International Journal of Innovative Computing, Information and Control
Volume **13**, Number **1**, February **2017**

ICIC International ©2017 ISSN 1349-4198
pp. 95–109

ROAD SIGN CLASSIFICATION SYSTEM USING CASCADE CONVOLUTIONAL NEURAL NETWORK

Reza Fuad Rachmadi[1,2], Yoshinori Komokata[1], Keiichi Uchimura[1] and Gou Koutaki[1]

[1]Graduate School of Science and Technology
Kumamoto University
Kumamoto-shi, Chuo-ku, Kurokami 2-39-1, Kumamoto 860-8555, Japan
{fuad; komokata}@navi.cs.kumamoto-u.ac.jp; {uchimura; koutaki}@cs.kumamoto-u.ac.jp

[2]Department of Multimedia and Network Engineering
Institut Teknologi Sepuluh Nopember
Keputih, Sukolilo, Surabaya 60111, Indonesia

Received June 2016; revised November 2016

Abstract. *We proposed a road sign classification system using C-CNN (cascade convolutional neural network) classifier. The cascade configuration is designed so that the classifier can easily converge with the data. Our system consists of six stages of Network in Network (NiN) architecture based CNN classifier. The data augmentation method is used to enrich the training and testing dataset which also tests the robustness of our system. Our Japan road sign dataset consists of ten classes with 7,500 examples for each class. Each image cropped from real street images is taken by the camera attached to the top of the car. From the experiments, our system is more efficient compared with bag-of-features method. The execution time of our system is less than 20 ms using appropriate hardware configuration, which is suitable for real-time application approaches like an autonomous car or driver assistance system.*
Keywords: Road sign classification, C-CNN, Data augmentation, NiN architecture

1. **Introduction.** The road sign recognition system is very useful for a lot of ITS (intelligent transport system) applications, including autonomous car, driver assistance system, and road traffic mapping. Autonomous car and driver assistance system required real-time road sign recognition method to support the system which is very crucial to the system performance. Each country or area may have different road sign symbols and the solution for road sign recognition system for each country may differ. Shape, color, and text are three essential things to recognize the road sign symbol. The road sign recognition system can be divided into two different subsystems, road sign detection system and road sign classification system. The road sign detection system is responsible for detecting each road sign symbol in some real street image input, and the road sign classification system is responsible for categorizing each road sign symbol to some predefined category. The system described in [4, 9, 10, 11, 12, 15, 19, 21, 22] is covering all subsystems, road sign detection system and road sign classification system. Other researches, like described in [5, 14, 16, 20], only cover detection or classification of the road sign image.

Yin et al. [4] report a progress of the road sign recognition system using rotation invariant binary pattern (RIBP) features, originally used for texture recognition. Yin et al. use Hough transform method to extract road sign candidate window in the image followed by RIBP features extraction and classification using neural network classifier. They test the system using GTSRB (German traffic sign recognition dataset) dataset [17]

and STS (Sweden traffic sign) dataset [18]. As a result, Yin et al.'s system achieved an average accuracy of 98.62% on GTSRB dataset and 98.33% on STS dataset.

Other research in road sign recognition was described in [10]. Mathias et al. [10] use the combination of grayscale value (I), PHOG, and HOG as features. Mathias et al. [10] try several state-of-the-art classifiers for road sign classification, including nearest neighbor classifier (NN), sparse representation-based classifier (SRC), iterative nearest neighbor classifier (INNC), and SVM classifier. The best accuracy of the system is 98.53%, achieved using a combination of I, PHOG, and HOG features with iterative nearest neighbors based linear projection (INNLP) as dimensional reduction and INNC as a classifier. For road sign detection system, Mathias et al. [10] use cascade Viola-Jones in HSI color space and achieve a good AUC score for localization of the road sign in the image.

Zeng et al. [5] describe a road sign classification system using deep CNN features and ELM (extreme learning machine) classifier. Deep CNN features were extracted from the convolutional neural network with eight layers and 43 high features output. The deep features are treated as input to ELM classifier. Zeng et al. [5] report the average accuracy of the system is 99.4% on GTSRB dataset.

According to three related works previously discussed, gradient based features (RIBP, HOG, or PHOG) are very famous features used for road sign classification problem and produce high accuracy system. In other hands, the combination of shape information and the color features of the road sign may increase the accuracy of the system. CNN classifier offers a complete system that forms a feature extraction system that learned from data and a neural network based classifier. The main problem of CNN classifier is required of huge and balance dataset for the training process. Balance dataset means that the number of examples between the classes is balanced or at least the difference is not too much. Cascading the classifier by grouping the dataset using some configuration may reduce the unbalance problem and may help the classifier to find the appropriate visual features for the problem.

In this paper, we proposed a road sign classification system using C-CNN (cascade CNN) previously described in [1]. We extend the experiments from previous paper and balance the number of examples in our Japan road sign dataset using data augmentation method. The cascade configuration will help the classifier easily find the solution for the problem because by grouping the data that share the visual cues, the complexity of the problem will reduce gradually. We use CNN based classifier because it is a complete solution, and the feature extraction stage is learning from data. Our C-CNN classifier consists of six stages of CNN classifier based on NiN architecture [6]. For the experiments, we use Japan road sign image dataset. Our contributions can be described as follows.

- We present the Japan road sign dataset that consists of ten classes of road signs. The dataset is created by cropping the image from Japan real street images taken using a panoramic camera attached to the top of the car.
- We proposed C-CNN classifier as a solution for our road sign classification system. Each stage in our C-CNN designed manually by choosing an appropriate split within the data, and it consists of one CNN classifier based on NiN architecture. The split of each stage forms the cascade configuration of the system with a specific dataset. Our method can be applied to another dataset as well, but with different cascade configuration. The cascade configuration was formed manually by selectively grouping the data based on visual cues as described in Section 3.
- We investigate the effect of input resolution of the image and data augmentation with the average accuracy of the system. We use affine transformation for data augmentation method to enrich the training and testing data.

