# Traffic Signs Recognition and Classification based on Deep Feature Learning

**Conference Paper** · January 2018

DOI: 10.5220/0006718806220629



**4 authors**, including:

Nanxin Wang
Tongji University
**3** PUBLICATIONS **6** CITATIONS
SEE PROFILE

Lan Lin
Tongji University
**30** PUBLICATIONS **45** CITATIONS
SEE PROFILE

# Traffic Signs Recognition and Classification based on Deep Feature Learning

Yan Lai, Nanxin Wang, Yusi Yang and Lan Lin
*School of Electronics and Information Engineering, Tongji University, Shanghai, China*

Keywords: Traffic Signs Recognition, Convolutional Neural Network, YCbCr Color Space, Support Vector Machine.

Abstract: Traffic signs recognition and classification play an important role in the unmanned automatic driving. Various methods were proposed in the past years to deal with this problem, yet the performance of these algorithms still needs to be improved to meet the requirements in real applications. In this paper, a novel traffic signs recognition and classification method is presented based on Convolutional Neural Network and Support Vector Machine (CNN-SVM). In this method, the YCbCr color space is introduced in CNN to divide the color channels for feature extraction. A SVM classifier is used for classification based on the extracted features. The experiments are conducted on a real world data set with images and videos captured from ordinary car driving. The experimental results show that compared with the state-of-the-art methods, our method achieves the best performance on traffic signs recognition and classification, with a highest 98.6% accuracy rate.

## 1 INTRODUCTION

Nowadays, unmanned automatic driving technology (Sezer et al., 2011) has attracted increasing attentions from researches and industry communities. The traffic signs recognition and classification play an important role in this field. A lots of research efforts have been devoted to dealing with the problem. However, many factors, such as insufficient illumination, partial occlusion and serious deformation, make traffic detection a challenging problem.

Feature extraction in traditional traffic signs recognition is based on hand-crafted methods, such as Scale-Invariant Feature Transform (SIFT) (Nassu and Ukai, 2010), Histogram of Oriented Gradient (HOG) (Creusen et al., 2010) and Speeded-Up Robust Features (SURF) (Duan and Viktor, 2015). With the rise of neural network theory, the application in recognition has also increased, e.g., the Semantic Segmentation-Aware CNN Model (Gidaris and Komodakis, 2015), showing a better feature-learning capabilities.

Even though traffic signs detection and classification had been developed for a long time, a complete data set was inadequate until the launch of the German Traffic Signs Recognition Benchmark (GTSRB) (Stallkamp et al., 2012) and Detection Benchmark (GTSDB) (Houben et al., 2014). Various methods have been making progress in these two tasks, such as the DP-KELM method (Zeng et al., 2017). However, the GTSDB cannot adapt very well in real world tasks, due to the small size of the training data. After that, a benchmark named Tsinghua-Tencent 100K (Zhu et al., 2016) has been proposed along with the end-to-end CNNs method, which shows a good performance of detection and classification of tiny traffic signs. However, the processing speed is still slow.

(a) Warning (b) Prohibition (c) Mandatory

Figure 1: Three Main Categories of Traffic Signs in China. Warning signs (mostly yellow triangles with a black boundary), Prohibition signs (mostly white surrounded by a red circle) and Mandatory signs (mostly blue circles with white information).

In this paper, three main traffic signs categories, i.e., warning signs, prohibition signs and mandatory signs (shown in Figure 1), are covered for experiments. Specifically, in our video-based CNN-SVM recognition framework, by introducing the YCbCr color space (Basilio et al., 2011), we firstly divide the color channels, secondly employ CNN deep network for deep feature extraction and then adopt SVM for classification. The experiments are conducted on a real world data set, based on which, a synthetically comparison illustrates the superiority of our model.



Lai, Y., Wang, N., Yang, Y. and Lin, L.
Traffic Signs Recognition and Classification based on Deep Feature Learning.
DOI: 10.5220/0006718806220629
In *Proceedings of the 7th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2018)*, pages 622-629
ISBN: 978-989-758-276-9

The rest of the paper is organized as follows: Section 2 discusses the related work. The framework and experimental data are shown in Section 3, while the introduction of the YCbCr-based network is presented in Section 4. The parameters adjustment and image preprocessing in CNN-SVM are done in Section 5. Finally, we give experimental results in Section 6 and conclusions in Section 7.

