenging condition

00-no challenge
01-decolorization
02-lens blur
03-codec error
04-darkening
05-dirty lens
06-exposure
07-gaussian blur
08-noise
09-rain
10-shadow
11-snow
12-haze

Figure 12 five levels of severity

There are two groups of classes in this dataset: real data and unreal data. Real data is captured from real world. Unreal data corresponds to synthesized sequences generated in a virtual environment. There are 49 real sequences and 49 unreal sequences that contains no challenge. Each video sequence is processed by all types of challenging conditions. Totally, there are 5733 video sequences. The dataset is separated into 70% training videos and 30% testing videos. There are 300 frames in each video sequence. Some of the frames contain traffic sign while others not. Although many traffic signs appear in the video sequences, only 14 classes of traffic signs were defined.

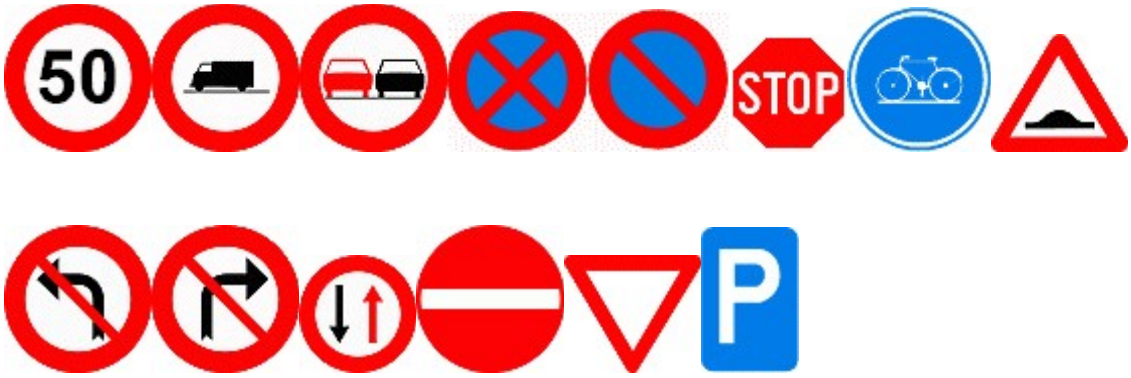

Figure 13 From left to right, top to bottom: speed limit, goods vehicles, no overtaking, no parking, stop, bicycle, hump, no left, no right, priority to, no entry, yield, parking

B. German Traffic Sign Recognition Benchmark

The German Traffic Sign Recognition Benchmark(GTSRB) is an official dataset available years ago. It includes 39,209 training images in 43 classes and 12,630 testing images. Compared to the VIP Cup Dataset, it has the following differences.

More training images

The GTSRB has over 10,000 more training images than our self-generated dataset. But the number of testing images are similar. More training samples means that the training space is wider and more inclusive. This can be one of the factor leading to higher classification accuracy.

More balanced data

The sample amounts of all classes do not vary a lot. The samples are carefully balanced so that the training data are sufficient for every class. The dataset also provides samples in different scales so that no matter a sign is large or small, it can be accurately classified. It is very important to consider this because in reality, signs can be in any scales, regarding its distance to the driver.

V. EXPERIMENT

A. Data Extraction and Augmentation

As the VIP dataset only provides videos, we had to extract traffic-sign images from the videos. Our extraction method is very simple. First, we extract frames, where the road traffic-sign regions are larger than a threshold. A very small road-sign region is not useful, which should be very far away from the vehicle concerned. Then, we crop the road signs from the extracted images according to the ground truth provided. Note that we enlarge the true bounding boxes by 6-pixels in width and height. The resulting images are then normalized to 48×48 pixels to fit the network. After these steps, we generated 23,311 training samples and 12,159 testing samples.

However, statistical analysis showed that Classes 1 and 5 have around 3,000 training images each. Classes 7 and 12 have less than 600 training images each. The training data is severely unbalanced.

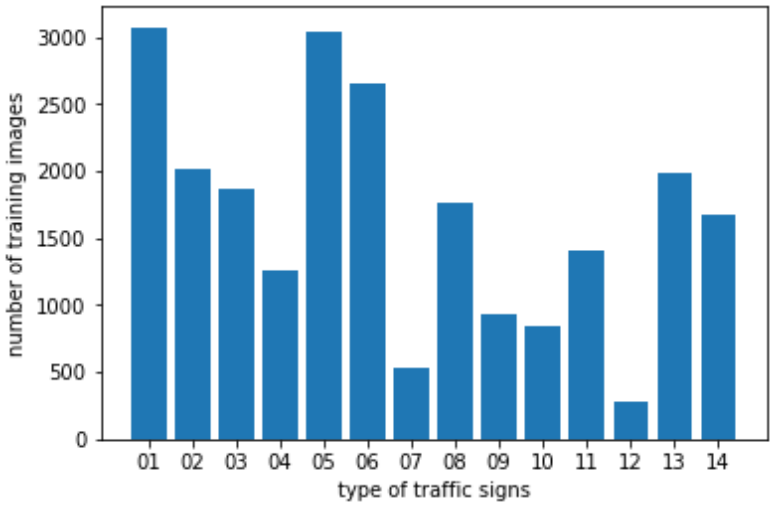

Figure 14 Histogram of dataset

Therefore, we modify the training data by randomly picking out some images from Classes 1 and 5, and extracting more images from Classes 7 and 12 from the videos. It is very difficult to fully balance the data for the respective classes because some traffic signs appear a few times only, while some appear frequently. Besides, since the traffic sign detector proposes many false alarms, we added another 8000 negative samples into the training dataset. These negative samples are false alarms proposed by the detector, which could be anything from simple background to the window of buildings. The final dataset consists of around 35000 training images and 12159 testing images.

