NOVATEUR PUBLICATIONS
INTERNATIONAL JOURNAL OF INNOVATIONS IN ENGINEERING RESEARCH AND TECHNOLOGY
[IJIERT] ISSN: 2394-3696 Website: ijiert.org
VOLUME 8, ISSUE 5, May.-2021

VEHICLES AND TRAFFIC SIGNS OBJECT DETECTION AND CLASSIFICATION USING DEEP NEURAL NETWORK APPROACHES IN REAL-TIME CONDITIONS

Han Honggui
Department of Information, Beijing University of Technology, China

Malicehnko Viktor
Department of Information, Beijing University of Technology, China
malicv@qq.com

ABSTRACT

This paper deals with a comprehensive project in order to create a system for detecting, recognizing, and classifying objects on the road in real-time conditions, with further use in unmanned vehicles. The evolution consists of several steps. The first is to create a classification model based on SVM for two classes of images recognition (vehicle and road signs in general); the second is an improvement of the constructed model using standard CNN. After gaining high-precision results, we moved on to the third step of development, which is building combined CNN model with YOLOv2, as well as performance cooperation with Faster R-CNN. Description of the third step is mainly discussed in this paper. The model trained and tested on our own data set, received from different countries (most of the video data was recorder in real-time, in Ukraine). Along the course, there were problems with signs detection, the localization on a video fragment and recognition accuracy. This led to the necessity of a thorough analysis of the model and additional algorithms. Finally, experiments demonstrated that our model is competitive with state-of-art models.

Keywords: Object detection and classification, CNN, YOLOv2, Faster R-CNN, object detector

INTRODUCTION

One of the most widespread research directions is Intelligent Transport Systems. Technological models the modern cars are equipped with are part of this trend. Increasingly, it becomes necessary to teach vehicles understand the current situation on the roads and involve “smart cars” in our daily lives. However, several resources have expressed doubts about the use of fully autonomous vehicles on the roads, saying that it poses a danger to people's lives. One way or another, we can claim that today there is no single system in the world that can be applied on any roads in different countries and ensure the reliability of its operation.

The purpose of this project is to create a model, which will be a part of Autonomous Vehicle system, able to detect and classify two classes of objects – “road signs” and “vehicles”, in real-time video. We collected own unique dataset, compiled two custom detectors for fast raw data labeling, and created mingled neural network able to track and classify multiple objects. Another significant and unique part of this work lies in the development of a model capable to adapt to countries and cities, where not much attention is paid to the development of such systems. Ukraine is one of such countries. Chaotic signs and traffic congestion make it difficult for existing systems to adapt to such conditions. Using modern technologies, we have developed a system that can cope in difficult conditions.

Why “traffic signs” and “vehicle”? These are two main categories constantly present on the roads. Moreover, driver's attention is focused primarily on these objects.

Today, the developed model has been tested for industrial purposes, classification of objects of other categories and has established itself as a fast-learning model with high accuracy.

RELATED WORKS

Preliminary work in the field of object detection refers to Pylyshyn and Storm in 1988 [2], and the road sign recognition – to Japanese researchers H. Akatsuka and S. Imai in 1987 [1]. Essentially, their studies are based on color processing system, which reduces the effects of brightness and shadow and image recognition. Since then, technology has progressed and a huge number of methods have been proposed that are capable of handling large data streams with high speed and accuracy. Speaking of road sign recognition, many researchers have dedicated their projects in this field using ROI [3], [4] and color segmentation [5]. Hasan Fleyeh [6] focused on building a model that showed good recognition results in severe weather conditions, using color-to-binary conversion, histogram equalization, changing saturation and hue. Other studies have also been conducted using HOG, Canny Edge detection, LBP, SIFT, SURF, BRISK etc. SURF method [7] proposed for the road signs classification. However, the system turned out to be not resistant to light changes in real conditions. Aside from that, for fare classification a huge image dataset need to be locally stored, which makes the system resource-intensive.

In recent years, we adopt neural networks capable of both detecting objects and classifying them simultaneously. Yann Le Cun et al.[13] suggested back-propagation to learn the convolution kernel coefficients directly from images. This made a significant contribution to the technological progress, which we took as a basis. In 1990, Yamaguchi et al presented Max pooling, a layer most commonly used in neural networks, as well as Faster R-CNN [8]. Besides, our experimental set up bears a close resemblance to such leading object detection and classification methods as SSD [9], YOLO [10], U-net [11] proposed by Google and NVIDIA.

EXPERIMENTAL METHODS

1.1.The first stage of research. Object classification based on SVM model

As a part of our SVM model, we introduce a cross-validation technique. Recently, this approach rarely used for in a tasks of intelligent transport systems however it simple, effective way to improve classification problem [19]. Implementing a standard KCV model with suggested parameters led as to very slow performance. It can be explained of external usage of memory. Improvement of this issue was achieved by retraining the SVM on the entire dataset. Should be noted that the hyperparameters in subset of the dataset contains of $\frac{(k-1)}{k}$ patterns. Architecture representation showed on the figure 1.