## 2 RELATE WORK

To solve the target recognition problem, most of the previous works use hand-crafted feature extracting methods mentioned in Section 1. Those conventional feature extraction methods, over-reliant on the designer's experience, meet some restriction in feature expression. On the contrary, the deep network model based on CNN is more powerful in feature expression. CNN method, which possessing the characteristics of rotation, translation and scaling invariance, is able to realize weight sharing through local receptive fields. It's widely used in the sub-area of target recognition, e.g., image classification (Krizhevsky et al., 2012) (Schmidhuber, 2012), face recognition (Sun et al., 2014) (Li et al., 2015) and pedestrian detection (Ouyang and Wang, 2014) (Zeng et al., 2013). Afterwards, various CNN-based models have been proposed, such as AlexNet (Krizhevsky et al., 2012), VGG (Simonyan and Zisserman, 2014) , GoogleNet (Szegedy et al., 2015), ResNet (He et al., 2016) and so on. The VGG network model has been widely used and improved by researchers thanks to its multi-layer construction and excellent performance on the most representative data set in target recognition field which named ImageNet (Krizhevsky et al., 2012). Based on this, some famous network models, e.g., the R-CNN series (Girshick, 2015) (Ren et al., 2017), YOLO (Redmon et al., 2016), SSD (Liu et al., 2016), R-FCN (Dai et al., 2016), appears in succession.

## 3 FRAMEWORK AND EXPERIMENTAL DATA

### 3.1 Framework of CNN-SVM Method

Figure 2 shows a process diagram of the working procedures of CNN-SVM. The CNN-SVM method can be concluded as the following six steps:

- Training images and testing street-view images are collected, and then the training images are labeled and transformed in YCbCr color space.
- Visual features of these training images are extracted in CNN.
- A SVM classifier is used to classify the training images based on the extracted features.
- Testing street-view images are sent to preprocessing progress, including homomorphic filter, morphological treatment and area threshold segmentation.
- Region of Interest (ROIs) in street view are acquired and delivered to CNN-SVM model.
- Detection and classification results are obtained.

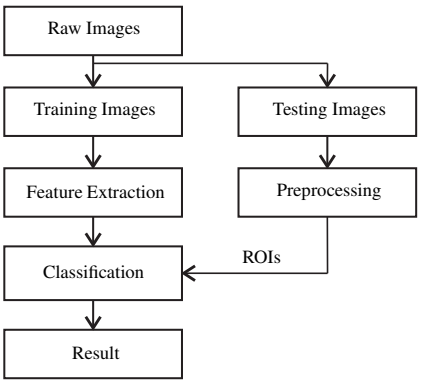


Figure 2: Framework of Traffic Signs Recognition.

### 3.2 Experimental Data

Eight kinds of signs, individually from the warning signs, prohibition signs and mandatory signs, are taken into consideration in our research. The training images mainly consists of mobile phone shootings, GTSDB and Baidu exploration. These images, through some transformation like rotating and affine transformations, reach a total number of 1000. Each image is unified with the size of $48\times48$. Then, these images are labeled and composed as a training data set which is shown in Figure 3. The testing street-view images comes from 4 videos, which are captured by PIKE F-100 FireWire camera. Each frame size of the videos is $1920\times1080$.

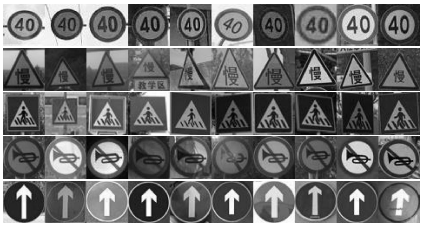

Figure 3: Traffic Signs Training Data Set.

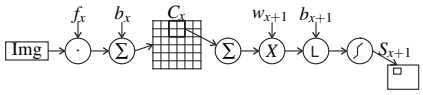


Figure 4: The Diagram of Convolution Progress. An input sample with the size of 48×48 is convoluted with a 5×5 convolutional kernel. Then, six 44×44 feature maps are generated. Each neuron in the feature map is connected with 5×5 convex kernel. After the summation of each group of pixels in the feature map, the weight and offset are added. After taking activate function, the neurons in the next layer are automatically acquired. Naturally, the next volume can be obtained by continuous translation and traverse operation.