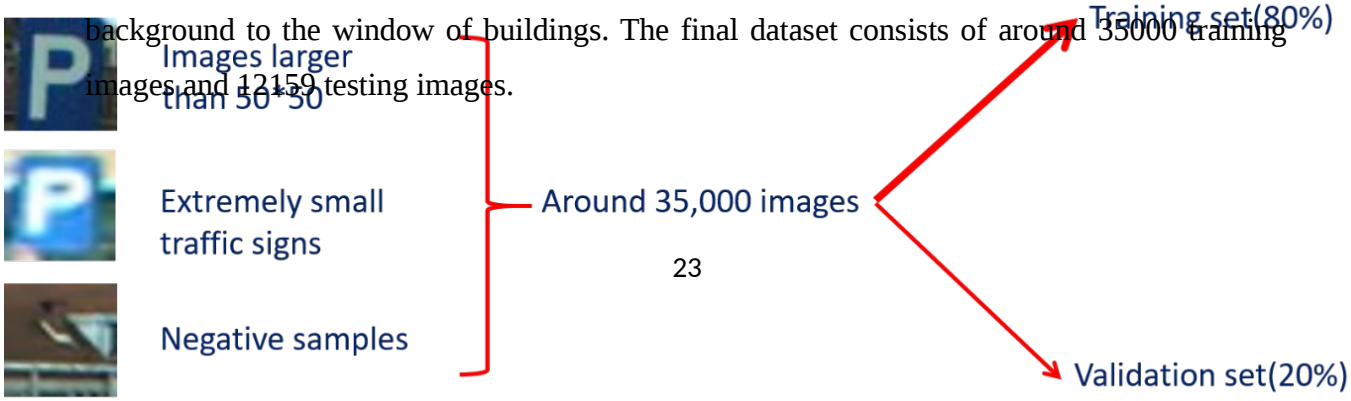

Figure 15 Data preparation procedure

As for the ensemble of CNNs, we prepared 5 different training sets and validation sets. First of all, we randomly divide the existing training dataset into 5 subsets. Then we combined any 4 of them into training set and the rest one as validation set. Therefore, we came up with 5 different pairs of training set and validation set.

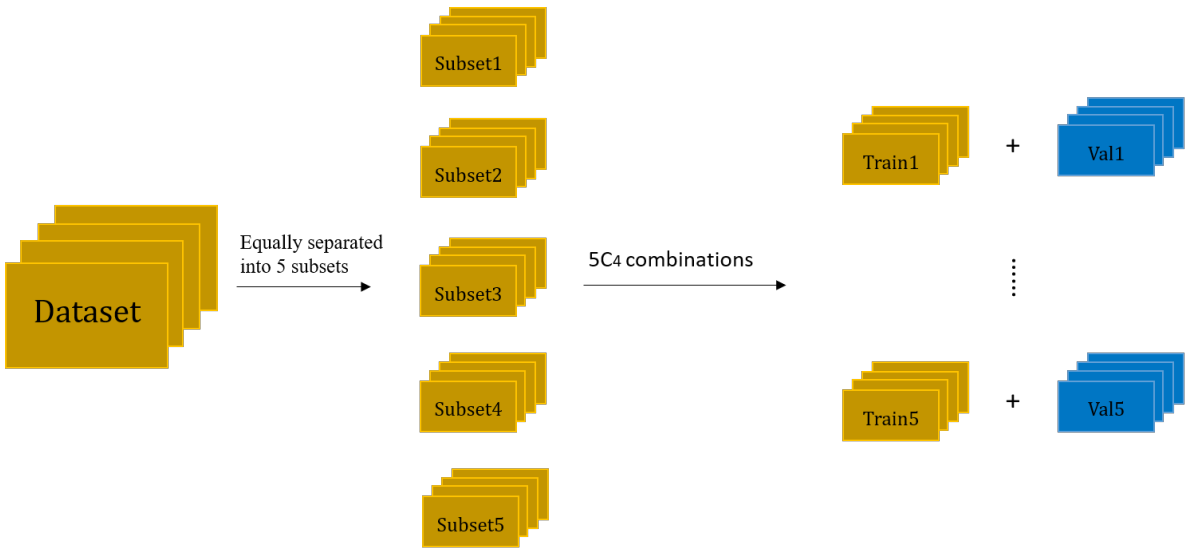

Figure 16 Data preparation for ensemble of CNNs

When training our neural network, we consider multiple data augmentation methods, although various challenging conditions and levels of challenges already exist in the training samples. In fact, we found that only about 2% of the training data does not include challenging conditions. In the experiments, we have tried the following augmentation: (1) Random Crop: the training samples are randomly cropped to 40×40 pixels. (2) Random Saturation: We change the saturation of the RGB images by a random factor between 0.9 and 1.1. (3) Random Brightness: We randomize the brightness of images within the range [B-63, B+63] (4) Random Contrast. The range of random contrast is [0.2, 1.8]. (5) Horizontal Flip. Images are randomly flipped along their vertical dimension. These methods, except the last one, can improve the performance of the network by 0.5%~1%. The reason for this is that some of the road signs are not symmetrical, i.e. the left half is different from the right half.

B. Training of subnetworks

The network is trained using the Adam Stochastic Optimization [17]. Batch size was fixed to 128. The network is trained for 4100 steps with a learning rate of 1e-3 and then 5000 steps with a learning rate of 1e-4. The dataset was separated into 80% training set and 20% validation set. As we wanted to see the effects of negative samples on our networks, we also trained excluding negative samples. The evaluation results are given in the next section.

Training of ensemble of CNNs

We trained 5 single CNN as mentioned before. These 5 networks used different dataset so that the correlation between them were decreased. After that, the score vectors from 5 subnetworks were collected. We would like to compare the averaging method and linear optimization method to see if there is any obvious improvement coming from the latter. Therefore, the score vectors were averaged at the beginning.

$$n = argmax\left(\sum_{i=1}^{5} w_i \left[s_{i,1}, s_{i,2}, \ldots, s_{i,15} \right]\right) w_i = \frac{1}{5}$$

Then, we trained the ensemble of CNNs by minimizing the following error function:

$$E = \sum_{k=1}^{K} \left\| y_i - \sum_{i=1}^{5} w_i \acute{S}_i \right\|$$

$$w_i \in [0.150, 0.250], \sum_{i=1}^{5} w_i = 1$$

C. Image Processing for Random Forests

1) Contrast-Limited Adaptive Histogram Equalization

Contrast-Limited Adaptive Histogram Equalization(CLAHE)[27] was proposed based on the original histogram equalization. Ordinary histogram equalization first get a histogram of distribution of all pixels in the image. It tries to find a mapping function that can transform the original histogram into a uniformly distributed histogram. Figure 17 shows a successful Histogram Equalization.