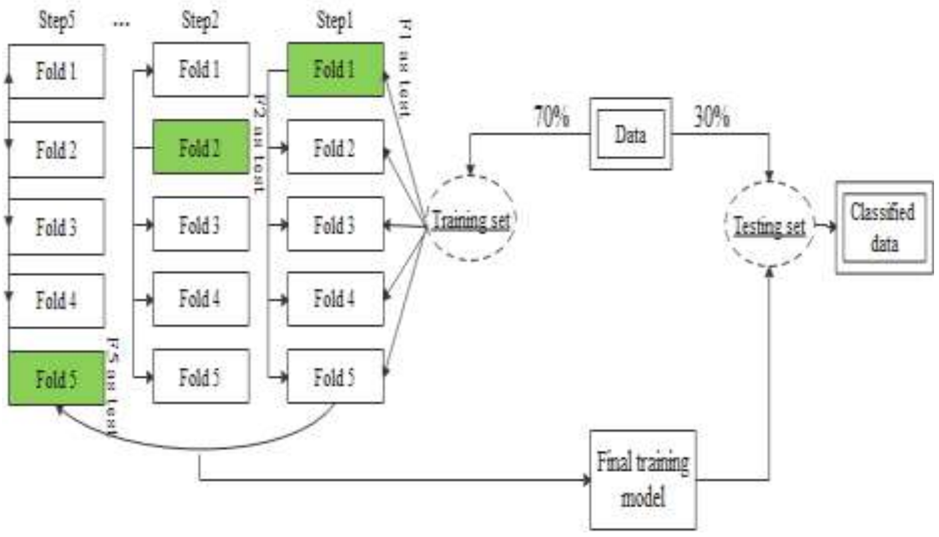

Figure 1. Cross-validation functioning with k=5 folds

The SVM model applying three classifiers – Linear, Quadratic and Medium Gaussian. As we know, the model consists of parameters and we find a parameter's values through the fitting process. This process involves some types of iterative algorithms that minimize the error. The algorithm depends on parameters that control how it works. Because most machine learning problems are non-convenes, which means the model depends on our choice of parameters. So a model may depend on parameters value. By changing these parameters we can find a better model. The Bayesian optimization algorithm attempts to minimize a scalar objective function f(x) for x in a bounded domain. One of the advantages of Bayesian Optimizer [20][21] is that it doesn't only look at how accurate the model is but also analyzes how long it does take for a model to train. Some hyperparameters cause the training time to increase by a factor of 100. That may influence badly if we try to achieve improvement in time. We choose expected improvement per second plus as Bayesian optimization configuration, which penalizes as hyperparameter values that are expected to take a very long time to train. It prefers to acquire points that are not only likely to be good, but that is also likely to be evaluated quickly.
Using optimization parameters (see Figure 2), the Gaussian classifier coped with the task of classifying "road signs" and "vehicle" with an accuracy of 99.4%.

Table 1. Classification results of "road signs" and "vehicle" by applying SVM

SVM classifier	Time	Kernel scale	TPR	FPR	Accuracy (%)
Linear	4.88	1	0.95	0.0125	96.9
Quadratic	4.725	1	0.9875	0.025	98.1
Gaussian	4.434	16	1	0.025	98.8
Optimized Gaussian	31.62	27.561	1	0.0125	99.4

As shown in Table 1 the increase of accuracy entails a significant increase of training time. It should be noted that the studies were carried out on a database consisting of 2000 cropped images, containing only objects of each category and at the same time the system classified, but did not detect the object against the general background.

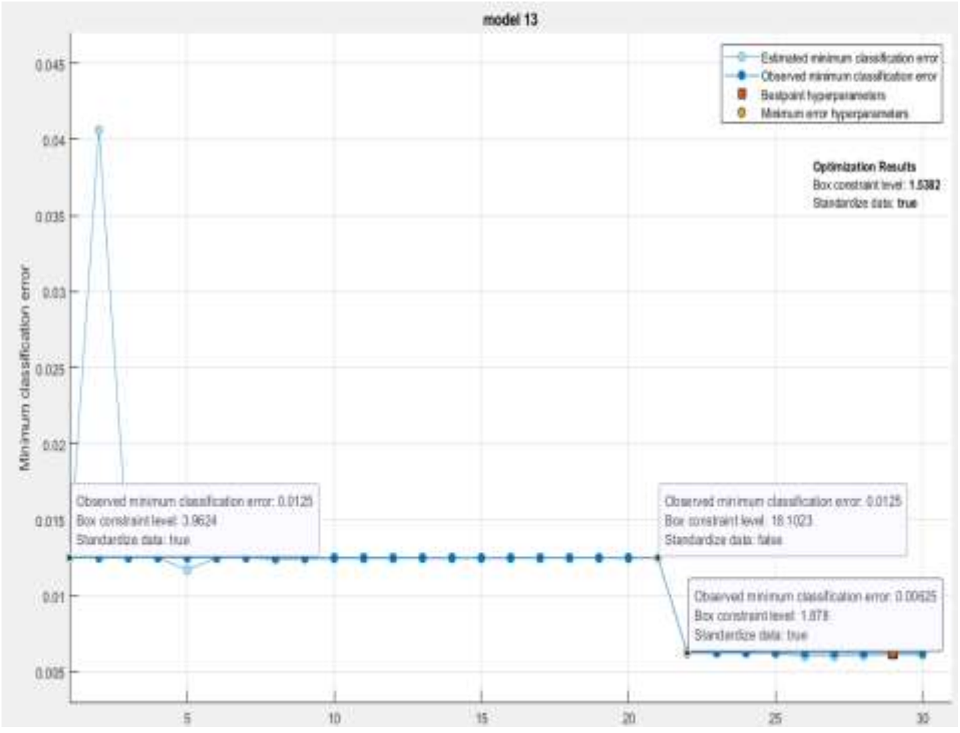

Figure 2. Minimum classification error of trained Optimized Gaussian SVM within 30 iterations

1.2. The second stage of research. Convolutional neural network implementation

We built a CNN model capable of classifying two categories, which significantly improved accuracy and learning rate compared to SVM. The network consists of three convolution layers and a fully-connected layer. The influence of the following hyper parameters (quantity of layers, the number and size of filters and the activation function) was studied in a low-level. It has been experimentally proven, that the best results are achieved with parameters shown in Table 2.