The rest of the paper is organized as follows. Section 2 describes the detail of the dataset and data augmentation method used in our experiments. The detail of our proposed system is described in Section 3. Section 4 describes the results of the training and testing process along with discussion and the detail analysis of the model. At the end of the paper, concluding remarks and future works describe briefly as a summary of our experiments.

2. **The Dataset.** We introduce our Japan road signs dataset that consists of ten classes of road sign image. Each image cropped from Japan real street images is taken by a camera attached at the top of the car. The images saved in the spherical panoramic coordinate, and an example of the Japan real street image can be viewed in Figure 1. We selectively cropped and rearranged the road sign image into ten classes based on those real street images. Table 1 described the class number, class name, and road signs symbol

FIGURE 1. An example of full sphere panoramic image taken from the real street environment in Japan. The images are used to form our dataset.

TABLE 1. Ten classes of the road sign in our dataset

Class Name	Examples Sign Symbol
0 - No road sign	–
1 - Restricted Paths & Roadways	
2 - Follow the directions	
3 - Speed limit signs	
4 - Closed signs	
5 - Prohibition signs	
6 - No Parking signs	
7 - Size & weight limit signs	
8 - No entry sign	
9 - Stop sign	

for each category. After cropping each road sign symbol in the sphere panoramic images, we enrich the dataset by applying a geometrical transformation to the dataset or it called data augmentation.

2.1. **Data augmentation.** Data augmentation is a paradigm that describes the duplication of a dataset using a transformation pattern, so the result of the duplication process is not identically the same with the original one. The original idea is to train a robust classifier by extending the transform variation or subset region (cropping region) of the training dataset. Researches described in [6, 7, 13] use data augmentation method to enrich the dataset in the training process. In our case, we use a combination of 3D rotation to enrich the dataset. Let (x, y, z) be a pixel coordination in some 3D space; we apply following transformation to each pixel

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \mathbf{R_z}(\theta_\mathbf{z})\mathbf{R_y}(\theta_\mathbf{y})\mathbf{R_x}(\theta_\mathbf{x}) \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{1}$$

with

$$\mathbf{R_z}(\theta_\mathbf{z}) = \begin{bmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

$$\mathbf{R_y}(\theta_\mathbf{y}) = \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{bmatrix} \tag{3}$$

$$\mathbf{R_x}(\theta_\mathbf{x}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{bmatrix} \tag{4}$$

The detail illustration of the data augmentation process can be viewed in Figure 2, with d being the focal point of the virtual camera and it defined variable. Final coordinate of the pixel is calculated using $x_0 = \frac{x'}{1+z'/d}$ and $y_0 = \frac{y'}{1+z'/d}$. We use θ_x of $-0.5°$, $-0.25°$, $0°$; θ_y of $-0.5°$, $-0.25°$, $0°$, $0.25°$, $0.5°$; θ_z of $-0.25°$, $0°$, $0.25°$; and focal point d of 250, 750 for data augmentation process. After data augmentation process, our dataset consists of 7,500 examples for each class with 5,000 examples used as training data and 2,500 examples used as testing data. Output examples of the data augmentation process can be viewed in Figure 3.

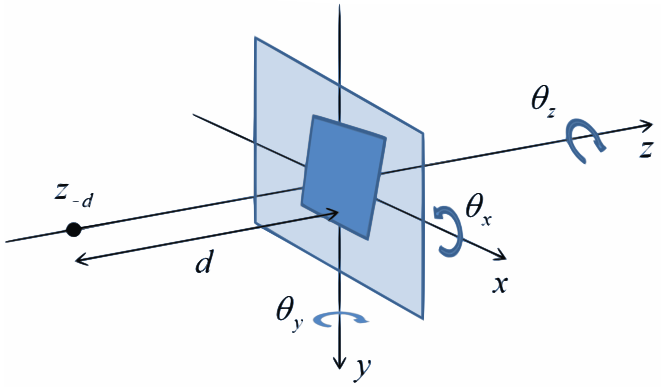

FIGURE 2. The illustration of a geometrical transformation of a road sign image in 3D space

FIGURE 3. An example of data augmentation process: (a) the original image, (b)-(e) output from data augmentation process

2.2. **Dataset summary.** For a summary, our dataset consists of ten classes and 75,000 examples with 7,500 examples for each class. We split the dataset for training and testing process. For training process, we provide 5,000 examples for each class with a total number of examples being 50,000 for ten classes. For testing process, we provide 2,500 examples of each class with a total number of examples being 25,000 for ten classes. The minimum and maximum resolutions of our road sign image dataset are 30×30 and 414×414 respectively. The average resolution of our road sign dataset is 58×58 pixel, which covers from small to large image scale of a road sign. To compare the difference between dataset with and without data augmentation, we also tested our system using unbalance dataset in which the total number of examples for each class varied from 63 to 3,713. Some road sign image in our dataset suffers from blur and distortion noise. The blurring effect appears because the image is out of focus and distortion appears because the image saved in the spherical panoramic coordinate.

3. **The C-CNN.** C-CNN means several CNN classifiers that configured using some cascade configuration. By using the proposed cascade configuration, the classifier can converge easier than classifier without cascade configuration. In this section, we briefly explain CNN classifier and our cascade configuration scheme.

