Traffic sign recognition and detection using SVM and CNN

Sai Jayanth Gollapudi, Harshith Singathala, Harsha Kata

[1]Sai Jayanth Gollapudi, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore – 632014, Tamil Nadu, India
[2]Harshith Singathala, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore – 632014, Tamil Nadu, India
[3]Harsha Kata, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore – 632014, Tamil Nadu, India

---***---

Abstract - *For adding the safety of the drivers, pedestrians and vehicles as well, to the driver easement systems, traffic sign recognition feature is required. For developing TSR systems, we need the use of CV (Computer Vision) techniques, which could be viewed as principal in the field of pattern recognition all in all. We are going to use two latest architectures called Lenet-5 model and VGGNet model architectures in two different approaches. In this project, we are going to present the study of two major approaches which are required for developing traffic sign detection and recognition systems. We propose a methodology for traffic sign identification dependent on Convolutional Neural Networks (CNN). First, we are going to transform the original image into greyscale image with the help of SVM(support vector machine) and then use CNN(convolutional neural network) for detecting and recognizing things with fixed and learnable layers we use CNN(convolutional neural network). With fixed layers, we can lessen the measure of interest zones to identify, and trim the limits near the boundaries of traffic signs. The accuracy of detection can be increased with the help of learnable layers. By researching and study of many research papers, we want to give a real-time solution for this challenging problem called TSR (Traffic Sign detection and Recognition).*

Key Words: CNN, Driving Assistance, Neural Networks, Q Learning Reinforcement Learning, RNN.

1.INTRODUCTION

Researchers are trying to develop advanced driver easement systems and by it's name, requires more assistance features in it. One of the features in it is traffic sign detection and recognition. This feature helps in detecting and recognizing different traffic signs and alert the driver as a warning signal which helps in adding safety of the drivers, pedestrians and vehicles as well. The main aim of our project is developing TSR(Traffic sign detection and recognition) by making it to detect different traffic signs and classify them from the live images captured by a sensor(Ex: Camera).

2. LITERATURE REVIEW

[1] Detecting traffic signs has become a vital point in artificial intelligence, computer vision and deep learning with applications, taking everything into account, for example robot navigation and safe driving. In this paper, they proposed a framework with two deep learning fragments that includes (FCN) Fully convolutional association guided traffic sign suggestions and deep (CNN) Convolutional neural association for object request. Their thinking is to use CNN to arrange traffic sign suggestions to perform fast and exact traffic sign detection and affirmation. To improve the identification, they are utilizing edge box strategy by utilizing prepared FCN. They utilized shading division, colour segmentation, shape location and sliding window examining to discover traffic regions. They additionally have utilized a FCN guided item strategy. They utilized this calculation on the Swedish Traffic signs Dataset. They accomplished a generally excellent exactness on this Swedish signs dataset (98.67). In future, the creators are arranging an end to end network to produce the proposed FCN guided recommendations and growing ongoing traffic sign framework dependent on the calculations utilized in this paper.

[2] There are very good results achieved in traffic sign detection. In this paper, they are detecting and classifying the traffic signs dataset by using multi scale CNN algorithm. They also used selective search edge box techniques and multi scale combinatorial grouping (MCG). In this paper, they prepared/trained two networks on this benchmark: one treats every sign class as a solitary classification and can be viewed as a traffic sign detector and the other network can all the while recognize and classify traffic signs. The two algorithms beat past works. They utilized a dataset in which they took 10 areas from 5 unique urban communities in China, d 100000 scenes from the Tencent Data Center. It gives 100000 pictures containing 30000 traffic-sign occurrences. They achieved 84percent accuracy tested on the 90000 panoramas that contained no traffic signs, and the network perfectly identified them all. In future, the authors are planning to seek out more traffic signs of the classes that rarely appear in present work and they are also planning to accelerate speed of process in order to run it on mobile phones etc.. in real life.