Table 2. CNN model parameters with 99.8% classification accuracy

Layer	Filter value	Kernal size	Pooling
Conv_layer_1	32	(3,3)	(2,2)
Conv_layer_2	64	(3,3)	(2,2)
Conv_layer_3	64	(3,3)	(2,2)
Full_conected	32	-	-

Inspired by Yingying Wang et al. [12] we modified standard activation function Relu by replacing it with proposed LS–Relu function. LS-ReLu activation function can be described by the equation below:

$$f(x) = \begin{cases} x/1 + |x|, & x \leq 0 \\ \max(0, x), & k \geq x > 0 \\ \log(a * x + 1) + |\log(a * k + 1) - k|, & x > k \end{cases} \tag{1}$$

Thus, we were able to avoid overfitting the model during training and reduce the problem of fluctuations. Tensorflow analysis (see Figure 3) was used to predict the accuracy of a model showed in Table 2.

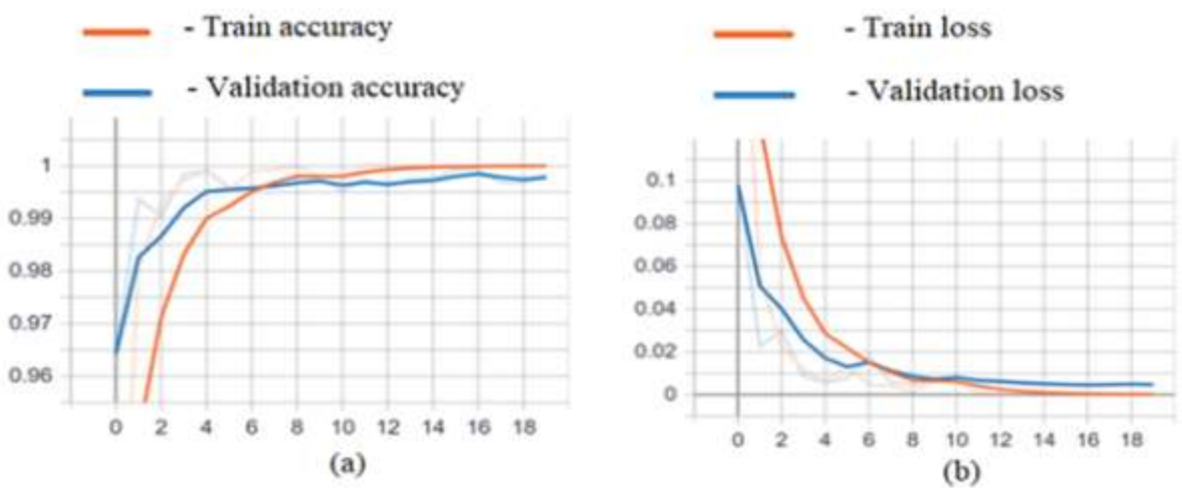

Figure 3. (a) Accuracy Graph of trained model, (d) Loss Graph of trained model. Graphs represent a model with 3 convolutional layers where 1st layer – 32, 2nd layer -64, and 3d layer - 64. Smoothing parameter was set as 0.6.

1.3.The third stage of research. CNN and Yolov2 for vehicle and traffic sign detection and classification

Stage 1 and stage 2 showed good results when classifying the two types of objects; however, after applying the CNN model to a video stream, processing slows down, and the classification accuracy is not acceptable. To overcome this problem, we modify and combine our model with Yolo2 architecture.

1.3.1. Data preprocessing

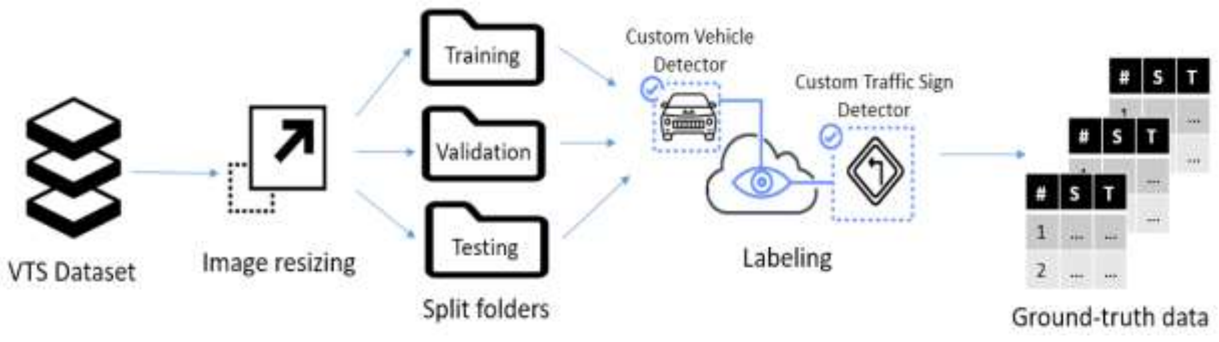

Figure 4. Data preprocessing architecture

Image preprocessing is a common step, yet in this paper, we suggested an approach and generated a code, which allowed us rapidly preprocessing and labeling any new coming data. Visualization of data preprocessing shown in Figure 4.