3.1. **CNN classifier [2].** Convolutional neural network (CNN) is a type of neural network classifier that makes use of convolution process as features extractor and multi-layer perceptron as a classifier. The convolution process in CNN has the same analogy as convolution or filtering process in image processing algorithm. Let I_i be an input image and k be some convolution kernel; the convolution process is computed as follows

$$I_o[m,n] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} I_i[i,j]h[m-i,n-j] \tag{5}$$

with $[m,n]$ being a pixel value at coordinate (m,n). The training process will choose the suitable value of h depending on the problem. There are three other processes usually appearing in CNN architecture, activation process, normalization process, and pooling process. There are a lot of activation equations that can be used for activation process and the most used function is the ReLU (rectified linear unit) activation function. The normalization process is computed after activation layer. The LRN (local response normalization) method is usually used for the normalization process followed by a pooling process. LRN and ReLU are computed by Equations (6) and (7) respectively.

$$V_{x,y}^i = C_{x,y}^i \Big/ \left(1 + \frac{\alpha}{n} \sum_{j=i-n/2}^{i+n/2} \left(C_{x,y}^i\right)^2\right)^{\beta} \tag{6}$$

$$A_{x,y}^i = \max\left(0, A_{x,y}^i\right) \tag{7}$$

$V^i_{x,y}$ is the output of LRN process and $C^i_{x,y}$ is un-normalize input. Parameters α, n, and β are defined constant parameters for the normalization process. The ReLU activation process computes using max function and it vanishes the value of the neuron that has a negative value. In the last layer of CNN classifier, the softmax function is usually used as output normalization of the classifier. The softmax function makes sure that the sum of the classifier output is one. Following equation is used to compute the softmax output.

$$S^{(i)} = e^{f\left(x^{(i)}\right)} \Big/ \sum_{l=1}^{k} e^{f\left(x^{(l)}\right)} \tag{8}$$

$S^{(i)}$ is the output of softmax function, $f\left(x^{(i)}\right)$ is i-th output probability of the network, and k is total number of network output.

In recent years, CNN classifier is used very often in a lot of problems, including image classification, object detection, and scene understanding. The evolution of CNN classifier appears when Krizhevsky et al. [13] win the ILSVRC (ImageNet Large Scale Visual Recognition Challenge) 2012. The network consists of 9 layers with 5 layers of convolutional layers and 4 fully connected layers. The convolutional layers consist of convolution process with activation, normalization, and pooling process; or only activation and normalization process. A regularization method called dropout is used in 4 fully connected layers to reduce the classifier overfitting.

Other CNN architecture was proposed by Lin et al. [6] called Network in Network (NiN) which combine the responses of the convolution process using multi-layer perceptron in between convolutional layers. Instead of using fully connected layers, NiN architecture uses global average pooling for final classification result in the last layer. NiN architecture is more compact and has higher accuracy than Krizhevsky network [13] on ImageNet dataset.

3.2. **Our cascade configuration.** Why did we propose cascade configuration and not just a single CNN? At the first time, we try to solve the road sign classification problem using only one NiN classifier with ten outputs, but the result of the system is not so promising. To help NiN classifier converge to the solution, we design the cascade configuration as viewed in Figure 5. Our cascade configuration consists of six stages and for each stage, the split is decided manually by analyzing the road sign image in the dataset. The complete NiN CNN architecture used for our road sign classification system can be viewed in Figure 4. NiN uses ReLU activation function for all neurons and one dropout regularization method in between the pool3 and conv4 layer. We trained the classifier each stage independently. Caffe framework [8] is used to train the classifier with SGD (stochastic gradient descent) method as training algorithm.

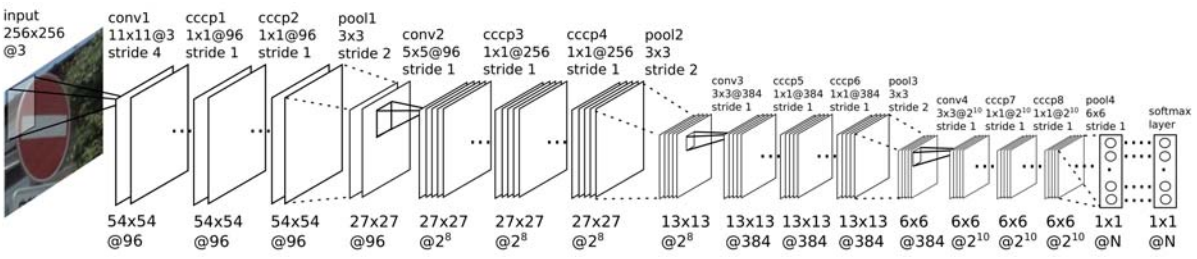

FIGURE 4. "Network in Network" CNN architecture with N output classes (the diagram was adopted from NiN paper [6]). ConvX is X-th convolutional layer, cccpX is X-th of multi-layer perceptron layer, and poolX is X-th pooling layer.

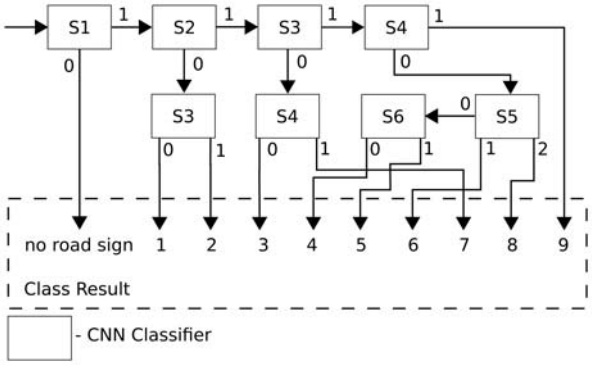

FIGURE 5. Cascade configuration used for our road sign classification system. SX is the X-th stage classifier of the cascade configuration.

As viewed in Figure 5, S1 to S6 are the stages of our cascade configuration. The color and shape are the visual cues used to decide the cascade configuration of the classifier. Each stage in the cascade configuration classifies a different group of the dataset and it is described as follows.