[3] In this paper, they are presenting a new method to detect and recognize the traffic signs. This is based on 3 steps. First step is image segmentation using thresholding of HIS colour space components and extracts ROIs. The subsequent

step recognizes traffic signs by handling the blobs from the ROI. The last step perceives the data included in the identified traffic signs. They utilized blend of (HOG) histogram of oriented gradients processed from the HSI shading/color space with LSS features to frame new descriptor. They utilized arbitrary forest classifier to perform acknowledgment. They performed this on German traffic sign dataset and Swedish traffic sign dataset. They achieved 94.21percent AUC on the GTSDB data set and 92.11percent in STS data set. In the future work, they are planning to use adaptive thresholding to overcome the colour segmentation problems.

[4] Despite the fact that traffic sign acknowledgment/recognition has been read for a long time, most existing works are centered around the image-based traffic signs. This paper bargains about acknowledgment of both image-based and text based signs. The framework comprises of three phases, (ROIs) traffic sign regions of interest extraction, ROIs refinement and grouping, and post-processing. Traffic sign ROIs from each casing are first extricated utilizing maximally stable extremal areas on grey and standardized RGB channels. At that point, they are refined and allocated to their definite classes through the proposed multi tasks Convolutional neural networks, which is prepared with a huge measure of information, including manufactured traffic signs and pictures marked from road sees. The post handling finally consolidates the outcomes in all frames to settle on an recognition choice. Here they used German traffic sign detection dataset. Their model gets the great result on a challenging new data set also. They achieved 87percent recognition rate. Improving the speed of system is also included in future work.

[5] In this paper, Yingying Zhu et al are proposing another framework for traffic sign location utilizing two deep learning segments. They applied a fully Convolutional organization to section candidate traffic sign regions showing applicant regions of interest (RoI), trailed by a quick neural network to recognize messages on the extricated RoI. The proposed strategy utilizes the attributes of traffic signs to improve the proficiency and exactness of text recognition. The proposed two-stage detection technique lessens the pursuit space of text identification and eliminates messages outside traffic signs and FCN is utilized to accomplish this. It solves the problem of multi-scales for the text detection part to a large extent. They used Text-based Traffic Sign Dataset in Chinese and English (TTSDCE) and the Traffic guide panel dataset (approximately 3900 images). The experimental results show that the proposed method is not only efficient and effective but also can be easily applied to text-based traffic signs in other languages. In future, they are planning to improve the accuracy by using information in the videos with text traffic signs.

[6] Road signs from one country to another may look very different, which makes it difficult for the classification system to work successfully. The training data set for the proposed model includes road signs from six European countries: Belgium, Croatia, France, Germany, the Netherlands, and Sweden. The classes belong to 4 main categories and subcategories: Danger/warning, regulatory, informative, others. A comparative study of 5 CNNs architectures trained with our proposed European dataset and the German Traffic Sign Recognition Benchmark (GTSRB). They described the 5 CNNs that achieve the best performances in the state of the art regarding Traffic Sign Classification as: Le-Net 5, IDISA model, URV model, CNN with asymmetric kernels, CNN 8 layered. Their proposed European traffic sign dataset proved to be more robust than the GTSRB dataset with the 5 CNN architectures trained on, making it reliable and more complete for traffic sign recognition. Their future work includes intent to take into account the class imbalance problem to improve recognition accuracy.

[7] Lately, the consequence of traffic sign recognition (TSR) has been maintained, and TSR is additionally advancing quickly in deep learning. The TSR consists of essentially a couple of ways namely, traffic sign classification (TSC) and traffic sign detection (TSD). In this paper, they introduce a new efficient TSC network called Ent (efficient network) and a TSD network called EmdNet, which can achieve an accuracy of 98.6percent on the GTSRB. In this paper, An innovative network construction method is proposed for both TSC and TSD net- works. Their future work includes: improved performance of TSD network, inclusion of video instead of images for input to networks and research multitask learning and improve the generalization ability and commercialization of the network.