Our first steps proceed largely in the same way as suggested in [10]. For correct Yolov2 training, the input Vehicle Traffic Sign (VTS) dataset is cropped to size (416 x 416). The data then splits into three folders – training data 60%, validation data 30%, and test data 10%. Next step distinguishes our approach from classical data processing as we add extra steps. To label the entire database, we created two client detectors using the Aggregated Channel Functions (ACF). To train Custom Vehicle Detector (CVD) and Custom Traffic Sign Detector (CTSD), we used a small amount of our data (about 400 images); all the remaining frames were marked using CVD and CTSD. CVD and CTSD are used to quickly tag additional data and to augment the database.

1.3.2. Network architecture

Table 3. CNN and Yolov2 based network architecture for object detection and classification

№	Type	Filter Value/ channel
1	Input	128 x 128 x 3
2	Convolution 1	16 3 x 3 x 3
3	Batch normalization 1	16
4	LS-Relu 1	-
5	Max Pooling 1	2 x 2
6	Convolution 2	32 3 x 3 x 16
7	Batch normalization 2	32
8	LS-Relu 2	-
9	Max Pooling 2	2 x 2
10	Convolution 3	64 3 x 3 x 32
11	Batch normalization 3	64
12	LS-Relu 3	-
13	Max Pooling3	2 x 2
14	Convolution 4	128 3 x 3 x 64
15	Batch normalization 4	128
16	LS-Relu 4	-
17	Yolo v2 Convolution1	128 3 x 3 x 128
18	Yolo v2 BN 1	128
19	Yolo v2 Relu 1	-
20	Yolo v2 Convolution 2	128 3 x 3 x 128
21	Yolo v2 BN 2	128
22	Yolo v2 Relu 2	-
23	Yolo v2 class Convolution	128 1 x 1 x 128
24	Yolo v2 Transform	-
25	Yolo v2 output layer	-

Table 3 shows the final architecture of CNN- YOLOV2 training model and contains 25 layers. Based on our previous work and CNN parameters with best results, we created custom model layers with 4 convolutional layer. To stabilize learning process, each convolutional layer was followed by batch normalization layer that also allow us using less epochs for training and significantly reducing training time and making a model more stable during training. After batch normalization layer, we have applied purposed modified Re-lu activation (1).

After setting convolutional layers, we extract YOLOv2 subnetwork, followed by two convolutional layers, standard Re-lu activation function and batch normalization layers.

1.3.3. Anchors

As we follow a task of multiple object tracking with a different size and scales, anchors were estimated and customized according to our dataset. To combine all anchor boxes according to their sizes and shapes k-medoids clustering [15] is used. After calculating numbers of boxes per cluster, we compute cluster centers that used then for computing IOU value. Intersection over union (IOU) is used as a distance metric. It's value response on how well anchor boxes overlap with training data boxes. IOU can be described as follows:

$$\mathrm{IOU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}, \qquad (2)$$

While the union can be defined as shown below:

$$\mathrm{Union} = \frac{\mathrm{area}(A \cap B)}{\mathrm{area}(A \cup B)}, \qquad (3)$$

where A and B are bounding boxes.

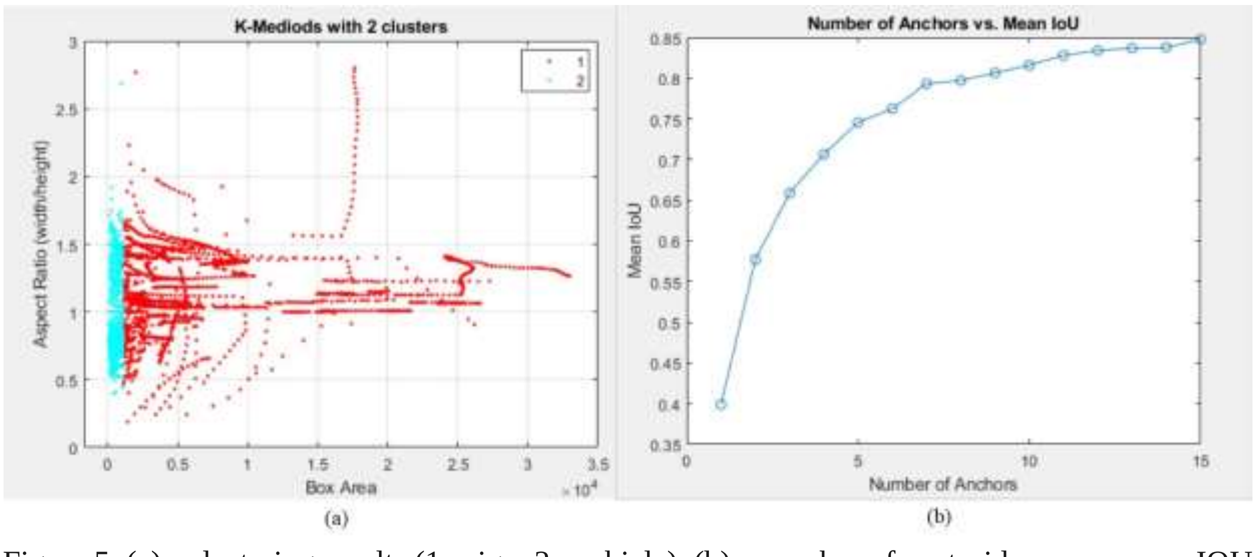

Figure 5. (a) - clustering results (1 –sign, 2 –vehicle), (b) – number of centroids versus mean IOU

