International Journal of Recent Advances in Multidisciplinary Topics
Volume 2, Issue 3, March 2021
https://www.ijramt.com | ISSN (Online): 2582-7839

Traffic Sign Detection and Recognition

Harshil Patel[1], Yash Kundariya[2], Fenil Patel[3*], Harshal Shah[4]
[1,2,3,4]Department of Computer Science and Engineering, Parul University, Vadodara, India

***Abstract*: The objective of this work is the development of an algorithm for the automatic recognition of traffic signs. Two major problems exist in the process of detection and recognition of traffic signals. Road signs are frequently occluded partially by other vehicles and many objects are present in traffic scenes which make the sign detection hard and pedestrians, other vehicles, buildings and billboards may confuse the detection system by patterns similar to that of road signs. Also, color information from traffic scene images is affected by varying illumination caused by weather conditions, time (day night) and shadowing. This method detects the location of the sign in the image, based on its geometrical characteristics and recognizes it using color information. Partial occlusion is dealt by the use of the Hough Transform.**

***Keywords*: Classification and Regression Tree (CART), Quality Assurance (QA), Predictive Model Markup Language (PMML), Convolutional Neural Network (CNN), Support vector.**

1. Introduction

Traffic signs regulate traffic in all countries. In order to avoid confusion and improve recognition across countries, efforts are being made to harmonise the appearance of traffic signs. This is the aim of the Vienna Convention on Road Signs and Signals, which was developed in 1968 and retired by numerous countries. With the introduction of Advanced Driver Assistance Systems and the goal of autonomous driving, automatic traffic sign recognition (TSR) has become more important. Video cameras are installed in vehicles for this purpose. The images are continuously recorded and evaluated. The recognition of traffic signs is done in two steps: In the first step, the location of the traffic sign is detected and the track sign is extracted. The classical detection methods can be divided into color based and shape based [1][2]. Based on the camera image of the captured environment, a bounding box can be found which encloses the detected track sign.

2. Materials and Methods

1) CNN for object detection

There have been several classic object recognition networks in the last few years [5], for instance AlexNet [6] (2012), VGG [7] (2014), GoogLeNet [8] (2015–2016), ResNet [9,10] (2016), SqueezeNet [11] (2016), Initially, the convolutional neural network was developed and enlarged to achieve greater precision accuracy. However, networks have grown smaller and more efficient in recent years. In highly accurate target sensing tasks, the new deep learning algorithms, especially those that apply to CNN, such as You Only Look Once, (Yolo) v3, show huge potential. The multiscale and sliding window approach that produces bounding boxes and scores via CNN can be implemented efficiently within a ConvNet, and R-CNN. Besides, R-CNN is also expensive in time and memory, as it executes a CNN forward-pass for all object proposal without sharing computation. To solve this problem, spatial pyramid pooling networks (SPPnets) were introduced to increase the efficiency of R-CNN through computational sharing. SPPnet calculates feature maps from the entire input image only once and then supplies feature in arbitrary-size sub-images to generate fixed-length representations and for detectors training. However, SSPnet eliminates the replicated evaluation of convolutional feature maps, it still needs training in a multi-stage pipeline as the fixed-length feature vectors generated by numerous SPP layers are also moved on to fully-connected layers. Therefore, the whole process is still slow. Certain techniques, including single shot multiBox detector (SSD) and Yolo, exemplify all the processing in a single fully-convolutional neural network rather than making a persistent pipeline of regional proposals and object classification. This knowledge conducts to a significantly more expeditious object detector. The one-stage method relies on the end-to-end regression approach technology. Yolo V3 applied Darknet-53 to substitute Darknet-19 as the backbone network and employed multiscale prediction.

2) Spatial Pyramid pooling

In terms of object recognition tasks, spatial pyramid pooling (SPP) was significantly victorious. Consider its severity, it is competing among methods that use more complicated spatial models. For the interpretation of the spatial pyramid, the image is split into a range of finer grids at each level of the pyramid. In addition, it is commonly-known as spatial pyramid matching (SPM), a development of the bag-of-words (BoW) model, which is one of the most famous and successful methods in computer vision methods. SPP has continued been an important component and superior system to win the competition in the classification and detection before the recent ascendance of CNN.