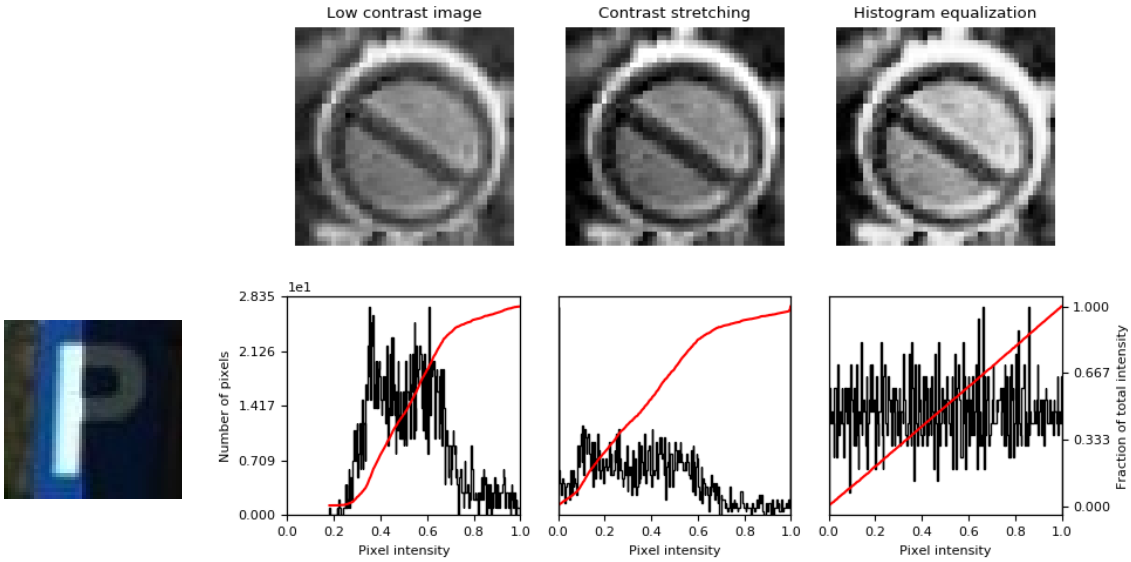

Figure 17 A successful Global Histogram Equalization

However, as Global Histogram Equalization focuses on the overall enhancement, it fails to adapt to local contrast features of the images[28]. For example, majority of the pixels are within a small range of gray-level values, which means that they have a high frequency. A small part of the pixels is largely different from the majority. Then after Global Histogram Equalization, the low-frequency pixels may be heavily affected and the contrast information within a certain area can be distorted. Figure 18 shows an example where Global Histogram Equalization does not work well.

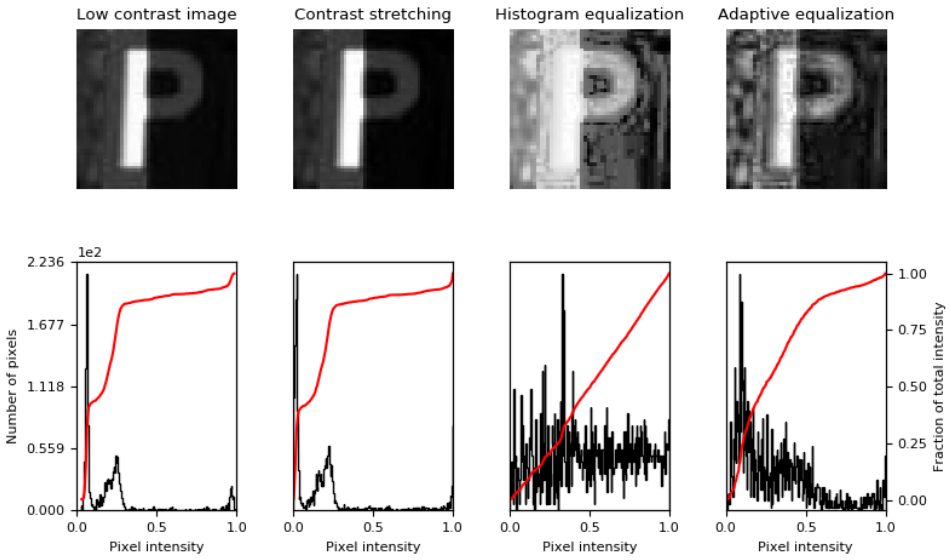

Figure 18 An example where Global Histogram Equalization does not work well

Therefore, Contrast-Limited Adaptive Histogram Equalization[27] were proposed to adapt to local contrast features. With a pre-set kernel size, Histogram Equalization is first done in each local area. For each pixel, it will be located at the central point of the kernel. Then we apply Histogram Equalization within the kernel region. The new pixel value can be found from the new histogram. All pixels in the image will be calculated following this procedure, considering only neighbouring pixel intensities. Therefore, the local contrast can be enhanced. Even if the region is especially darker or brighter than other parts of the image, its local contrast information can be well kept. In Figure 18, we can clearly see that the Adaptive Histogram Equalization has better effects than Histogram Equalization.

D. Feature Extraction for Random Forests

1) Histogram of Oriented Gradients(HoGs)

Histogram of Oriented Gradients is a feature descriptor widely used in Computer Vision and Pattern Recognition. It was first proposed for human detection[4]. Later people also found it an effective feature for recongnition. HoGs, as the name implies, is formed by concatenating histograms together. Each individual histogram is generated from a **block** of the image.