[8] This paper expresses the real-world application of intelligent deep learning techniques in TSR (traffic sign recognition). These include application in intelligent transportation surveillance and analysis. Difficulty during the deployment of deep neural networks toward embedded traffic sign recognition comprises huge computational and memory demands concerning such networks. To approach this prob- lem they have performed MicronNet, a deeply dense deep convolutional neural network for real-time embedded traffic sign recognition designed based on macro-architecture de- sign principles. Their algorithm includes Numerical Micro- architecture Optimization, Spectral Macro architecture Aug- mentation, Parameter Precision Optimization and Activation Function Selection and Training. Later citing its correctness they examined it with other state-of-the-art traffic sign recognition networks: STDNN, HLSGD, MCDNN, CDNN; Data set used for testing is The German traffic sign recognition benchmark (GTSRB). The resulting MicronNet network pro- duces a good balance amid accuracy and model size as well as inference speed. Their future work includes: exploring additions upon MicronNet over a more comprehensive range of traffic datasets to enhance generalizability in diverse situations.

3. METHODOLOGY

Architectures like multi-scale training etc, are very useful for image detection. By training NN (Neural network), they can recognize patterns which can certain colors. For reducing the color resolution of image, color segmentation neural networks are very useful. By this we can say that neural networks are very powerful in classify things. We are using LeNet and VGGNet to recognize traffic signs' features within a region of interest by training them.

There are a total of 7 modules in our proposed Architecture. In this we discuss the results at every module.

The modules are:

A. LOAD THE DATA

Initially we download the dataset from the Kaggle web-site which consists of more than 50,000 images. Link: http://benchmark.ini.rub.de/?section=gtsrb&subsection=dataset

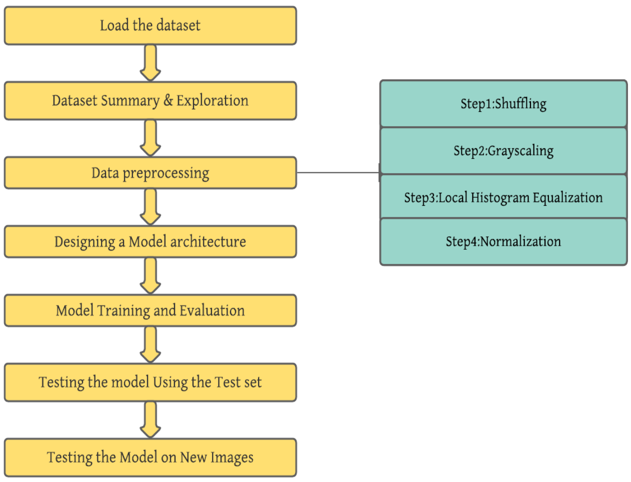

Fig -1: Architecture Diagram

We load all the images from the dataset which are resized to 32x32 and we do pickling for every image using python pickle module which results all the images as a matrix with each image[i,j] represents as pixel data of the image

.we divided the dataset into 3 categories. They are training dataset, validation dataset, and testing dataset.

B. ABOUT THE DATASET AND EXPLORATION OF THESE PICKLED FILES GENERATED BY US

As we divided the dataset into three categories so we have the final three pickled files they are

1. Train.p
2. Test.p
3. Valid.p

C. DATA PREPROCESSING

As we discussed in methodology, we used four preprocessing techniques before training the dataset. We will discuss the results for each preprocessing technique here.

1) Shuffling:

Its aim is to shuffle data to avoid element bias and it helps in increasing the predictive performance and improving model quality. We use sklearn for this technique.

2) Gray scaling:

Its aim is to convert the images in dataset to gray scale images which helps in increasing the accuracy of ConvNet. We use OpenCV for this technique.

3) Local Histogram Equalization:

Its aim is to enhance the contrast of the image. After applying this technique, the results are the images enhancing with low contrast. We use skimage for this technique.

4) Normalization:

Its aim is to normalize the image data so that the data has zero mean and equal variance. It rescale the range of pixel intensity values to 0-1 range.

D. DESIGNING MODEL ARCHITECTURE

In this module, we are going to create and implement a traffic sign recognizing deep learning model and train it for detecting traffic signs by using the data from our dataset. For classifying the images in dataset, we use CNN (Convolutional Neural Networks). With minimal preprocessing, the model can recognize the visual patterns from images by using ConvNet and we also use grayscaling for increasing its accuracy. On validation set, we are aiming for an accuracy of atleast 96%. Then we work on this module using tensorflow. We will utilize 0.001 learning rate, which advises the organization how rapidly to refresh the loads. We are going to use the Adam (Adaptive Moment Estimation) Algorithm as it is an enchancement. Adam calculation processes versatile learning rates for every boundary. As well as putting away an exponentially decaying normal of past squared slopes like Adadelta and RMSprop algorithms, Adam additionally keeps an exponentially decaying normal of past angles mtmt, like energy calculation, which thusly produce better outcomes.