# 4 CONVOLUTIONAL NEURAL NETWORK TRAINING BASED ON YCBCR COLOR SPACE

## 4.1 Convolutional Neural Network

Our basic model follows the classic LeNet-5 network, which consists of three convolutional layers ($C_1$, $C_3$ and $C_5$) and two sub-sampling layers ($S_2$ and $S_4$). The input of each layer represents a small group of local units from the upper layer. Each convolution layer contains multiple convolution kernels. These convolution kernels are able to scan the image features via different expressions, based on this, we can acquire various feature maps in different locations. The sub-sampling layer, following the convolution layer, is mainly used to reduce the resolution of the feature map, to extract the existing image features and to determine the features' relative location.

## 4.2 CNN Training

The training of the layer-concatenated CNN includes 4 main parts, i.e., forward propagation, error calculation, back propagation and weights adjustment. The forward propagation represents a progress of information delivery from the input layer to the output layer. Since initialization of the weighting parameters is random, the results obtained by forward propagation tend to be deviated. To modify this deviation, error estimation and parameters adjustment are included between each forward and backward propagation. More specifically, it runs with the following procedures:

- Extract a sample from the training data set and send it into the training network.
- Each layer's output, generated by the activation function, are continuously led into the next hidden layer till the output layer.
- Calculate the deviation matrix of the output.
- Conduct a layer-by-layer reverse calculation according to gradient descent algorithm.
- Acquire the updated weight and gradient values.
- Repeat the forward propagation to start the next iteration.

## 4.3 YCbCr Color Space for CNN's Feature Extraction

The previous traffic signs recognition and classification methods usually take CNN training in RGB space. In that space, the color information and the luminance information are mixed among channels, generating the variance of the extracted features. Nonetheless, the color distribution in RGB space is not uniform. Some subtle changes in color are captured difficultly.

YCbCr color space is mainly used for continuous image processing in the video. We can calculate each component by the transform formula shown as:

$$Y = 0.299R + 0.587G + 0.114B \quad (1)$$

$$Cb = 0.564(B - Y) \quad (2)$$

$$Cr = 0.713(R - Y), \quad (3)$$

where Y is luminance component, Cb represents the blue chrominance component and Cr denotes the red chrominance component.

In order to choose a better color space for feature extraction, we conduct an error evaluation among three typical color space, i.e., Grayscale, RGB and YCbCr. To do this, we send the training data set for CNN training. Taking the $b_n$ as the number of wrong samples in each batch training, the $n$ as the number of batch training time, the $S$ as the batch size. We are able to calculate the training error in each batch training which represented by $E_n$:

$$E_n = \frac{b_n}{S} \quad (4)$$

The results of different training errors shown as Table 1, we can find that the corresponding result of YCbCr space is better than the others.

Table 1: Training Error in Different Color Spaces.

| Color Space | Training Error |
|---|---|
| Grayscale | 0.138 |
| RGB | 0.105 |
| YCbCr | 0.073 |

# 5 PARAMETERS ADJUSTMENT BASED ON CNN-SVM

In this section, we will take training data set and testing street-view images to conduct parameters adjustment and image preprocessing in training and testing progress respectively. The goal of parameters adjustment is to increase training accuracy and to shorten the training time in the training parts. While in the testing progress, the image preprocessing operation is mainly used to eliminate the negative impacts, e.g., insufficient illumination and similar background, so as to acquire ROIs precisely.

## 5.1 Parameters Adjustment for Training

In the training part, some parameters, e.g., kernel size, iteration numbers and the connecting methods between convolutional layer and sub-sampling layer, greatly impact on training accuracy and speed. Thus some experiments are carried out based on the training data set for choosing the best parameters.