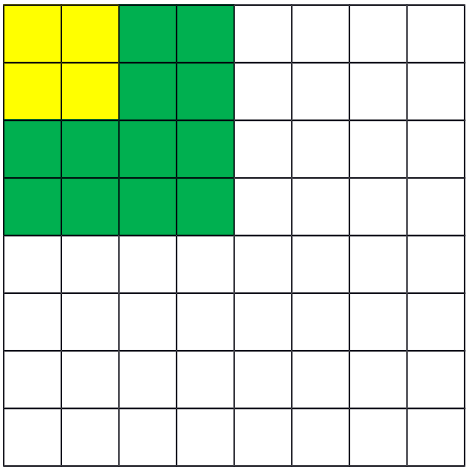

Figure 19 Concepts of Block and Cell

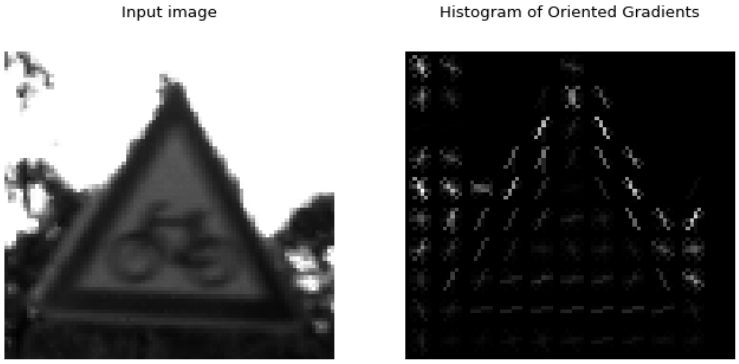

Figure 20 HoG feature after visualization

For an input image, we divide it into blocks of regions. In Figure 19, the green area represents an 4x4 **block**. Inside each block, we subdivide it into cells. Normally, one block contains integer number of cells. In Figure 19, the yellow region represents a 2x2 **cell**. In each cell, we apply two operators:

$$I_x = [-1,0,1]$$

$$I_y = [-1,0,1]^T$$

The we get the magnitude and gradients at each pixel location:

$$I_{mag} = \sqrt{I_x^2 + I_y^2}$$

$$I_\theta = \tan^{-1}\left(\frac{I_y}{I_x}\right)$$

Inside each cell, the unsigned gradient can vary from 0 to 180. We evenly subdivide the orientation domain into 9 different segments, [0,20], [20,40],…… The we categorize each pixel orientations into corresponding segments. The magnitude of pixels in the same segment will add up together. This is called weighted votes. Therefore, in each cell, we will have a 9-bin histogram where each bin corresponds to a range of orientation and its magnitude correspond to the sum of pixel magnitudes within the region.

For each block, we concatenate the histograms of different cells into one single histogram. The dimension will thus be 4 times. It was also found that overlapping between blocks can significantly improve the effectiveness of HoG feature[4]. Normally, the overlap between neighboring blocks is 50%.

HoG is a good feature to describe local appearance and shape by using gradient intensity information. HoG feature also ignores the color information of the original image because

color information doesn't help a lot for most of the time. Thus, the dimension of the feature vector decreases as a result, which saves computation costs. It is much faster than SIFT feature.

VI. RESULTS

A. Single Convolutional Neural Network

The model has been trained 5 times. Table 1 shows the results. In general, the accuracy with the validation set is higher than 98%, while the accuracy with the testing set is about 92%.

Challenge type	Accuracy	Challenge type	Accuracy
No challenge	99.19	Gaussian blur	91.41
Decolorization	96.99	Noise	96.73
Lens blur	94.07	Rain	92.27
Codec error	59.71	Shadow	97.59
Darkening	96.48	Snow	95.62
Dirty lens	98.36	Haze	95.19
Exposure	93.04	Overall	92.21

Table 1: Experiment results in terms of the recognition rates for the validation set and testing set, which are repeated five times with samples selected randomly.

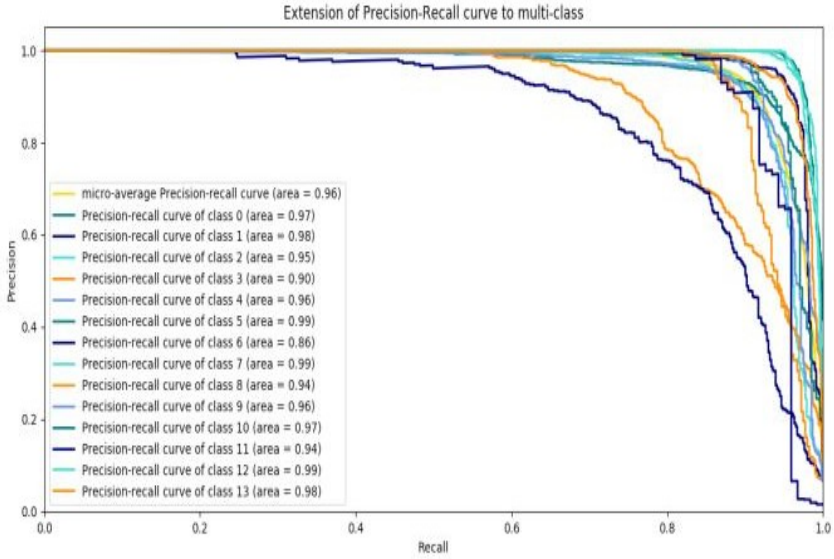

Figure 21 Precision-Recall Curve of single CNN

We also trained the network excluding any negative samples. The results are shown below.

Dataset	Including Negative Samples	Excluding Negative samples
Accuracy	92.21	94.10

Table 2 Single CNN on two kinds of datasets

In order to improve the performance, we have collected all the misclassified traffic-sign images. We found that codec error (challenge 03) is the most influential condition on our network. Around 47% of the misclassified images belong to the codec-error challenge, among which 74% have a level-3 or higher challenging level. Figure 2 shows some images of the codec-error challenge, and some images belonging to other challenges. Figure 3 shows the number of misclassified images for each of the 12 challenges. We can see that, except for codec error, our network can have nearly equal performance for the other challenge types. In Fig. 4, we show the number of misclassified images for each of the 14 road-sign classes. We observe that Road-sign Classes 1, 4, and 5 have higher numbers of misclassified images.