Some benefits of SPP could be explained as follow: First, SPP can produce a fixed-length output despite the input dimension. Second, SPP applies multi-level spatial bins, while the sliding window pooling employs just a single-window size. Next, SPP allows us not only to generate images from arbitrarily sized images for testing but also to feed images with different sizes and scales during training. Additionally, training

*Corresponding author: thefenilpatel09@gmail.com

with variable-size images raises invariance in size and decreases overfitting. In addition, SPP is extremely effective in object detection. In the foremost object detection method R-CNN, the features from candidate windows are obtained through deep convolutional networks. Furthermore, SPP can combine features derived at variable scales to the flexibility of input scales. CNN layers receive some despotic input sizes, but they generate outputs of variable sizes. The softmax classifiers or fully-connected layers require fixed-length vectors. Such vectors can be generated by the BoW approach [29] that pools the features together at the same time. SPP improves the performance of BoW in that stage, and it can preserve spatial information by pooling in local spatial bins. The space bins have proportional sizes to the image size, and regardless of the image size the number of bins is fixed. On the contrary, the sliding window pooling of former deep networks and the number of sliding windows depends on the scale of the data. Hence, to implement the deep network for images of arbitrary sizes, the last pooling layer will substitute with an SPP layer. In the particular spatial bin, we pool the replies of each filter and apply max pooling. The outputs of the spatial pyramid pooling are kM dimensional vectors with the number of bins indicated as M. Further, k is the number of filters in the latest convolutional layer. The fixed-dimensional vectors are the input to the fully-connected layer. By using SPP, the input image can vary in size, which allows not only arbitrary aspect ratios but also enables absolute scales. The input image can resize to any scale and adopt an identical deep network. When the input image is at diverse scales, the network with the equivalent filter sizes will extract features at various sizes and scales. A network structure with an SPP layer can be seen in Figure 1. In our work, the SPP blocks layer is inserted to the Yolo V3, Resnet 50, Densenet, and Tiny Yolo V3 configuration file. Moreover, we use the same SPP blocks layer in the configuration file with a spatial model. The spatial model uses down sampling in convolutional layers to receive the important features in the max-pooling layers. It applies three different sizes of the max pool for each image by using [route]. Different layers −2, −4 and −1, −3, −5, −6 in $conv$□ were uses in each [route].

3) Object Detection Architecture

Deep learning algorithms are used to get the task done in this particular algorithm. Fully Conventional neural network is used for traffic sign recognition whereas Conventional Neural Network is used for object detection. Once an image is sent through CNN network it is passed through a normalization layer which aids in convergence speed. Our paper is based on the work established here. The processing of the image is done in pixel and this method is found to be efficient and robust. Joint transformation correlation with image segmentation is sided by shape analysis. Out of all methods that are quoted here this one has the capability to detect signboard of any color and shape. The detection that is the region of interest is found by clustering the pixels. Then by using Fourier joint transformation correlation (FJTC) this algorithm tries to match the sign recognized ROI to the known shapes and signs. Once the system is trained with different signs in a variety of environments then it is ready to detect and recognize the signs.

This paper deals with no only the software and also the hardware built of the system. This is done with the help of Virtex4 FPGA family which in turn is connected to the mounted camera of the car, then the appropriate result is displayed on the dashboard. This deliberately uses artificial neural network with RGB colour space ratio. A road sign recognition application called Self organizing maps (SOP) are used. Despite all good aspects of this algorithm it gives non adequate results when there is change in luminosity.

In here an embedded System is built to recognize and detect the required signs. Illumination is one of the major problems faced. To resolve this issue image is pre-processed which normalizes RGB colour using multi-scale retinex. Then region of interest is detected using colour segmentation and then Hough transform to detect the known forms. And then in secondary processing, the image is sent through a normalized cross-function to cross verify with the existing database. This algorithm compared to other ones has higher detection rates.

3. Methodology

1) Software Development Methodology

To classify the traffic signs, we should build a CNN (Convolutional Neural Networks) model and we should train and validate the model and finally test the model with the test dataset. In the training dataset, we divided images into 43 folders so that each folder represents a different image, the range of the folder is from 0 to 42. Using the OS module, we iterated over all the classes and add images and their respective labels in the data and label list. The PIL (Python Image Library) library is used to open image content into an array.

After storing all the images and their labels into lists we need to convert the list into NumPy arrays for feeding to the model. The shape of data is (n,x,y,z) which means that there are n images of size x*y pixels and the last z means the data contains colored images. With the sklearn package, we use the train_test_split() function for splitting data arrays into subsets (for training data and for testing data) and using Keras.utils.to_categorical() method, we convert class vectors to the binary class matrix. To classify images into their respective categories, we need to build a CNN (Convolutional Neural Network) model as CNN is best for image classification purposes.