Figure 5.(b) shows that if the number of the anchor (or centroid) increases, the ratio of the mean IOU increases respectively. This also indicates that we avoid multiple detection of the same object. It is important to keep in mind, that IOU must be greater than 0.5. In our case mean IoU = 0.896.

1.3.4. Model Training

The proposed network, trained within ~10 000 iterations, while instead of applying Adaptive Moment Estimation at CNN model (in 3.3) we used stochastic gradient descent with momentum [14] and a training mini batch size of 16. The learning rate value, that is usually specified as a rate between 0.0 – 1.0, in our model is set to 0.001. In order to increase optimization process the speed momentum value was set to 0.9. Earlier, when building a CNN model, we found that increasing filter values for each following layer gives the best results. Based on this, we doubled filter size value as 16, 32, 64 and 128 respectively.

RESULTS

1.4. Dataset

We have gathered our own database - vehicle and traffic sign (VTS), which in total consists of about 4000 images of both categories of representative frames with a resolution of -10920x 1080pi, filmed in different places and cities (Figure 6). This is another function that makes a model unique and allows implementing it into any scenario. Cities of Ukraine are one of the priority scenarios; since today there are a huge number of signs on the roads, but unclear visibility or noise from the environment makes it difficult for existing models of “smart cars” to detect objects with high accuracy.

Figure 6. Visualization of VTS dataset. Full framed images with multiple object on it. Representative images describe a real-time road situation including traffic jams, half-cover traffic signs, blurry objects, day and night scene

As mentioned earlier, further, the VTS database will expanded, in order to achieve higher precision and classify more objects.

Previously, it was said that the CVD and CTSD models detect the corresponding objects on raw images to create a ground truth table, which is then used to train the neural network.

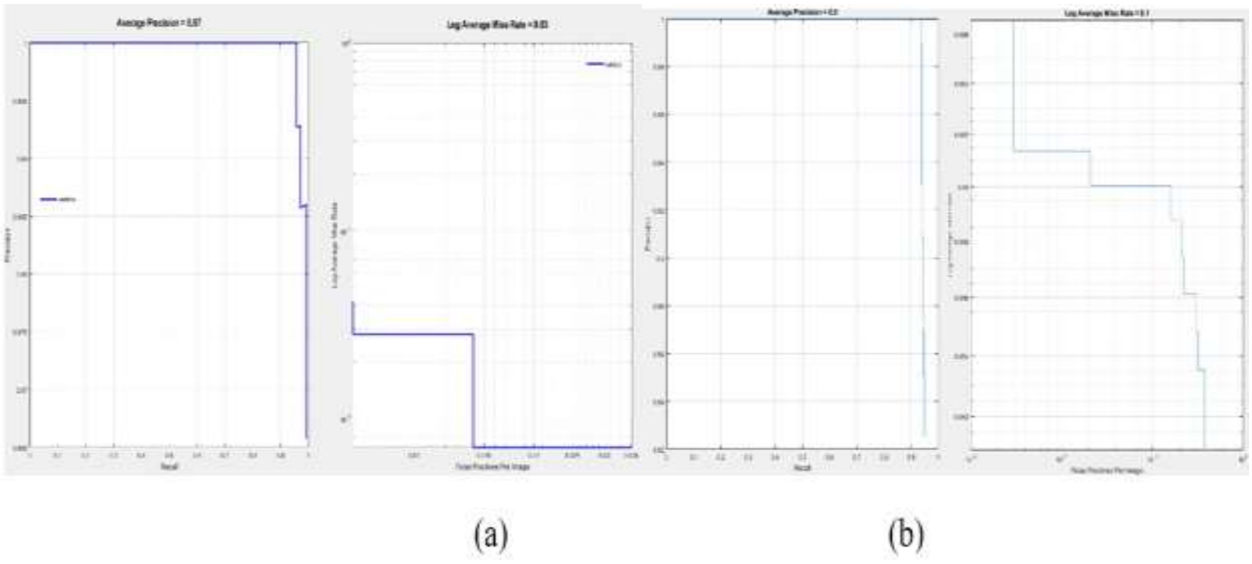

Figure 7. Custom detector Precision and Log Average Miss Rate results. (a) CVD within 2500 image dataset; (b) CTSD within 2000 image dataset

The left part of the image displays Average precision - 0.97% and average miss rate - 0.33% for the “vehicle” class. On the right side of the graph (b) – the results of the Custom Traffic Sign Detector, where the Average precision is 0.9% and the average miss rate is 0.1%. One image may contain more than three labeled objects of the same category, thus we triple the ground truth data. Results shown in Figure 7 indicate excellent object detection, but still, cannot be used as fundamental detector for autonomous vehicle. We believe, that the result of detector emphasizes the validity of our detector model. Further training was performed using CNN and YOLOv2.