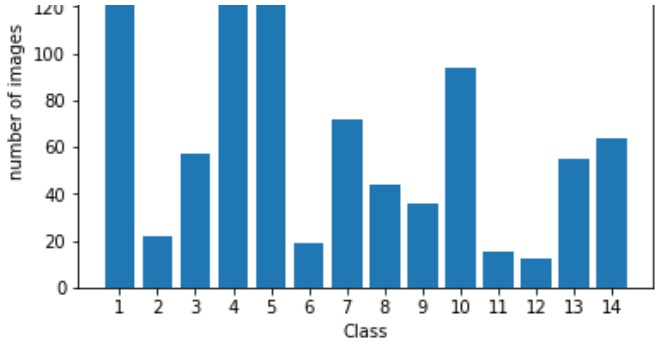

Figure 22 The number of misclassified images for each class of the 14 road signs.

In order to show the effect of batch normalization and ReLU in this experiment. We tried different combinations of activation function and batch normalization. The experiment results are shown below.

Combination	ReLU only	tanh + BN	ReLU + BN
Accuracy	77.53%	83.38%	92.21%

Table 3 Different activation and batch normalization combination

B. Ensemble of CNNs

We trained both average ensemble and linearly optimized ensemble in order to see whether significant improvement could be made.

When negative samples are included during training stage, the ensemble of CNNs using averaging method reached an accuracy of 92.94%. The linear combination methods reached 93.13% classification accuracy. Using linear combination slightly increased the accuracy by 0.19%, which corresponds to about 23 testing images.

When negative samples were eliminated, the ensemble of CNNs using averaging method reached an accuracy of 95.17%. The linear combination methods reached 95.19% classification accuracy. There seemed no difference between these two methods in dealing with multi-column CNNs.

Dataset	Including negative samples		
Network	Single CNN	Ensemble – Averaging	Ensemble – linear combination
Accuracy	92.21%	92.94%	93.13%

Dataset	Excluding negative samples		
Network	Single CNN	Ensemble – Averaging	Ensemble – linear combination

Accuracy	94.10%	95.17%	95.19%

Table 4 Performance of Ensemble of CNNs

We noticed that for both kinds of datasets, there is no significant increase in the classification accuracy. However, we did saw 1-2% increment. We have been trying to come up with a possible explanation of this results.

First of all, the subnetworks of the ensemble may be highly-correlated. Although we did split the dataset into 5 different sets, overlaps between them cannot be omitted. For example, each image appeared as training image in four of the subnetworks and as validation image in only one subnetwork. The first four subnetworks must have learned the same feature of this image. Therefore, there might be some strong correlation between these subnetworks. But as we didn't evaluate the correlation in number, we could just make some speculation.

Second, the linear combination method seemed working out better in the first dataset compared to the other. Note that the first dataset contained over 8000 negative samples. These negative samples varied a lot in features. Unlike negative samples, traffic signs looked more alike even if they come from different classes. For example, traffic signs all followed specific shapes like circle and rectangle and had limited colors, mostly red and blue. However, background could be in any color and any shape. Although we didn't visualize their highly abstract feature from the CNNs, we might still expect them to be a lot different from features of traffic signs.

C. Random Forests

1) GTSRB

For the GTSRB dataset, there are 43 classes, 39,209 training images and 12,630 testing images. The organizer also provided pre-calculated HoG 1, 2 and 3 for both training and testing data.

Name	Dimension	Cell Size	Block Size	Overlap	Bin	Orientation
HoG1	1568	5x5	10x10	50%	8	Unsigned
HoG2	1568	5x5	10x10	50%	8	Signed
HoG2	2916	4x4	8x8	50%	9	Unsigned

Table 5 HoG parameters

All images are resized to 40x40 pixels using a bilinear interpolation. We didn't apply any dimension reduction algorithms on feature vectors.

According to [8], they reached the best performance by trying out different combination of number of trees, number of features and number of samples. In our experiments, it took a lot of time to change the number of samples. Therefore, we only finished experiments on number of features and number of trees.

During training, we also set maximum tree depth and minimum samples at each node to avoid overfitting problem. If we split nodes until there is only one sample, the leaf will return a probability equals to 1. However, overfitting can be a significant problem limiting the classification accuracy. After several trials, we found that the maximum depth of a tree is normally around 40. So we reduce the max depth to 30 and minimum samples at node to 10-20.

HoG	Number of Features	Number of Trees	Classification Accuracy(%)
1	Log2	50	90.36
		100	92.18
		500	93.21
	sqrt	50	93.19
		100	93.80
		500	94.53
	100	50	93.98
		100	94.46
		500	95.56

HoG	Number of Features	Number of Trees	Classification Accuracy(%)
2	Log2	50	93.91
		100	94.93
		500	95.20
	sqrt	50	95.68
		100	96.15
		500	96.77
	100	50	95.66
		100	96.28
		500	96.92

HoG	Number of Features	Number of Trees	Classification Accuracy(%)
3	Log2	50	90.19
		100	91.87

		500	92.82
	sqrt	50	93.16
		100	93.76
		500	94.33
	100	50	93.37
		100	94.23
		500	

Table 6 Results for GTSRB

The highest accuracy was achieved when we used HoG2 + 500 trees + 100 features at each node. The classification accuracy is 96.9%.

HoG1 and HoG3 do not have the same satisfying results as HoG2 does. In fact, although our results are better than that reported in the original paper, the fact that we have more training data cannot be ignored. In [8], they only had over 20,000 training images. The amount of our training images is 1.6 times of theirs. With such a large amount of data, we were expecting the results to be significantly improved.