### 5.1.1 Kernel Size

In terms of kernel size, a comparison experiment is conducted both in the 2-hidden-layer network and the 4-hidden-layer network. In the 2-hidden-layer network, an input layer, a convolutional layer, a sub-sampling layer and an output layer are included. While in the 4-hidden-layer network, there exists one more convolution layer and sub-sampling layer. In addition, the kernel sizes of the convolution layer selected for the former are $9\times9$ and $5\times5$, yet $7\times7$ and $5\times5$ for the later. This comparison proves that a larger corresponding kernel size refers to less convolution layers, contributing to a higher training speed and less parameters. However, the over large size will cause the loss of local features.

Table 2: Training Time of Different Kernel Sizes.

| Hidden Layers | 2 | 2 | 4 | 4 |
|---|---|---|---|---|
| Kernel Size | $9\times9$ | $5\times5$ | $7\times7$ | $5\times5$ |
| Training Time | 377s | 260s | 481s | 325s |

### 5.1.2 Connecting Methods

The connection modes between the sub-sampling layer and the convolution layer include two methods, i.e., the fully connected method and the non-fully connected method. In order to connect more features, sub-sampling layer $S_2$ and convolution layer $C_3$ is fully connected. In this case, there are a total number of $16\times(6\times5\times5+1)=2,416$ training parameters according to Table 3. Thus, this connection method will lead to an increasing of computational cost. In the deeper layers, we can take the non-fully connected method like the connection between $S_4$ and $C_5$ to reduce the information redundancy. Taking this method, we can effectively reduce the training parameters which access to less but more useful features.

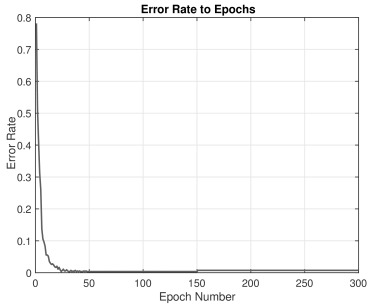


Figure 5: Training Error Rate with 300 Epochs.

Table 3: CNN Constructing Parameters.

| Layer | Type | Neurons | Kernel |
|---|---|---|---|
| 0 | Input | $48\times48$ | |
| 1 | Convolutional | $44\times44$ | $5\times5$ |
| 2 | Subsampling | $22\times22$ | $2\times2$ |
| 3 | Convolutional | $18\times18$ | $5\times5$ |
| 4 | Subsampling | $9\times9$ | $2\times2$ |
| 5 | Convolutional | $5\times5$ | $5\times5$ |
| 6 | Fully | $1\times600$ | |

### 5.1.3 Iteration Numbers

Theoretically, a higher iteration number, means a more thorough process. We are able to get more features from each iteration. However, the redundant iteration number will lead to an increase of training error, if noise or the non-representative features in the photos are fitted. Thus, a test with several iteration numbers is conducted. We firstly load the training data set for the test and choose the iteration numbers of 300. As shown in the Figure 5, we can find that the error rate of training process experiences a slight increase after it converging to a very low rate with 150 epochs, which may ascribe to the introduce of non-representative features.

Going deeply, we can load the training data set, randomly select the number of $T$ as training samples, the rest are testing samples for validation. The total batch training time can be represented by $N=\frac{T}{S}$. Based on the Formula 4, the recognition accuracies which is defined as $A$ can be calculated by the Formula:

$$A = (1 - \frac{\sum_{n=1}^{N}(E_n)}{N}) \times 100\% \quad (5)$$

We are able to test the recognition accuracies with the parameters $T$=840 and $S$=60, the results of CNN accuracies with different iteration numbers are shown in Table 4, in which the $I$ represent Iterations and $A$ means Accuracy. Clearly, the 150-epoch-iteration is verified to be the best.

Table 4: CNN Accuracies of Different Iterations.

| I (epochs) | 100 | 150 | 350 | 500 |
|---|---|---|---|---|
| A (%) | 97.27 | 98.18 | 94.4 | 92.73 |

Briefly, the experimental parameters can be concluded in Table 5.