1.5. Training

Once the training, validation and test data is all set, another challenging task is to tune a training model. The main parameters we operated on: number of convolution layer, learning rate, mini batch size, and number of epochs.

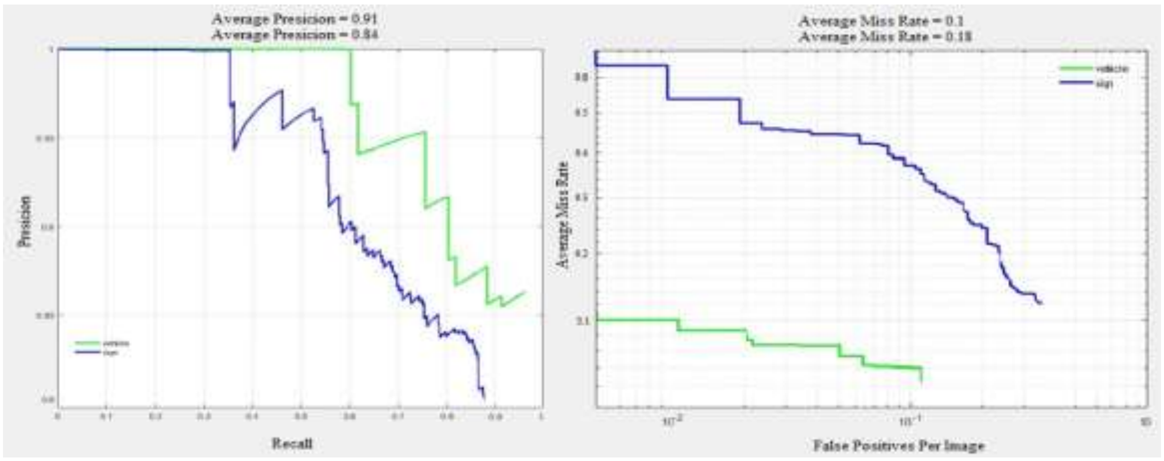

Figure 8. Average precision and Average Miss Rate for CNN – Yolo architecture. Blue curve represents "traffic sign" class and green curve – "vehicle" class

Average Precision (AP) and Average Miss Rate (AMR) shown in Figure 8 correspond to neural network in Table 3. AP plots how precisely model reacts for each recall and AMP responds for false positive detections per each test image. From Figure 8 we see – the higher precision is the smaller miss rate we get. As have been expected, AP correlation of "traffic sing" is lower than for the "vehicle", due to smaller amount of ground truth data. Despite this, we can still state that, AP=0.84 is higher in contradiction with earlier findings [17],[18]. Equations that estimate precision, recall and miss rate can be defined as follows:

$$\text{Precision} = \text{TP}/\text{TP} + \text{FP} \qquad (4)$$

$$\text{Recall} = \text{TP}/\text{TP} + \text{FN} \qquad (5)$$

$$\text{Miss Rate} = \text{FN}/\text{TP} + \text{FN} \qquad (6)$$

More configurations tested during designing a CNN-Yolo network and performance results can be found in Table 4. Results are based on the same structured model, detecting two categories with multiple objects localization and classification.

Table 4. Combination of parameters used during training within different amount of dataset

Epochs	Dataset (images)	Learn Rate	Vehicle		Traffic Sign	
			AP, %	AMR, %	AP, %	AMR, %
60	~500	0.1 – 0.01	0.54	0.66	0.41	0.63
80	~500	0.01 - 0.001	0.84	0.17	0.57	0.54
100	~500	0.01 - 0.001	0.71	0.40	0.67	0.62
Epochs	Dataset (images)	Learn Rate	Vehicle		Traffic Sign	
			AP, %	AMR, %	AP, %	AMR, %
80	~4000	0.01 - 0.001	0.91	0.1	0.84	0.18
100	~4000	0.02 - 0.001	0.89	0.2	0.82	0.20

Based on a variety of test studies, a model was formed with the main parameters indicated in the Table 5.

Table 5. Eventual parameters of final neural network architecture

Parameters	Layers	Optimizer	Learn rate	Mini Batch	Epochs	Iterations
Value	25	SGDM	10^{-3}	16	80	10^{4}

The task of detecting and classifying road signs and a car was tested on different models of neural networks. After getting satisfactory results using a combination of CNN and YOLOv2, we repeated the training with similar parameters on other models that we studied earlier. The Figure 9 plots a comparison of different models in terms of time spent on training and achieved accuracy.