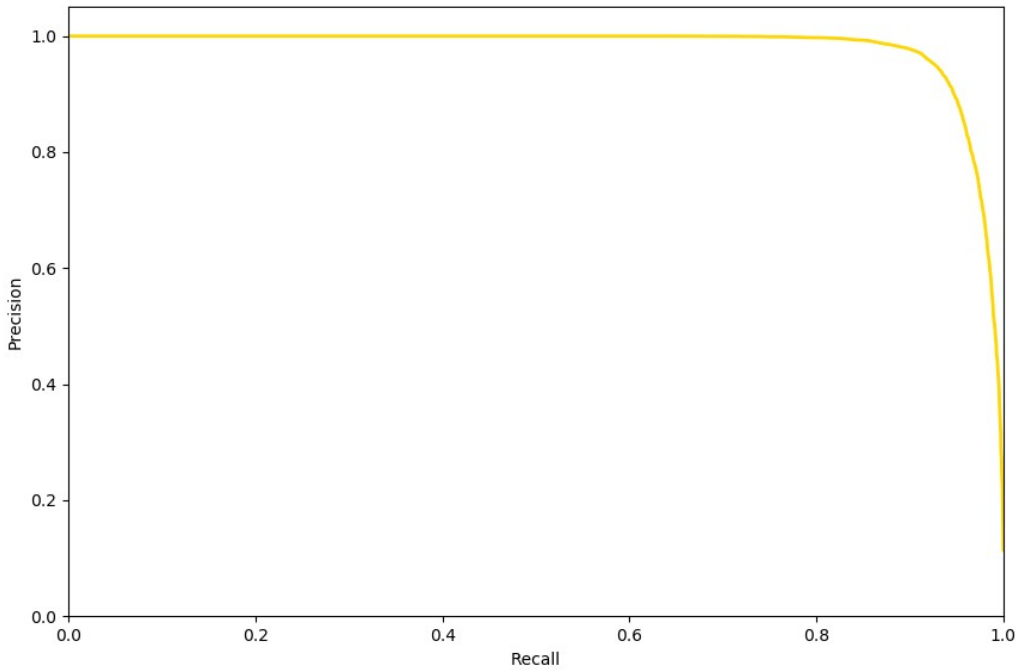

Figure 23 PR Curve

2) VIP Cup

The Random Forests model was also tested on the VIP Cup dataset. It reached an accuracy around 83% using HoG feature.

HoG block	Cells/block	nbin	No. of Trees	No. of Features	Accuracy
6x6	2x2	9	500	Sqrt	82.76
8x8	2x2	9	500	Sqrt	81.50
Concatenate above HoGs			500	Sqrt	83.54