At every level, model learn hieracrches of features that are invariant from data automatically. We will implement LeNet-5 and VGGNet. We are aiming for an accuracy of above 96% on validation set. Then we will work on this architecture using tensor flow library in python.

The latest two models used are:

1) LeNet-5:

This convolutional network is used for recognizing charac- ters and for OCR in given data. In real-time using MNIST dataset, banks recognize the characters by detecting from the handwritten cheque by using LeNet CNN.

2) VGGNet:

VGGNet is a deep CNN, meaning that it has a large number of layers. The original VGGNet has 16 layers, while a later version called VGGNet-19 has 19 layers. The layers in a CNN are arranged in a hierarchical fashion, with each layer extracting increasingly complex features from the input image.

E. TRAINING AND EVALUATING MODEL

In this module, we use normalized images (which are done by normalization preprocessing technique) for training. At test time, it is not difficult to surmised the impact of averaging the expectations of all these diminished networks by basically utilizing a solitary unthinned network that has more modest weights. These altogether can help to reduce over-fitting problems and gives great upgrades compared to other strategies that are used for regualrization.

F. TESTING THE MODEL WITH TESTING DATASET

In this part, by using random unknown examples, we are going to measure the performance and accuracy by using the testing model. And we are plotting confusion matrix. By plotting it, we can figure out whether the model is failed or succeeded by testing it on random unknown test samples. And for further improvement, we use hierarchal CNN's.

G. TESTING THE MODEL ON NEW IMAGES

In this module, we are going to test our model after all possible improvements, by using it for predicting 5 random traffic sign images from dataset

4. PROPOSED MODEL

4.1. LeNet

This convolutional network is used for recognizing charac- ters and for OCR in given data. In real-time using MNIST dataset, banks recognize the characters by detecting from the handwritten cheque by using LeNet CNN. Al- beit this ConvNet is planned to characterize manually written digits, we're certain It has a high precision when managing traffic signs, given that both manually written digits and traffic signs are given to the PC as pixel pictures.

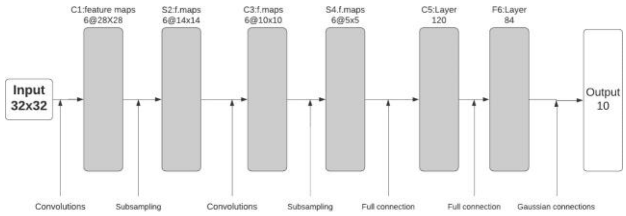

Fig -2: LeNet Architecture Diagram

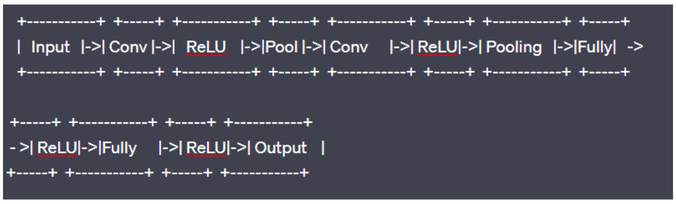

Chart-1: Steps undergoing in ConvNet process

In fig 3, "Conv" references to Convolution process, "Fully" references to FullyConnected process.

4.2. VGGNet

VGG Net Convolutional network depth is toward its effi- ciency in a significant large-scale image recognition envi- ronment. The main contribution is a thorough evaluation of networks of increasing depth using an architecture with very small (3x3) convolution filters, which proves that a notable development on the prior-art arrangements can be accomplished by pushing the depth to 16-19 weight layers.

1) VGGNet architecture:

The original VGGNet architecture will be having 16-19 lay- ers, but we have excluded some of them to reduce complexity and implemented a modified version of only 12 layers to save computational resources.