Table 5: CNN Experimental Parameters.

| Kernel Size | Batch Size | I (epochs) |
|---|---|---|
| 5×5 / 2×2 | 60 | 150 |

In order to optimize the classifying performance, we have trained the SVM classifier by libsvm package (Suralkar et al., 2012) (Chang and Lin, 2011) after removing fully connected layer. The output vector of the last layer in CNN is regarded as the input of the SVM. We are able to expand the normal classifier to solve the multi-classify problem, with training and connecting $\frac{k\times(k-1)}{2}$ normal classifier as the construction of a binary tree, in which $k$ represents the number of classes. Kernel function is a key factor in constructing the SVM classifier. There are several kernel functions provided for us to select, e.g., RBF, Linear and Gaussian Kernel, with different parameters and outcomes. Taking 100% and 50% training data set as the experimental samples, we are able to conduct an experiment to choose the most suitable kernel function as Table 6. The overall training time and training accuracy are two factors to evaluate the performance of different functions.

From the point of training accuracy, the RBF kernel outperforms the Linear kernel. As the parameter number of RBF is more than Linear kernel, it will take times to find a better result. Based on large number of parameters, the training speed of RBF is obviously slower than the Linear kernel. Thus, the Linear kernel can be a more suitable choice for our model. In addition, the Table 6 also illustrate that the SVM method is robust when the training data set is in a small scale. Because our data set only contains 1000 images, the 50% of them still show good performance.

Table 6: The Performance of Different Kernel Sizes.

| Training Data | Kernel | Accuracy | Time |
|---|---|---|---|
| 100% | RBF | 98.81% | 507s |
| 100% | Linear | 98.6% | 371s |
| 50% | RBF | 96.41% | 285s |
| 50% | Linear | 96.1% | 122s |

(a) Original (b) Homomorphic (c) Gamma

Figure 6: Comparison between Homomorphic Filtering and Gamma Correction.

## 5.2 Preprocessing for Testing

Before testing progress, we have to take some image preprocessing operations in street-view images to eliminate the effect of insufficient illumination, partial occlusion and serious deformation. The main preprocessing operations include image enhancement and image segmentation.

Firstly, we need to deal with the images under a poor illumination. A typical method is the homomorphic filter method (Cai et al., 2011), which uses a suitable homomorphic filter function $H(u,v)$. In this method, the coefficients are $H_l < 1$ and $H_h > 1$, the function $H(u,v)$ would decrease. Therefore, we are able to enhance the images by reducing the low frequency and increasing the high frequency. Another way is gamma correction (Huang et al., 2013), which compensates the deficiency in dim light. The Figure 6 shows a comparison of the homomorphic filtering as Figure 6(b) and gamma correction as Figure 6(c). Clearly, the former performs better, as the color of the testing image has a certain distortion after gamma correction.

In addition, the images with normal light will not be sent to the filtering process directly. To judge the predicable images, we can introduce the YCbCr color space, as the Y component represents the illumination factor. We are able to select the thresholds by drawing the pixel distribution in histogram and counting the pixel numbers ($a$), in images with extremely light. After taking the experiments, the threshold can be determined by Y$\in$[200,234] and $a < 53901$.

Secondly, we need to take morphological treatment before segmentation to eliminate discontinuous tiny areas. However, some obstacles in the street view have not been removed totally, for instance, the non-signs areas which are similar with the signs areas. Thus, we define the size and proportion of the traffic signs by experimental results. In addition, the one

Figure 7: ROIs in Different Scenarios.

third part in the bottom image, where we set the pixel value to zero, would not contain any signs. Thus, the non-signs area can be removed.

Finally, the ROIs can be selected by a bounding box with fixed size and proportion in the whole street-view image as shown in Figure 7. Among these results, we can find that the most of traffic signs are successfully recognized while some are ignored or double recognized.

# 6 ANALYSIS OF EXPERIMENTAL RESULTS

In this paper, both training and testing were done on a Linux PC with an Intel i7-6900k CPU and NVIDIA GTX 1080 Ti GPU. The collected training data set is loaded and trained in our network. From the training data set, we randomly select 840 images as training samples, the rest as testing samples for cross validation.

## 6.1 CNN-SVM Recognition and Classification Results

The CNN is able to detect and classify the targets, via extracting the training samples' features. Then, SVM contribute to a better result in classification. With a number of parameter adjustments, the CNN is successfully trained with 98.18 % accuracy rate, while the CNN-SVM's accuracy reaches 98.6 %. Even though the growth of accuracy is slight, the training time is very near from each other, with 366s and 371s respectively.