- **Stage 1 (S1)**. In this stage, the data is divided based on color and shape into two classes, road sign image and background image. We believe that road sign image will contain more stable shape and color than the background image, which contains unstructured shape and color.
- **Stage 2 (S2)**. Color similarity was used to decide the split at this stage. We divided the road sign image data into two classes, road sign with blue color and road sign with red color.
- **Stage 3 (S3)**. Stage 3 has two sub-stages, sub-stage for road sign image with blue color dominant and sub-stage for road sign image with red color dominant. In the sub-stage for road sign image with blue color dominant, the original class of the data is used and the output of the sub-stage is the final classification decision. In the sub-stage for road sign image with red color dominant, the data is divided based on shape, the road sign image that contains number symbol and not. Each sub-stage contains a single NiN classifier.
- **Stage 4 (S4)**. Stage 4 will process the red color road sign image passed from the previous stage. Same as stage 3, stage 4 has two sub-stages, the road sign image with number symbol and without number symbol. For road sign image with number symbol, the data is divided directly by the original class, class 3 and 7, and performs the final classification process. For road sign image without number symbol, the data is divided based on shape, circle and triangle shape of the road sign. This stage will be the final classification decision for class 9 because the triangle shape only appears in class 9 data.
- **Stage 5 (S5)**. In this stage, the road sign dataset from the previous stage is divided into three groups. At the first time we try to create stage 5 as final classification decision, but the classifier cannot converge well. We analyzed the result and decided to group the data in class 4 and 5 into one group. The rest of the data, class 6 and 8, is divided directly by the original class.
- **Stage 6 (S6)**. Stage 6 classifies the data directly into class 4 or class 5 and uses the original class of the data.

The number that appears on the side of the cascade stage in Figure 5 indicates the class number used in the intermediate result. The final class is converted from the intermediate class on the last stage of the cascade configuration.

Our cascade CNN classifier can be used to solve another problem but with different cascade configuration. As a summary, the steps of designing the cascade configuration can be described as follows.

Step i). Regroup the dataset that shares visual features by analyzing the dataset manually. The color or shape is used as visual features for regrouping the dataset.

Step ii). Train the classifier using the dataset that regroups in the previous step and validate the model using validation dataset.

Step iii). Forward to the next stage if the model achieves high accuracy in validation process or go back to step (i).

Step iv). Split the dataset based on the group in the step (i) and do the same procedure for the next stage.

4. **The Result.** We test our system using two datasets, unbalance dataset and balance dataset. The difference between those two datasets is about data augmentation method that only applied to balance dataset. For comparison, we deploy the Bag-of-Features (BoF) method with SIFT as features and Linear SVM as the classifier. The BoF method was a famous image classification method and used as the solution to several different problems like described in [14] which has accuracy over 90%. For more comprehensive analysis, we try the system using several different input resolutions of the road sign image. Different input resolutions may vanish the spatial features of the image, but with faster running time.

4.1. **Training process.** We use caffe framework [8] to conduct the training and testing process. Stochastic gradient descent (SGD) algorithm is used to train the network and initialize the weights of the network by weights trained using very large scale dataset, ImageNet dataset [3]. The fine-tuning will reduce the overfitting problem in the training process. We selectively choose the appropriate training parameters to avoid the exploding gradient problem.

For stage 1 until stage 5, we use learning rate $\alpha = 10^{-3}$ and decreased to 10^{-4} at 5,000 iterations. These values are enough to prevent the exploding gradient problem and appropriately update the weights of the network to the solution. Stage 6 was trained using lower learning rate $\alpha = 10^{-6}$ and decreased to 10^{-7} at 5,000 iterations. The learning rate is lower than other stages because the exploding gradient problem always occurs when the bigger learning rate is used. The side effect of lower learning rate is that the weights of the network will be updated more slowly. The training process was running for 10,000 iterations using the SGD algorithm with momentum $\gamma = 0.9$ and weight decay $\zeta = 0.0005$. The values of momentum and weight decay parameter were chosen based on several publications [6, 13] that use the same algorithm for the training process. The final update of the SGD algorithm in the training process computes as follows,

$$\Delta W_{i+1} = \gamma \Delta W_i - \zeta \alpha W_i - \alpha \nabla L(z, W_i) \tag{9}$$

$$W_{i+1} = W_i + \Delta W_{i+1} \tag{10}$$

where W_{i+1} is the final weights update of the network, W_i is the current weights, and $\nabla L(z, W_i)$ is the gradient of the loss function L with respect to the input z and weights W_i. There are some choices of loss function for classification problem including multinomial loss function and Euclidean loss function. In the last layer of the classifier, softmax layer with multinomial loss function is used to calculate the loss between output and target.

The cascade CNN classifier was trained using two different datasets, balance dataset and unbalance dataset. The maximum iterations of 10,000 in the training process cover 88 epochs for unbalance dataset and cover 22 epochs for balance dataset. Although the

epochs for the training of the balance dataset are lower than those of the unbalance dataset, the experiments show that the epochs are enough to train the classifier.

4.2. **Unbalance road sign dataset.** Table 2 shows the number of examples for each class of our unbalance road sign dataset. As we can see in Table 2, the number of examples for each class is varied from 63 to 3,713 examples. The lowest number of examples in unbalance road sign dataset is class 7 with only 63 examples for training and another 63 examples for testing. The highest number of examples in unbalance road sign dataset is class 0 with 3,713 examples for training and another 3,713 examples for testing. For further investigation, we use several different input resolutions of the data to the network. Table 3 shows the summary of our experiments. As we can see in Table 3, the best accuracy is achieved using 64×64 input resolution with 0.5% higher accuracy from original NiN input resolution, 256×256. The system with 64×64 input resolution has the highest accuracy in class 1, 2, 4, 5, 6, and 8 compared with other input resolution. Figure 6 is a confusion matrix of our system using 64×64 input resolution and unbalance road sign dataset. From Figure 6, the lowest accuracy appears in size & weight limit class (class 7), in which around 11% of our system misclassified the data to speed limit class (class 3). The phenomena occur because class 3 and class 7 have the same visual cues, which is number symbol. Another factor is a limited number of examples for class 7 which may not cover all visual cues required by the classifier.