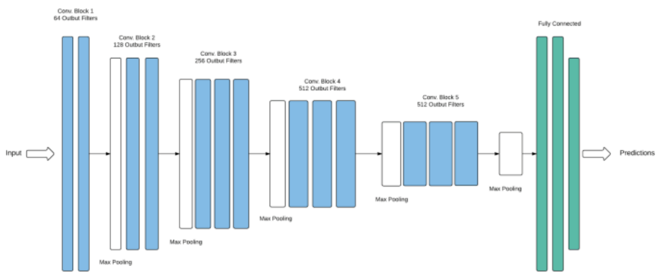

Fig -3: VGGNet Architecture Diagram

This ConvNet follows these steps:

1)Input[Input Image]

2)Convolution[Apply filter to image]

3)ReLU[Apply ReLU activation function]

4)Convolution[Apply filter to feature maps]

5)ReLU[Apply ReLU activation function]

6)Pooling[Reduce size of feature maps]

7) Convolution[Apply filter to feature maps]

8)ReLU[Apply ReLU activation function]

9)Convolution[Apply filter to feature maps]

10)ReLU[Apply ReLU activation function]

11)Pooling[Reduce size of feature maps]

12)Convolution[Apply filter to feature maps]

13)ReLU[Apply ReLU activation function]

14)Convolution[Apply filter to feature maps]

15)ReLU[Apply ReLU activation function]

16)Pooling[Reduce size of feature maps]

17) FullyConnected[Connect all neurons in previous layer]

18) ReLU[Apply ReLU activation function]

19) FullyConnected[Connect all neurons in previous layer]

20) ReLU[Apply ReLU activation function]

21) FullyConnected[Classify image] end

5. RESULTS AND DISCUSSIONS

There are a total of 7 modules in our proposed Architecture. In this we discuss the results at every module

A. LOAD THE DATA

Initially we download the dataset from the Kaggle website which consists of more than 50,000 images.

Link: We load all the images from the dataset which are resized to 32x32 and we do pickling for every image using python pickle module which results all the images as a matrix with each image[i,j] represents as pixel data of the image.

we divided the dataset into 3 categories. They are training dataset, validation dataset, and testing dataset.

Graphical Representation of dataset for all the three categories using histogram plot:

Total No. of training examples: 34799 Total No. of testing examples: 12630 Total No. of validation examples: 4410 Total No. of Image Classes: 43
Image size: (32x32x3)