2) How we built the convolutional neural network

We use a sequential model. So that, the layers in the network will be added in sequence. Add 2 convolutional layers with Kernal size of 5*5 and 32 filters, then the pooling layer to down sample the input representation(image) and allowing for assumptions to be made about features contained in the sub-regions binned, it is mainly used to extract sharp and smooth features. it is also done to reduce variance and computations. next max pool 2d layer (max-pooling) which helps in extracting features like edges, points, etc. Next dropout layer which helps in preventing overfitting of the neural network.it ignores some of its nodes to prevent overfitting. Next, we added another 2 layers with 64 filters and with 3*3 kernel size with activation function (relu), next max pool 2d layer (max-pooling) which helps in extracting features like edges, points, etc. again drop out layer for preventing the model from overfitting. Next, flatten layer which converts the data into a 1-dimensional array for inputting it into the next layers. Next the dense layer which takes the input from the activation function and 256 are the number of classes. Next, the dropout layer.

Lastly, the dense layer with 43 classes and with the softmax as the activation function which helps the neural network to run a multi-class function which helps us to determine the probability of a particular traffic sign is in the image as well as the probability of additional objects is included as well. After building the model architecture, we need to train the model using model fit(). With the batch size of 64 and after 15 epochs the accuracy was stable at it is 95% and using matplotlib, we need to plot the graph for both accuracy and loss.

The dataset contains a test folder and in a test.csv file, it has the details related to the image path and their respective class labels. From there, we extract the image path and labels using pandas. Then for predict the model, we have to resize our images to 30×30 pixels and send all the image data into a new NumPy array. From the sklearn metrics, we import the accuracy score and observed how our model predicted the actual labels. Finally, the model achieved an accuracy of 95%.

Finally, we saved the model that we have trained using Keras model save () function.

After saving the model, we built a graphical user interface (GUI) for traffic sign classifier with Tkinter, it is a GUI toolkit in the standard python library. So we have to create a new file in the same project folder and we have to load the previously saved trained model using Keras. And then we build the GUI for uploading the image or clicking an image and a button to classify which class the classify () function, the function which converts the images into the dimension of shape (1,30,30,3). To predict the traffic sign we have to provide the same dimension we have used when building the model so we use the same dimension. Then we predict the class between (0-43) which represents.

4. Algorithm

The algorithm of our proposed work is described as follows:

- Step 1: Upload the image either from camera or from directory.
- Step 2: The uploaded image will be sent through trained model.
- Step 3: Analyze the image for required parameters and classify the image accordingly.
- Step 4: If it is a traffic sign, it will identify the particular traffic sign and the output will be given in the form of text and voice notification will be given.
- Step 5: Else it will not give the output.

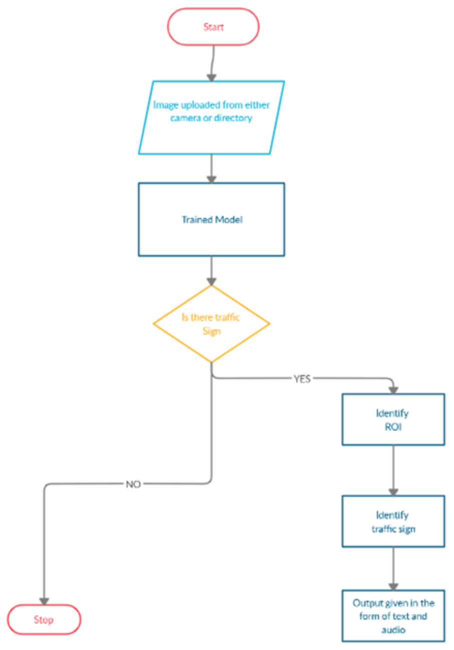
Start
Image uploaded from either camera or directory
Trained Model
Is there traffic Sign
YES
NO
Identify ROI
Identify traffic sign
Output given in the form of text and audio
Stop

Fig. 1. Flowchart of the algorithm

5. Results and Discussion

To check the above model and to get the accuracy value and loss value for both training and validation and the model took 15 epochs to maintain the accuracy and we compared the accuracy and the number of epochs, the ouput shown in Fig.2 and also compared loss and the number of epochs, the output shown in Fig.3.