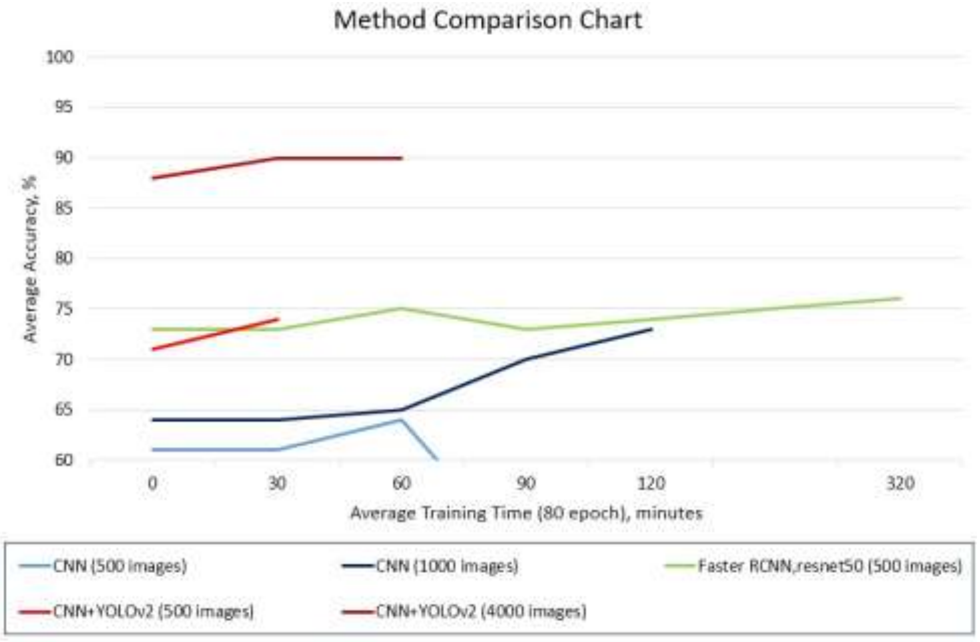

Figure 9. Method cooperation based on average training time and average accuracy

Average accuracy was estimated by formula:

$$\mathrm{Av.Accuracy} = \mathrm{AP_i(vehicle)} + \mathrm{AP_i(sign)}/2, \qquad (7)$$

where AP – is an average precision results for vehicle and traffic sign classification, for each time we trained a model.

The results from such analyses prove that classic CNN showed the least accuracy, while requiring more training time using significantly less data as compared to other cases. Faster R-CNN showed better accuracy than the previous network, but it took more than 3 hours to train a model with 500 images. The combined model proposed in this paper positions itself as the most efficient and fastest-learning model. Comparison of our model with other state-of-art approaches in same field shown in Table 6.

Table 6. Real - time vehicle and traffic sign detection compared to other related works

Method	Detected classes	AP,%
Our model	Vehicle	0.91
	Traffic sign	0.84
Yolov3 [16]	Multiple class	0.69
TTTL [17]	“go” sign	0.91
	“stop” sign	0.79
ACF-RP-YOLO [18]	Cyclist	0.83

The trained model was tested on real-time video (Figure 10). The system has shown a stable result on frames with increased noise and poor visibility, despite that in some cases when objects are too small, model couldn't recognize them. However, that was expected from the precision results shown in Figure 8 and can be improved by re-training a model through labeling undetected objects.

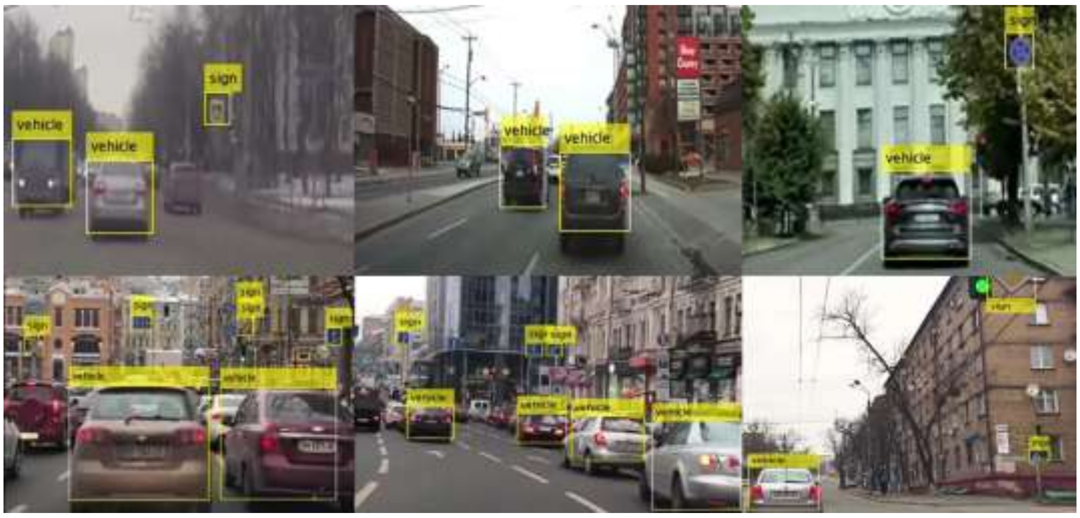

Figure 10. Demonstration of the CNN-YOLOv2 model for detecting vehicles and road signs on video filmed in real time. Detected objects are fitted in bounding boxes. Each category is marked accordingly.

1.6. Hardware

The work was performed on a DEL XPS laptop, Inter core i7. Environment for developing SVM models, Faster R-CNN and CNN-YOLOv2 - Matlab 2020, for CNN - Spider, implementing Tensorflow for training process analysis using Python language. The characteristics of the CPU and GPU are shown in the Table 7.

Table 7. CPU and GPU performance results based on comparison with top hardware

	Results for data-type 'double'(in GFLOPS)			Results for data-type 'single'(in GFLOPS)		
	MTimes	Backlash	FFT	MTimes	Backslash	FFT
GPU (GeForce GTX 1650)	111.21	92.34	47.00	2782	857	195
CPU	177.78	97.62	5.24	179.03	178	8.11

CONCLUSION

The main purpose of our research was to create a model capable to detect and recognize two classes of objects on the video. Our first step was to create a classification model using SVM. Despite good results, the model is well suited for classifying images rather than videos. So, we considered alternative approaches that met our expectations. Another important aspect was the application of our model on roads with poor conditions, congested roads, etc. but available image databases did not overwhelm our requirements. Thus, we have created our own database. In addition, a custom detector was created, which allows us quickly processing new images and re-training the model. During the development, we faced problems to get a satisfactory ratio of Precision and Recall, in spite of this we were able to fix this by increasing the input data and adjusting the hypermaters of the network.