For comparison with another method, we test a system based on BoF (Bag-of-Features) method with the unbalance road sign dataset. We implement the system described by Cao et al. [14] using SIFT features that extracted densely from the images. To form the dictionary of the features, the extracted features are clustered into some k cluster. In our experiments, we use $k = 1000$ for the features vocabulary and k-means for clustering

TABLE 2. The number of examples for each class in unbalance dataset

Class Name	**Number of Examples (training/testing)**
0 - No road sign	3,713 / 3,713
1 - Restricted Paths & Roadways	1,763 / 1,763
2 - Follow the directions	1,671 / 1,671
3 - Speed limit signs	751 / 751
4 - Closed signs	261 / 261
5 - Prohibition signs	123 / 123
6 - No Parking signs	1,158 / 1,158
7 - Size & weight limit signs	63 / 63
8 - No entry sign	1,056 / 1,056
9 - Stop sign	1,861 / 1,861

TABLE 3. The summary of our experiments using several resolutions of the input data and unbalance road sign dataset

Resolution	Class Number										Mean
	0	1	2	3	4	5	6	7	8	9	
256×256	**.986**	.994	**.998**	.991	.897	.919	.997	.873	.989	.995	.9637
128×128	.979	.993	**.998**	.988	.877	.870	**.998**	**.905**	.990	.992	.9589
64×64	.981	**.996**	**.998**	.993	**.931**	**.927**	**.998**	.873	**.991**	.994	**.9682**
32×32	.973	.990	**.998**	**.996**	.839	.870	.997	.794	.990	**.996**	.9441

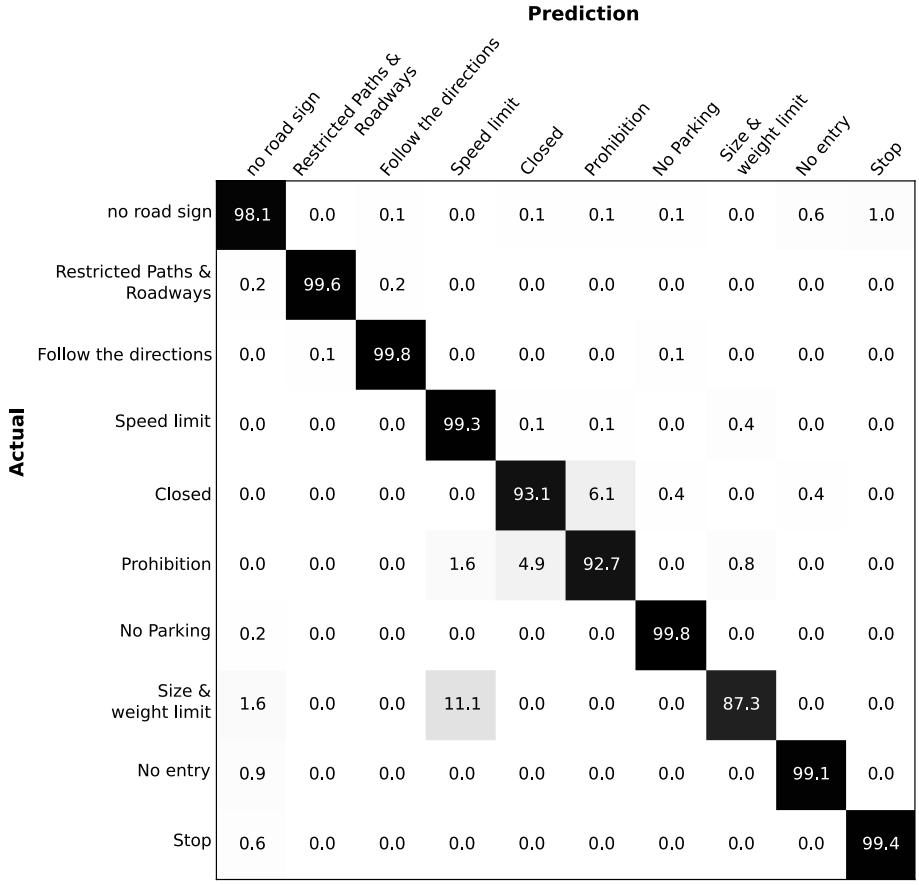

FIGURE 6. Confusion matrix for our system with 64×64 input resolution of unbalance road sign dataset

TABLE 4. Average accuracy of C-CNN, NiN, and BoF using unbalance road sign dataset

Method	Acc.
C-CNN using NiN architecture	**96.37**%
Single NiN	10.00%
BoF (1,000 codebook)	66.59%

algorithm. Table 4 is comparison of our system with two other systems, single NiN (without cascade configuration) and BoF with dense SIFT features. As we described in the previous section, a single NiN classifier cannot converge to the dataset and has the same accuracy with the random guess. By using our cascade configuration, the NiN classifier can achieve an average accuracy around 96% or ten times higher than single NiN. Those phenomena occur because by using some cascade configuration, the complexity of problem decreased and the classifier has more chance to converge to the solution. Our system outperforms the BoF method (with 1,000 codebooks) by 30%.

4.3. **Balance road sign dataset.** Our second experiment tested the system using balance dataset. The dataset created from the unbalance dataset with several new images and we extend the dataset using some geometric transformation as explained in Section 2. We use the same scenario as our experiments with unbalance road sign dataset. As we described in Section 2, the balance road sign dataset consists of 7,500 examples for each

class. For training process, we use 5,000 examples for each class and the rest of the data used for the testing process.