In addition, we take the mean shift method (Comaniciu and Meer, 2002) to track the traffic signs in a short video as Figure 8. The figure shows that the capabilities of this framework in detecting the tiny traffic signs which take a small area as 0.2 %−0.4 % of the whole image. The results of the recognition and classification of 8-classes traffic signs are shown in Table 7.

Figure 8: Tracking in Video by Mean Shift. From the upper left to the lower right sub-images represent the 1st, 3rd, 5th and 8th frame respectively.

Table 7: The Results of the Recognition and Classification of 8-classes Traffic Signs.

| Classes | Total Numbers | CNNs-SVM(%) |
|---|---|---|
| Limit 30 | 109 | 97.34 |
| Limit 40 | 123 | 98.32 |
| Limit 50 | 141 | 98.7 |
| Slow | 136 | 98.9 |
| Crossroads | 134 | 98.56 |
| No tooting | 153 | 99.43 |
| Right | 138 | 98.46 |
| Straihgt | 66 | 96.6 |

## 6.2 Comparison with State-of-the-Art Methods

The Figure 9 illustrates the comparison results with other state-of-the-art methods. It shows that our CNN-SVM method is able to achieve a higher accuracy (98.6%) than the others, e.g., HOG-LDA (Stallkamp et al., 2012), HOG-Random Forest (Zaklouta et al., 2011) and PCA-SVM (Chan et al., 2015). Furthermore, the training time of our method (around 371s) is the fastest, which nearly twice as fast as the HOG-LDA.

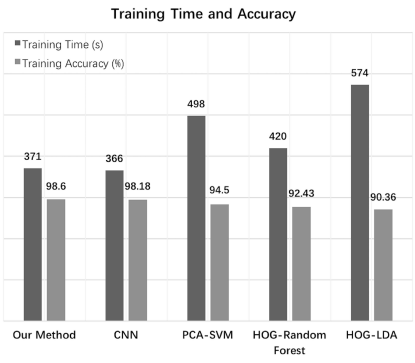


Figure 9: Comparison with State-of-the-Art Methods.

# 7 CONCLUSIONS

In this paper, a recognition and classification method based on CNN-SVM has been proposed. In the training process, the deep image features are extracted by CNN in the YCbCr color space. SVM is connected with the last layer of CNN for further classification, which contributes to a better training results. On the other hand, some images preprocessing procedures are conducted in the testing process, in order to eliminate those negative impacts, e.g., insufficient illumination, partial occlusion and serious deformation. Experiment-based comparison with other state-of-the-art methods verify that our model is superior than the others both in training accuracy and speed. Furthermore, we found that some traffic signs are miss-recognized when we apply this method in the unmanned ground vehicle. In near future, we plan to expand our data set by seeking out more images of traffic signs, especially the images at night. Then, we will accelerate the speed by optimizing the algorithm for real-time application in vehicles.

# ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China (NSFC), Grant No.61373106. The authors gratefully acknowledge everyone who helped in the work. *Corresponding author: Lan Lin.*

# REFERENCES

Basilio, J. A. M., Torres, G. A., Rez, G. S., nchez, Medina, L. K. T., Meana, H., and Ctor, M. P. (2011). Explicit image detection using ycbcr space color model as skin detection. In *American Conference on Applied Mathematics and the Wseas International Conference on Computer Engineering and Applications*, pages 123–128.

Cai, W., Liu, Y., Li, M., Cheng, L., and Zhang, C. (2011). A self-adaptive homomorphic filter method for removing thin cloud. In *International Conference on Geoinformatics*, pages 1–4.

Chan, T. H., Jia, K., Gao, S., Lu, J., Zeng, Z., and Ma, Y. (2015). Pcanet: A simple deep learning baseline for image classification? *IEEE Transactions on Image Processing A Publication of the IEEE Signal Processing Society*, 24(12):5017–5032.

Chang, C. C. and Lin, C. J. (2011). *LIBSVM: A library for support vector machines*. ACM.

Comaniciu, D. and Meer, P. (2002). Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 24(5):603–619.