Table 7 Results on VIP Cup

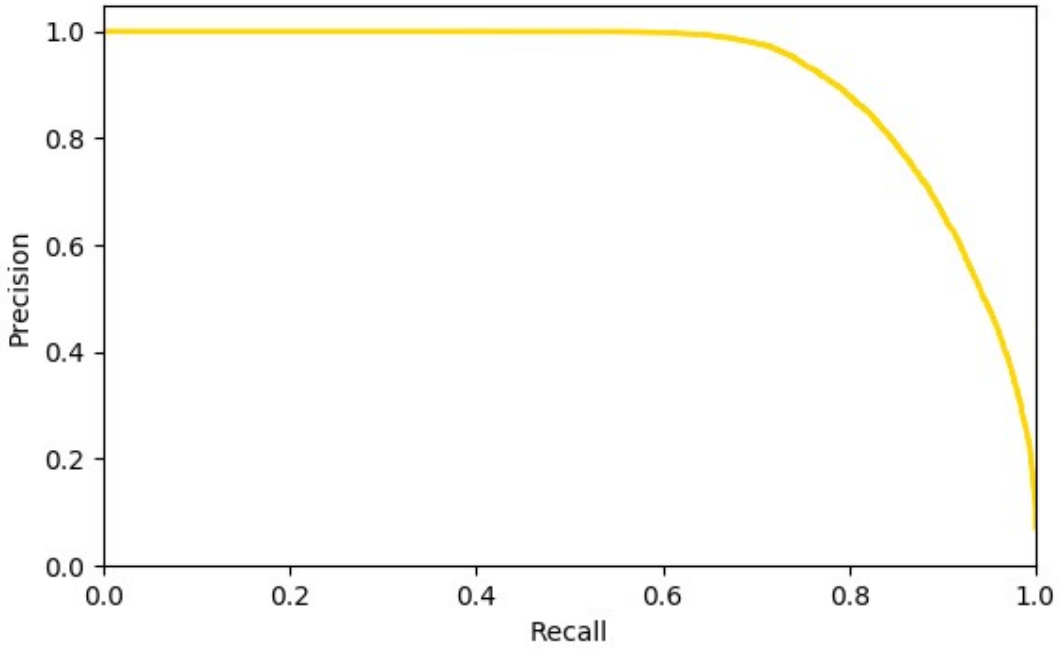

Figure 24 PR Curve on VIP Cup

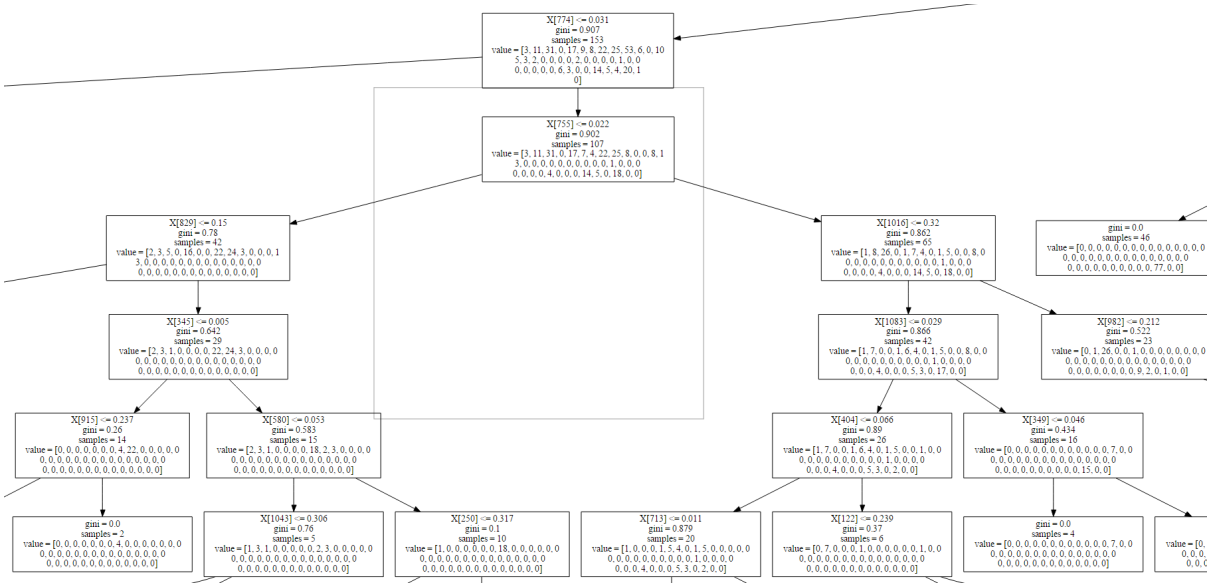

Figure 25 Part of the tree

We found that each individual tree has only about 3% accuracy, while the whole Random Forests can boost the value to over 80%. It was really an impressive improve by simply combine many tree structures together. This also implies that the trees are highly uncorrelated and the bias is little so that the overall performance could so much better than each individual classifiers.

VII. OBSERVATION & DISCUSSION

The above evaluation shows that our proposed model can achieve satisfactory performance, but some improvement still can be made. There is a great gap between the validation accuracy and test accuracy, which should be reduced as much as possible. As is shown in Fig. 4, our network does not perform well in classifying Classes 1, 4, and 5. The reason for this may be as follows. First, Classes 4 and 5 are very similar to each other. The only difference is the red, thick, diagonal line crossed from the upper right to the lower left in the road sign. Therefore, the network may mix up these two classes when a challenging condition is applied. As for Class 1, the problem may be caused by the selected training samples. There are 3 subclasses for Class1, which are the speed limit of 30, 50, and 70 km/h, respectively. Therefore, the training samples included may not be sufficient for all these 3 subclasses. Besides, occlusion may be another reason as a small number of the traffic signs were occluded in the video sequence

As for the codec error, it greatly destroys the color and shape information of an image. Thus, it becomes very difficult for the network to extract useful information from images under this challenge. Although there are some training images with codec error, there is no guarantee that these images will help improve the classification performance. These images may also have negative influence on our network if they are used for training.

We have noticed some additional road signs in the videos, which do not belong to any of the 14 classes. It is difficult to find them out as they appear occasionally, and their number is small. It will take a very long time to pick them out manually and discriminate them from the other 14 classes. Therefore, we simply ignore them during classification. If these additional road signs are to be considered, information, such as the location of their bounding boxes and their corresponding class numbers, must be available.

Last but not least, we also found another problem with the provided dataset. The videos are not synchronous. Sometimes, two videos of the same scene are not synchronous, but the provided ground truth does not discriminate them. Therefore, there are some incorrect ground truth.

In the Random Forests experiments, we didn’t use the bounding box information. Therefore, there might be more background pixels included during feature extraction and classification. These extra noises could be one of the reasons that our classification accuracy did not have huge improve. Besides, the original paper suggested that 100 training sample per tree would generate the best performance. However, we used the total amount of training samples for all trees, which could make the correlation between trees higher. Finally, the speed of Random Forests is much higher than the Convolutional Neural Networks. Random Forests showed its convenience, fastness and robustness to unbalanced data in this project. Although its performance was not as good as ConvNet on the VIP Cup dataset, there would still be some space to enhance the model.

VIII. JOINT CAPSTONE PROJECT WITH STUDENTS FROM UTS

This is a one-year international capstone project formed by students from PolyU and University of Technology Sydney. Through this one-year experience, I have learnt a lot in both academic knowledge and how to cooperate.

Starting from this April, our team worked on IEEE Video and Image Processing Cup 2017, Road Traffic Sign Detection under Challenging Conditions. Because we were separated by the huge ocean between Asia and Oceania, we were responsible for different methods. Jiawei and I were using deep learning, ConvNet to solve the problem while students at UTS applied traditional hand-crafted features and SVM. During the 4-month competition, we kept contact by using email and messenger. We kept updating our progress with students in UTS so that we could finally make a decision on which method should be used. Jiawei and I also got a chance to visit UTS and Sydney in person, which is the most exciting experience for me. We visited the campus of UTS and listened to Prof. Sean He's research group meeting. It was a great fun to open up horizons and see what other universities are doing.

During our visit in UTS, we made a presentation to the students and Prof. He. We clearly elaborated our work and results. We were very glad to see students from UTS in person so that we could have face-to-face discussion. During our conversations, we decided to continue working on traffic sign detection by applying different methods. Each of us use a different feature and classifier. After we went back to Hong Kong, we still kept good contact with Edward, talking about some technical problems and progress of project.

I truly appreciated this unique final year project from which I have learnt a lot. The international cooperation was also something special, making me feel total different from doing an individual final year project. I was very lucky to be selected to become a member of this team.

IX. CONCLUSION

In this project, we used Convolutional Neural Network and Random Forests to classify traffic signs under challenging conditions. The performance of single CNN is not bad. Creating ensemble of CNNs did improve the performance by 1-2% in accuracy. However, the correlation between models of the ensemble might have restricted the meaning of creating

ensemble. To further improve the performance, modifications on network architecture and other more advanced techniques should be considered.

As for Random Forests, it had a good performance on the GTSRB dataset, where it achieved 96.9% accuracy. However, its performance on the VIP Cup dataset is not that satisfying. Although some pre-processing techniques has been applied, they were not effective to all kinds of challenging conditions. Random Forests is a strong classifier. However, the hand-crafted feature might not be useful as the highly abstract feature in the Convolutional Neural Network. To further compare the classification ability of ConvNet and Random Forests, we may consider removing the fully-connected layer in ConvNet. Then use the extracted feature and Random Forests to form a system.

X. REFERENCE

[1] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, “Detection of traffic signs in real-world images: The German traffic sign detection benchmark,” in Proc. IJCNN, Aug. 2013, pp. 1–8, IEEE.

[2] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, “The German traffic sign recognition benchmark: A multi-class classification competition,” in Neural Networks (IJCNN), The 2011 International Joint Conference on, July 2011, pp. 1453–1460.