Table 5 shows the summary of our experiments using balance road sign dataset. There is average accuracy improvement of our system using balance road sign dataset. From the experiment, the highest average accuracy of the system is 97.94% achieved with 256×256 input resolution. The confusion matrix of our system using 256×256 input resolution can be viewed in Figure 7. From the confusion matrix, we can say that the problem with class 7, as discussed in the previous section, did not occur in the experiments with balance dataset because class 7 has enough examples to analyze by the classifier in the training process. The worst accuracy of our system using 256×256 input resolution appears in class 4 and 5. Stage 6 or S6 of our cascade configurations is the hardest problem for the

TABLE 5. The summary of our experiments using several resolutions of the input data and balance road sign dataset

Resolution	Class Number										Mean
	0	1	2	3	4	5	6	7	8	9	
256×256	.952	**.998**	**.995**	**.998**	**.906**	**.953**	**.999**	.998	.998	.997	**.9794**
128×128	.955	**.998**	.994	.968	.899	.901	**.999**	**.999**	.999	**.999**	.9711
64×64	**.954**	**.998**	.994	.977	.870	.922	**.999**	.992	.999	**.999**	.9704
32×32	.937	.994	.993	.944	.761	.915	.998	.908	**1.00**	**.999**	.9449

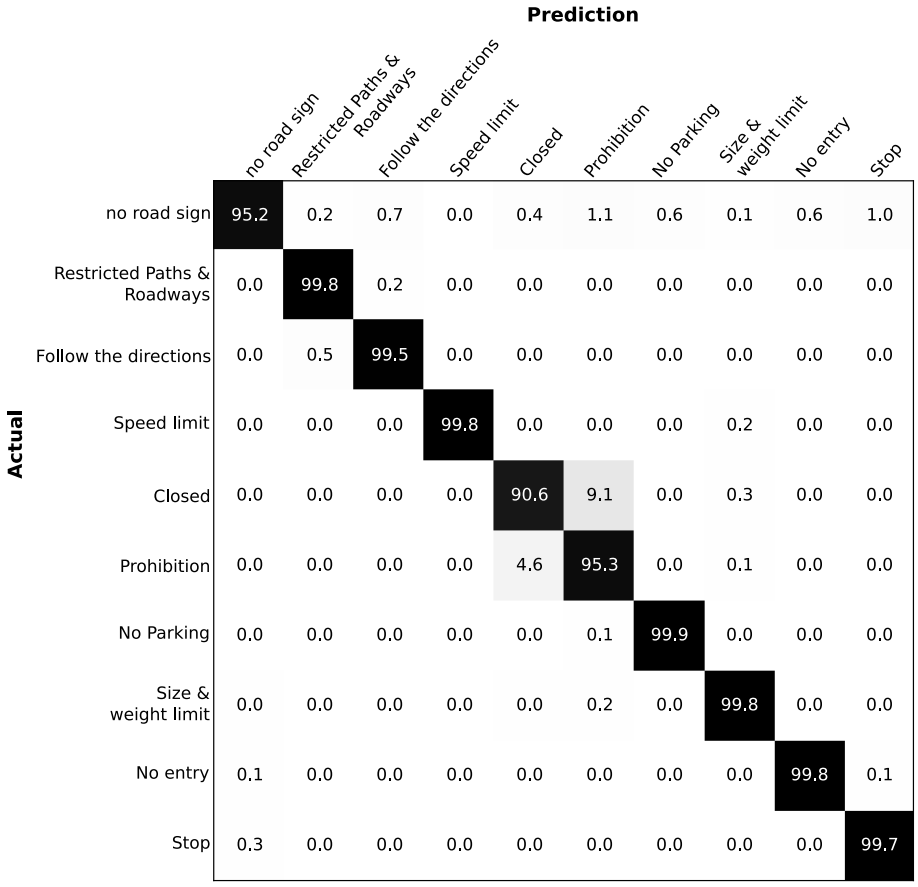

FIGURE 7. Confusion matrix for our system with 256×256 input resolution of balance road sign dataset

TABLE 6. The average accuracy of C-CNN, NiN, and BoF using balance road sign dataset

Method	Acc.
C-CNN using NiN architecture	**97.94**%
Single NiN	79.74%
BoF (1,000 codebook)	73.88%

NiN classifier and the training process of stage 6, need lower learning rate than another stage in our cascade configuration.

The comparison between our system with single NiN and BoF method can be viewed in Table 6. System using only single NiN classifier can converge very well compared with the experiment using unbalance dataset. The behavior shows that single NiN classifier required non-skew dataset to converge to the solution. Our solution, cascading the NiN classifier, decreases the complexity of the problem and the classifier converges easier with the problem. BoF method seems to have higher accuracy on this experiment compared with experiments using unbalance road sign dataset, but the average accuracy of the system is still lower than our system. Our system outperforms the BoF method by 20% and the single NiN classifier by 17%.

4.4. **Detail analysis of C-CNN classifier.** For detail analysis, we investigate the kernel weights of the network. Figure 8 shows the visualization of the kernel weights in the first convolutional layer of the NiN classifier on stages 1, 5, and 6. We perform a quick calculation of the difference between the kernel weights of NiN classifier after training process with the original kernel weights of NiN classifier used as initialized weights. The most weights change occurs in last several layers of the NiN classifier because the last several layers will perform the classification. Those phenomena occur because the features extraction network in the several first layers is enough to perform the classification process. The outputs of each layer of the NiN classifier are also analyzed. Figure 9 shows the visualization of the kernel weights and the outputs of the conv1, conv3, conv4, and cccp8 layers in the C-CNN. The outputs of convolutional layers in Figure 9 are visualized using heat map and the kernel weights are visualized using RGB color space. The cccp8 layer is the last layer of the NiN classifier before average pooling process at the end of the NiN classifier. The kernel weights of conv3 and conv4 layers are visualized by RGB color space but in the real calculation, the kernel is a multi-dimensional channel kernel rather