As a result, we built a model using CNN and Yolov2, which showed it competitive results with the available state-of-art methods. Once we got good results, we carried more experiments using standard CNN and Faster R-CNN, which confirmed the feasibility of using our model.

We also used a GPU that allow us reducing the training time. We compared our graphic processing unit; this makes it possible to evaluate the increase in dance performance when improving the GPU.

In future work, we plan to supplement our model using segmentation methods to recognize pedestrian zones, people and roads, increase the database of road signs, and learn to distinguish them. Then, deploy a model on a single-board computer on a vehicle.

REFERENCES

Akatsuka H, Imai S. Road signposts recognition system. SAE Tech. Pap., 1987. https://doi.org/10.4271/870239.

Storm RW, Pylyshyn ZW. Tracking multiple independent targets: Evidence for a parallel tracking mechanism. Spat Vis 1988;3:179–97.

Radzak MY, Alias MF, Arof S, Ahmad MR, Muniandy I. Study on Traffic Sign Recognition. Int J Res Stud Comput Sci Eng 2015;2:33–9.

Liang M, Yuan M, Hu X, Li J, Liu H. Traffic sign detection by ROI extraction and histogram features-based recognition. Proc Int Jt Conf Neural Networks 2013. https://doi.org/10.1109/IJCNN.2013.6706810.

Youssef A, Albani D, Nardi D, Bloisi DD. Fast traffic sign recognition using color segmentation and deep convolutional networks. Lect Notes Comput Sci (Including Subser Lect Notes Artif Intell Lect Notes Bioinformatics) 2016;10016 LNCS:205–16. https://doi.org/10.1007/978-3-319-48680-2_19

Fleyeh H. Traffic signs color detection and segmentation in poor light conditions. Proc 9th IAPR Conf Mach Vis Appl MVA 2005 2005:306–9.

Duan J, Viktor M. Real time road edges detection and road signs recognition. ICCAIS 2015 - 4th Int Conf Control Autom Inf Sci 2015:107–12. https://doi.org/10.1109/ICCAIS.2015.7338642.

Ren S, He K, Girshick R, Sun J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Trans Pattern Anal Mach Intell 2017;39:1137–49. https://doi.org/10.1109/TPAMI.2016.2577031.

Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, et al. SSD: Single shot multibox detector. Lect Notes Comput Sci (Including Subser Lect Notes Artif Intell Lect Notes Bioinformatics) 2016;9905 LNCS:21–37. https://doi.org/10.1007/978-3-319-46448-0_2.

Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: Unified, real-time object detection. Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit 2016;2016-Decem:779–88. https://doi.org/10.1109/CVPR.2016.91.

Hegde K, Yu J, Agrawal R, Yan M, Pellauer M, Fletcher CW. UCNN: Exploiting computational reuse in deep neural networks via weight repetition. Proc - Int Symp Comput Archit 2018:674–87. https://doi.org/10.1109/ISCA.2018.00062.

Wang Y, Li Y, Song Y, Rong X. The influence of the activation function in a convolution neural network model of facial expression recognition. Appl Sci 2020;10. zttps://doi.org/10.3390/app10051897.

Becker S, le Cun Y. Improving the Convergence of Back-Propagation Learning with Second Order Methods. Proc 1988 Connect Model Summer Sch 1989:29–37.

Ruder S. An overview of gradient descent optimization algorithms 2016:1–14.

Park HS, Jun CH. A simple and fast algorithm for K-medoids clustering. Expert Syst Appl 2009;36:3336–41. https://doi.org/10.1016/j.eswa.2008.01.039.

Redmon J, Farhadi A. YOLOv3: An incremental improvement. ArXiv 2018.

Lu Y, Lu J, Zhang S, Hall P. Traffic signal detection and classification in street views using an attention model. Comput Vis Media 2018;4:253–66. https://doi.org/10.1007/s41095-018-0116-x.

Liu C, Guo Y, Li S, Chang F. ACF based region proposal extraction for YOLOV3 network towards high-performance cyclist detection in high resolution images. Sensors (Switzerland) 2019;19. https://doi.org/10.3390/s19122671.

M. Kaariainen, “Semi–supervised model selection based on Cross–Validation”, Proc. of IEEE Int. Joint Conf. on Neural Networks 2006,IJCNN 2006, pp. 1894–1899, 2006.

C.-W. Su,C.-C. Chang, C.-J. Lin, “A practical guide to Support Vectorclassification”, Technical report, Dept. of Computer Science, NationalTaiwan University, 2003

J. Bruna and S. Mallat, Invariant scattering convolution networks, IEEE Trans. Pattern Anal. Mach. Intell., vol. 35, no. 8, pp. 1872-1886, Mar. 2013

Y. Li, S. Gu, V. Gool, and R. Timofte, Learning Filter Basis for Convolutional Neural Network Compression: Supplementary Material, arXiv:1908.08932v2, Dec 2