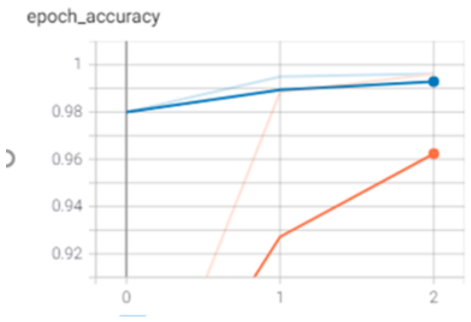

Fig. 2. Epoch gain

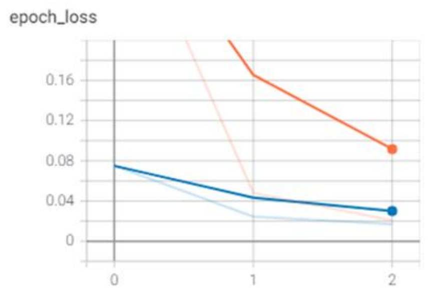

Fig. 3. Epoch Loss

6. Conclusion and Future Works

The rapid growth of technology in recent years has brought many changes into the people's lives. There has been growth of technology in every field and which helps people to do their tasks so easily from a small task to complex management system. The growth of technology in automobile industry is making the people's driver so easier and comfortable. And many companies like tesla, google, Mercedes-Benz are working on autonomous cars or self-driving cars, and the traffic sign recognition is one of the important factors to be considered while building a self-driving car. So, we build a model which recognizes the traffic sign through convolution neural network and alerts the driver or the system that there is a particular traffic sign ahead via text and voice notification. So that the driver will be alerted in normal cars and system will be alerted in autonomous cars and one more factor considered to build this model is the driver's carelessness or negligence towards the traffic signs, many of the driver will pay less attention toward traffic sign and suffer paying fines for speed limit etc. and some of the accidents also occur due to this. So this model recognizes the traffic sign and alerts the driver so that he will act accordingly. For example, example if there is speed breaker ahead, there will be a traffic sing of speed breaker and our model detects the speed breaker sign and indicates the system that there is a speed breaker ahead. The accuracy of the model obtained is around 95%. With this percentage, it can recognize the maximum number of traffic signs and the number of different traffic signboards it can recognize is 44. So with this model, we can recognize the traffic sign in front of the car and alert the driver with a voice message.

References

[1] Y. Aoyagi and T. Asakura, "A study on traffic sign recognition in scene image using genetic algorithms and neural networks," Proceedings of the 1996 IEEE IECON. 22nd International Conference on Industrial Electronics, Control, and Instrumentation, Taipei, Taiwan, vol.3, pp. 1838-1843 1996.

[2] Le T. T., Tran S. T., Mita S., Nguyen T. DReal Time Traffic Sign Detection Using Color and Shape-Based Features. In: Nguyen N.T., Le M.T., Świątek J. (eds) Intelligent. 2010

[3] Information and Database Systems. ACIIDS. Lecture Notes in Computer Science, vol 5991. Springer, Berlin, Heidelberg, 2010

[4] Hsu, S-H., and C-L. Huang. "Road sign detection and recognition using matching pursuit method." Image and Vision Computing Vol. 19, pp. 119-129. 2001.

[5] Zhu, Yingying & Chengquan, Zhang & Zhou, Duoyou & Wang, Xinggang & Bai, Xiang & Liu, Wenyu. Traffic Sign Detection and Recognition using Fully Convolutional Network Guided Proposals. Neurocomputing, 214. 2016.

[6] J. F. Khan, S. M. A. Bhuiyan and R. R. Adhami, "Image Segmentation and Shape Analysis for Road-Sign Detection," in IEEE Transactions on Intelligent Transportation Systems, vol. 12, no. 1, pp. 83-96, 2011.

[7] M. Mathias, R. Timofte, R. Benenson and L. Van Gool, "Traffic sign recognition —How far are we from the solution?," The 2013 International Joint Conference on Neural Networks (IJCNN), Dallas, TX, pp. 1-8, 2013.

[8] Souani, Chokri, Hassene Faiedh, and Kamel Besbes. "Efficient algorithm for automatic road sign recognition and its hardware implementation." Journal of real-time image processing Vol. 9, no. 1, pp. 79-93. 2014.

[9] Ayaou, Tarik, et al. "Enhancing road signs detection rate using Multi-Scale Retinex." 2012 International Conference on Multimedia Computing and Systems. IEEE. 2012.

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

[11] J. Stallkamp, M. Schlipsing, J. Salmen and C. Igel, (2011), "The German Traffic Sign Recognition Benchmark: A multi-class classification competition," The 2011 International Joint Conference on Neural Networks, San Jose, CA, pp. 1453-1460.