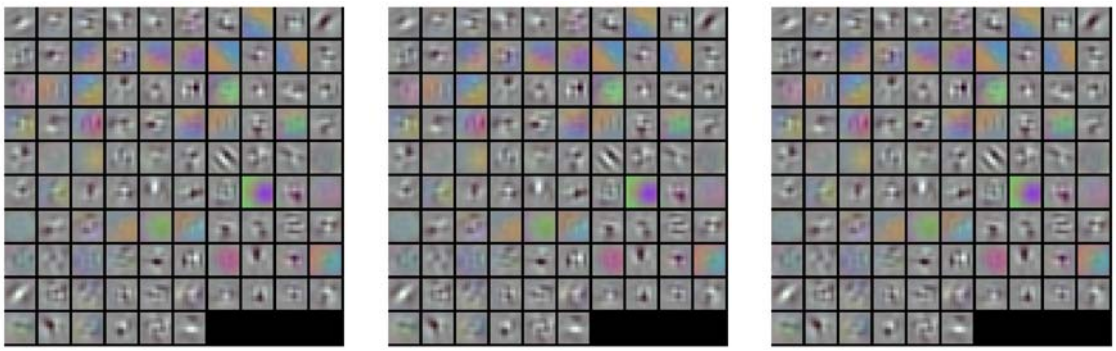

FIGURE 8. The visualization of the kernel in the first layer of the NiN classifier: (a) stage 1, (b) stage 5 and (c) stage 6

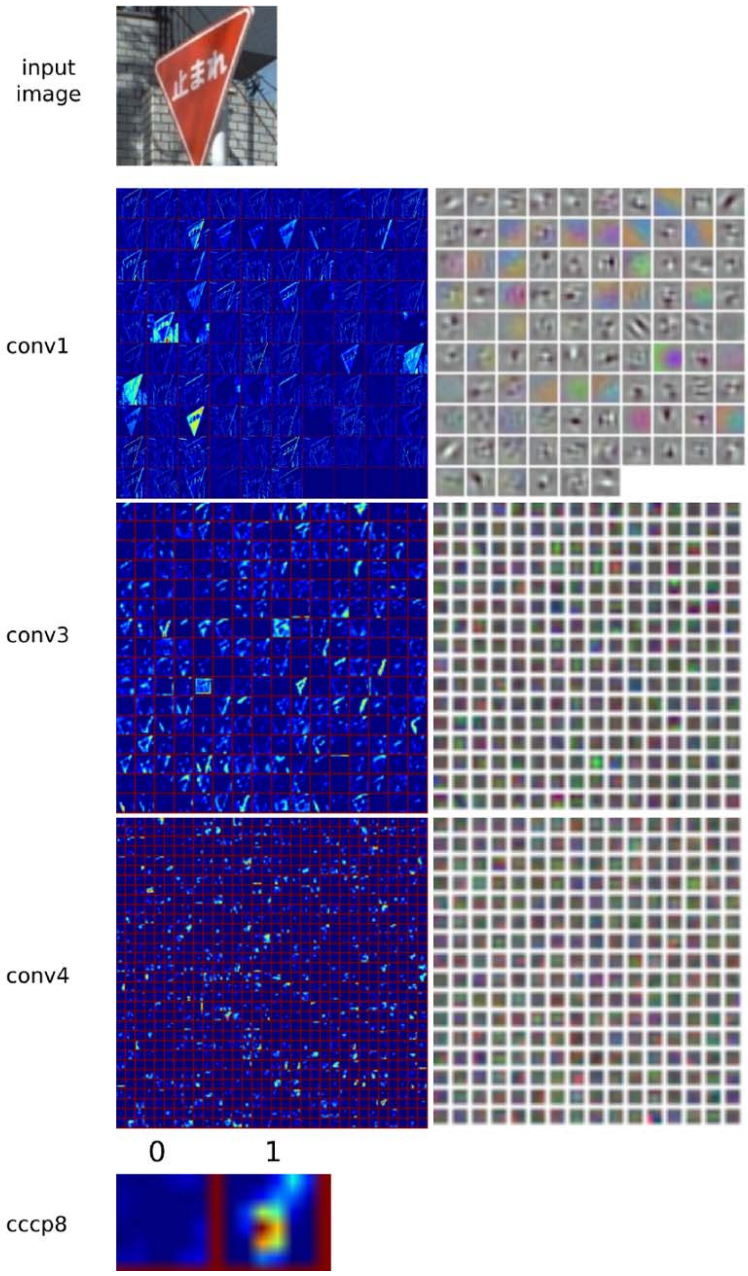

FIGURE 9. The visualization of each layer kernel and result for NiN CNN classifier in stage 1 for an example input image

than three channel kernel and only 768 kernels are visualized in Figure 9. As viewed in Figure 9, the conv1 layer extracts shape and color information of the input image and the multilayer perceptron between the convolutional layer calculated the contribution of each result of the convolution layer. The results of conv3 and conv4 in Figure 9 show that more neuron is activated using the combination of the previous layers, which is calculated by multilayer perceptron layer placed in between the convolutional layer.

4.5. **Execution time.** The summary of our system execution time can be viewed in Table 7. We measure the execution in two different resources, NVIDIA GTX 960 with 2 GB RAM and one core Intel i7@3.6 GHz. All time we measure is excluding the initialization time and copying data from CPU to GPU or GPU to CPU. As we can see in Table 7, if

TABLE 7. Average execution time of our C-CNN classifier with different input resolutions

Resolution	NVIDIA GPU GTX 960		CPU i7@3.6 GHz (1 core only)	
	Each Stage	Worst	Each Stage	Worst
256×256	3.49 ms	20.94 ms	97.73 ms	586.38 ms
128×128	2.64 ms	15.84 ms	36.69 ms	220.14 ms
64×64	2.63 ms	15.78 ms	31.46 ms	188.76 ms
32×32	2.61 ms	15.66 ms	23.05 ms	138.30 ms

smaller input resolution is used then the time execution of the system is reduced. There is around 10 ms to 20 ms overhead time for initialization time, which consists of a resizing process of the image from their original size and network initialization. If the system uses a GPU as computing resource, there is another overhead time for copying data from CPU to GPU and GPU to CPU. Road sign detection system can be adapted to create complete road sign recognition system. By combining our system with road sign detection system with road sign detection system described in [20], the total time execution for the complete system is around 80 ms. The execution time for the complete system is very suitable for the real-time ITS application, such as autonomous car or driver assistance system, or offline ITS application including road sign mapping and traffic flow mapping.