[3] J. Greenhalgh and M. Mirmehdi, “Real-time detection and recognition of road traffic signs,” *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 4, pp. 1498–1506, Dec. 2012.

[4] N. Dalal and B. Triggs, “Histograms of Oriented Gradients for Human Detection,” Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2005

[5] M. Mathias, R. Timofte, R. Benenson, and L. V. Gool, “Traffic sign recognition - how far are we from the solution?” in Proceedings of IEEE International Joint Conference on Neural Networks, 2013

[6] D. Cireşan, U. Meier, J. Masci and J. Schmidhuber, "Multi-column deep neural network for traffic sign classification", *Neural Networks*, vol. 32, pp. 333-338, 2012.

[7] P. Sermanet and Y. LeCun, "Traffic sign recognition with multi-scale Convolutional Networks," *The 2011 International Joint Conference on Neural Networks*, San Jose, CA, 2011, pp. 2809-2813.

[8] F. Zaklouta, B. Stanciulescu, and O. Hamdoun, “Traffic sign classification using K-d trees and Random Forests,” in Proc. IJCNN, Aug. 5–31, 2011, pp. 2151–2155.

[9] Nanni, L., Ghidoni, S. and Brahnam, S. (2017). Handcrafted vs. non-handcrafted features for computer vision classification. Pattern Recognition, 71, pp. 158-172.

[10] D. G. Lowe. Distinctive image features from scale-invariant keypoints. IJCV, 60(2):91–110, 2004.

[11] H. Aghdam, E. Heravi and D. Puig, "A Practical and Highly Optimized Convolutional Neural Network for Classifying Traffic Signs in Real-Time", *International Journal of Computer Vision*, vol. 122, no. 2, pp. 246-269, 2016.

[12] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In ICML, 2013

[13] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In ICML, 2015.

[14] D. E. Rumelhart, G. E. Hinton and R. J. Williams, “Learning Representations by Back-Propagating Errors,” Nature, vol. 323, pp. 533-536, 1986.

[15] C. M. Bishop. Pattern Recognition and Machine Learning. Springer, 2007.

[16] U. Meier, D. C. Ciresan, L. M. Gambardella, and J. Schmidhuber, “Better digit recognition with a committee of simple neural nets,” in Proc. IEEE Int. Conf. Doc. Anal. Recognit. (ICDAR), Beijing, China, Sep. 18–21, 2011, pp. 1250–1254.

[17] P. E. Utgoff, “Incremental induction of decision trees,” Machine Learning, vol. 4, pp. 161-186, 1989.

[18] D. Kingma and J. Ba. Adam: A method for stochastic optimization. ICLR, 2015.

[19] J. Quinlan, Learning Efficient Classification Procedures and Their Application to Chess end Games, Learning Efficient Classification Procedures and Their Application to Chess end Games, Morgan Kaufman, San Francisco, CA, 1983.

[20] J. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufman, San Francisco, CA, 1993

[21] L. Breiman, J. Friedman, R. Olshen, C. Stone, Classification and Regression Trees, Chapman and Hall, New York, 1993.

[22] L. Breiman, “Random Forests,” Machine Learning, vol. 45, no. 1, pp. 5-32, 2001.

[23] L. Breiman, ªBagging Predictors,ºMachine Learning,vol. 24, no. 2, pp. 123-140, 1996

[24] Y. Freund and R.E. Schapire, “A Short Introduction to Boosting,” J. Japanese Soc. Artificial Intelligence, vol. 14, no. 5, pp. 771-780, 1999.

[25] P. O. Gislason, J. A. Benediktsson, and J. R. Sveinsson, “Random forests for land cover classification,” Pattern Recognit. Lett., vol. 27, no. 4, pp. 294–300, Mar. 2006
[26] F. Zaklouta and B. Stanciulescu, “Real-time traffic-sign recognition using tree classifiers,” IEEE Trans. Intell. Transp. Syst., vol. 13, no. 4, pp. 1507– 1514, Dec. 2012
[27] K. Zuiderveld, “Contrast limited adaptive histogram equalization,” Graphics Gems, vol. 4, pp. 474–485, 1994.
[28] M. Abdullah-Al-Wadud, Md. Hasanul Kabir, M. Ali Akber Dewan, and Oksam Chae, “A dynamic histogram equalization for image contrast enhancement”, IEEE Trans. Consumer Electron., vol. 53, no. 2, pp. 593- 600, May 2007.

XI. SUMMARY

THE HONG KONG POLYTECHNIC UNIVERSITY

Department of Electronic and Information Engineering

EIE4430 / EIE4433 Honours Project / EIE4117 Capstone Project

1. Student Name: _____FENG Weixi_____ (Student No.: 14109625D)

2. Programme Code: 42477 / 42470 / 42480

3. Project Title: Road Traffic Sign Recognition under Challenging Conditions

4. Supervisor Name: Prof. Kin-Man Kenneth, Lam

5. Project summary (State clearly the project objectives and results achieved by yourself.)

This project aims at achieving satisfactory classification on traffic sign under challenging conditions, including but not limited to fog, haze, rain etc.

We applied two kinds of methods to solve the problem, deep learning and conventional classifiers. To be more specific, we implemented Convolutional Neural Network and Histogram of Oriented Gradients + Random Forests.

We made comparison between these two methods, getting into a deeper understanding towards machine learning and computer vision.

ConvNet has achieved better results in this project, a classification accuracy around 95%, while Random Forests achieved 83% on one dataset and around 97% on the other dataset.

Both methods have their advantages and disadvantages. Both of them are powerful and popular classifiers at present. Random Forest is fast and robust, while ConvNet is more accurate but computationally complex.

XII. DECLARATION OF ORIGINALITY

Except where reference is made in the text of this report, I declare that this report contains no material published elsewhere or extracted in whole or in part from any works or assignments presented by me or any other parties for another subject. In addition, it has not been submitted for the award of any other degree or diploma in any other tertiary institution.

No other person's work has been used without due acknowledgement in the main text of the Report.

I fully understand that any discrepancy from the above statements will constitute a case of plagiarism and be subject to the associated.

_____

Signature