Creusen, I. M., Wijnhoven, R. G. J., Herbschleb, E., and With, P. H. N. D. (2010). Color exploitation in hog-based traffic sign detection. In *IEEE International Conference on Image Processing*, pages 2669–2672.

Dai, J., Li, Y., He, K., and Sun, J. (2016). R-fcn: Object detection via region-based fully convolutional networks.

Duan, J. and Viktor, M. (2015). Real time road edges detection and road signs recognition. In *International Conference on Control, Automation and Information Sciences*, pages 107–112.

Gidaris, S. and Komodakis, N. (2015). Object detection via a multi-region and semantic segmentation-aware cnn model. In *IEEE International Conference on Computer Vision*, pages 1134–1142.

Girshick, R. (2015). Fast r-cnn. In *IEEE International Conference on Computer Vision*, pages 1440–1448.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. *Computer Vision and Pattern Recognition*, pages 770–778.

Houben, S., Stallkamp, J., Salmen, J., Schlipsing, M., and Igel, C. (2014). Detection of traffic signs in real-world images: The german traffic sign detection benchmark. In *International Joint Conference on Neural Networks*, pages 1–8.

Huang, S. C., Cheng, F. C., and Chiu, Y. S. (2013). Efficient contrast enhancement using adaptive gamma correction with weighting distribution. *IEEE Transactions on Image Processing A Publication of the IEEE Signal Processing Society*, 22(3):1032–41.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *International Conference on Neural Information Processing Systems*, pages 1097–1105.

Li, H., Lin, Z., Shen, X., Brandt, J., and Hua, G. (2015). A convolutional neural network cascade for face detection. In *Computer Vision and Pattern Recognition*, pages 5325–5334.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., and Berg, A. C. (2016). *SSD: Single Shot MultiBox Detector*. Springer International Publishing.

Nassu, B. T. and Ukai, M. (2010). Automatic recognition of railway signs using sift features. In *Intelligent Vehicles Symposium*, pages 348–354.

Ouyang, W. and Wang, X. (2014). Joint deep learning for pedestrian detection. In *IEEE International Conference on Computer Vision*, pages 2056–2063.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. *Computer Vision and Pattern Recognition*, pages 779–788.

Ren, S., Girshick, R., Girshick, R., and Sun, J. (2017). Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 39(6):1137–1149.

Schmidhuber, J. (2012). Multi-column deep neural networks for image classification. *Computer Vision and Pattern Recognition*, 157(10):3642–3649.

Sezer, V., anr Dikilita, Ercan, Z., Heceolu, H., ner, A., Apak, A., Gkaan, M., and Muan, A. (2011). Conversion of a conventional electric automobile into an unmanned ground vehicle (ugv). In *IEEE International Conference on Mechatronics*, pages 564–569.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *Computer Science*.

Stallkamp, J., Schlipsing, M., Salmen, J., and Igel, C. (2012). Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks the Official Journal of the International Neural Network Society*, 32(2):323.

Sun, Y., Wang, X., and Tang, X. (2014). Deep learning face representation from predicting 10,000 classes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1891–1898.

Suralkar, S. R., Karode, A. H., and Pawade, P. W. (2012). Texture image classification using support vector machine. *International Journal of Computer Technology & Applications*, 03(01).

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. *Computer Vision and Pattern Recognition*, pages 1–9.

Zaklouta, F., Stanciulescu, B., and Hamdoun, O. (2011). Traffic sign classification using k-d trees and random forests. In *International Joint Conference on Neural Networks*, pages 2151–2155.

Zeng, X., Ouyang, W., and Wang, X. (2013). Multi-stage contextual deep learning for pedestrian detection. In *IEEE International Conference on Computer Vision*, pages 121–128.

Zeng, Y., Xu, X., Shen, D., Fang, Y., and Xiao, Z. (2017). Traffic sign recognition using kernel extreme learning machines with deep perceptual features. *IEEE Transactions on Intelligent Transportation Systems*, 18(6):1647–1653.

Zhu, Z., Liang, D., Zhang, S., Huang, X., Li, B., and Hu, S. (2016). Traffic-sign detection and classification in the wild. In *Computer Vision and Pattern Recognition*, pages 2110–2118.