5. **Conclusion.** We present our C-CNN classifier which is a cascade classifier with NiN based CNN architecture for the road sign classification system. Our cascade configuration is designed manually by analyzing the road sign dataset and split the stage based on visual cues that appear on the dataset. By using cascade configuration, the NiN based CNN classifier can converge and find the solution for the problem easier than only using single NiN classifier. To prove the robustness of the classifier, we test our road sign classification system using two datasets, unbalance dataset and balance dataset. The unbalance dataset has a different number of examples for each class and it varied from 63 to 3,713 examples. To create more robust and balanced dataset, we extend the dataset using geometric transformation. The number of examples for each class is 7,500 with 5,000 examples for training process and 2,500 examples for the testing process. From the experiments, our system achieved high accuracy, more than 90%, and outperforms the BoF method with 1,000 codebooks. The average accuracy of the C-CNN classifier is not too affected by the input resolution of the data, but the bigger resolution can expose more spatial features to the classifier. The execution time of our system is quite fast even by using only a single core of the Intel i7 processor. Our proposed system has tons of applications including autonomous car, driver assistance system, road sign mapping, and traffic flow mapping.

For further system development, the combination of the road sign classification system with the road sign detection system can be implemented to create complete road sign recognition system. The main CNN classifier may be upgraded with deeper CNN architecture, such as GoogleNet or VGG-16 Net, to shallow the cascade configuration of the system.

REFERENCES

[1] R. F. Rachmadi, K. Uchimura, G. Koutaki and Y. Komokata, Japan road sign classification using cascade convolutional neural network, *ITS (Intelligent Transport System) World Congress*, pp.1-12, 2016.

[2] I. Goodfellow, Y. Bengio and A. Courville, Deep learning, *Book in Preparation for MIT Press*, http://www.deeplearningbook.org, 2016.

[3] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma and A. C. Berg, ImageNet large scale visual recognition challenge, *International Journal of Computer Vision*, vol.115, no.3, pp.211-252, 2015.

[4] S. Yin, P. Ouyang, L. Liu, Y. Guo and S. Wei, Fast traffic sign recognition with a rotation invariant binary pattern based feature, *Sensors*, vol.15, no.1, pp.2161-2180, 2015.

[5] Y. Zeng, X. Xu, Y. Fang and K. Zhao, Traffic sign recognition using extreme learning classifier with deep convolutional features, *International Conference on Intelligence Science and Big Data Engineering*, pp.272-280, 2015.

[6] M. Lin, Q. Chen and S. YanScherer, Network in network, *Proc. of International Conference on Learning Representation*, pp.1-10, 2014.

[7] K. Chatfield, K. Simonyan, A. Vedaldi and A. Zisserman, Return of the devil in the details: Delving deep into convolutional nets, *British Machine Vision Conference*, pp.1-12, 2014.

[8] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama and T. Darrell, Caffe: Convolutional architecture for fast feature embedding, *ACM International Conference on Multimedia*, pp.675-678, 2014.

[9] R. Timofte, K. Zimmermann and L. V. Gool, Multi-view traffic sign detection, recognition, and 3D localisation, *Journal of Machine Vision and Applications*, vol.25, no.3, pp.633-647, 2011.

[10] M. Mathias, R. Timofte, R. Benenson and L. V. Gool, Traffic sign recognition – How far are we from the solution? *International Joint Conference on Neural Networks*, pp.1-8, 2013.

[11] H.-H. Chiang, Y.-L. Chen and T.-T. Lee, Multi-stage with neuro-fuzzy approach for efficient on-road speed sign detection and recognition, *International Journal of Innovative Computing, Information and Control*, vol.9, no.7, pp.2919-2939, 2013.

[12] J. Stallkampa, M. Schlipsinga, J. Salmena and C. Igelb, Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition, *Neural Networks*, vol.32, pp.323-332, 2012.

[13] A. Krizhevsky, I. Sutskever and G. E. Hinton, ImageNet classification with deep convolutional neural networks, *Neural Information Processing Systems*, pp.1-9, 2012.

[14] J. Cao, L. Lou and Y. Zhang, Traffic sign recognition using visual feature toward driver assistance system, *International Conference Information and Business Intelligence*, pp.15-20, 2012.

[15] R. Timofte, V. A. Prisacariu, L. V. Gool and I. Reid, Combining traffic sign detection with 3D tracking towards better driver assistance, *Emerging Topics in Computer Vision and Its Applications, Series 1*, pp.425-446, 2011.

[16] P. Sermanet and Y. LeCun, Traffic sign recognition with multi-scale convolutional networks, *International Joint Conference on Neural Networks*, pp.2809-2813, 2011.

[17] J. Stallkamp, M. Schlipsing, J. Salmen and C. Igel, The German traffic sign recognition benchmark: A multi-class classification competition, *International Joint Conference on Neural Networks*, pp.1453-1460, 2011.

[18] F. Larsson and M. Felsberg, Using Fourier descriptors and spatial models for traffic sign recognition, *Proc. of the 17th Scandinavian Conference on Image Analysis*, pp.238-249, 2011.

[19] V. A. Prisacariu, R. Timofte, K. Zimmermann, I. Reid and L. V. Gool, Integrating object detection with 3D tracking towards a better driver assistance system, *IAPR International Conference on Pattern Recognition*, pp.3344-3347, 2010.

[20] T. Le, S. Tran, S. Mita and T. Nguyen, Realtime traffic sign detection using color and shape-based features, *The 2nd Asian Conference on Intelligent Information and Database Systems*, pp.268-278, 2010.

[21] S. Miyata, A. Yanou, H. Nakamura and S. Takehara, Road sign feature extraction and recognition using dynamic image processing, *International Journal of Innovative Computing, Information and Control*, vol.5, no.11(B), pp.4105-4113, 2009.

[22] W. J. Kuo and C. C. Lin, Two-stage road sign detection and recognition, *IEEE International Conference on Multimedia and Expo*, pp.1427-1430, 2007.