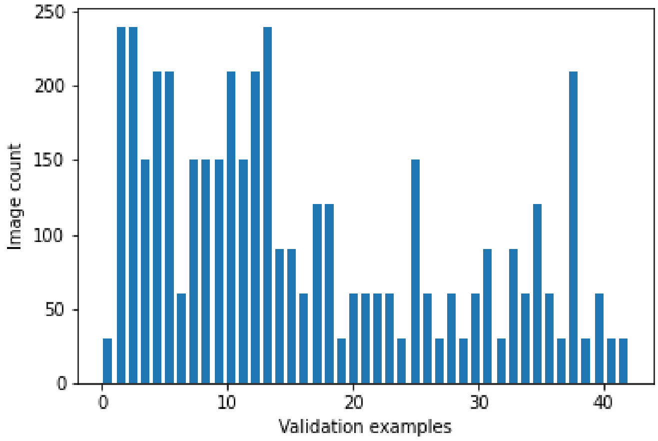

Chart -2: Image count in each folder in training examples

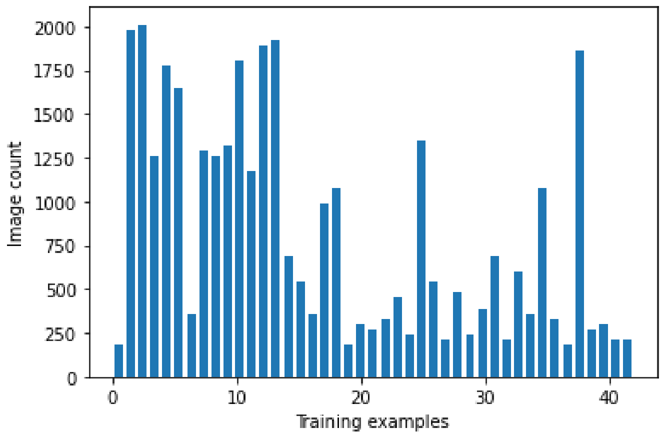

Chart-3: Image count in each folder in testing examples

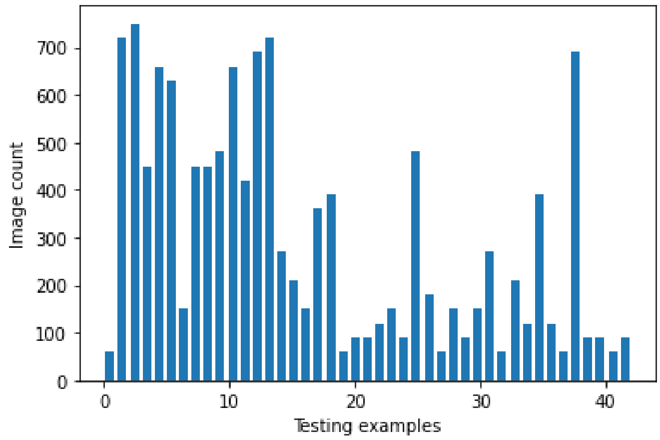

Chart-4: Image count in each folder in validation examples

As we divided the dataset into three categories so we have the final three pickled files they are

- Train.p
- Test.p
- Valid.p

Examples of our dataset:

Fig -4: Images from our dataset

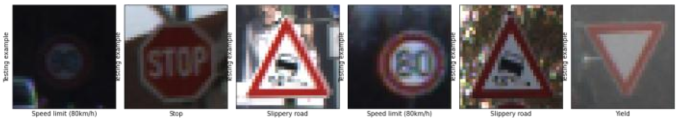

Fig -5: Images from our dataset

Fig -6: Images from our dataset

B. ABOUT THE DATASET AND EXPLORATION OF THESE PICKLED FILES GENERATED BY US

Here we have the pickled data which the dictionary containing key value pairs. They are
'Features': It is the first key. It is a 4X4 matrix containing the raw pixel data at each cell of the traffic sign image from the dataset

'labels': It is a vector having assigned class ids/labels of traffic sign images for later processing. There is signnames.csv file this file contains the id and it maps to the desired the traffic sign image.

'sizes': It is a list containing tuples with each tuple as the (width ,height) of every image from the dataset.
'coords : It is also a list containing tuples of which has (x1,y1,x2,y2) representing coordinates of a bounding field across the signal withinside the image.

C. DATA PREPROCESSING

As we discussed in methodology we used four preprocessing techniques before training the dataset. We will discuss the results for each preprocessing technique here .

1) Shuffling

From the sklearn we have libraries to shuffle the data set. Here is the sample code shown below for shuffling. To increase the randomness

2) Gray scaling
Using OpenCV we can grayscale the images

Output after grayscaling :

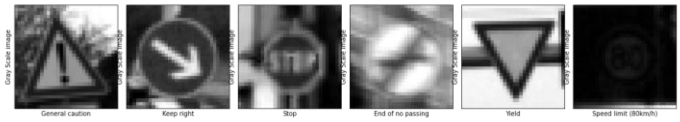

Fig -7: Images after Grayscaling

3) Local Histogram Equalization

The next step is histogram equalization of every image. The output after this step

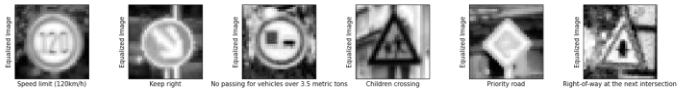

Fig -8: Images after Histogram Equalization Process

4) Normalization

The final step of preprocessing is Normalization. The output after this step

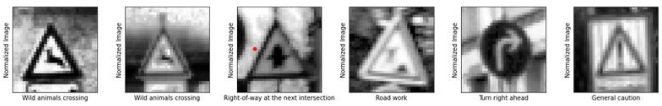

Fig -9: Normalized images

D. DESIGNING MODEL ARCHITECTURE (I E CODING THE CNN MODELS FOR TRAINING

In this module we write the code for our Models and run them. They are After coding the LeNet -5 Model. We also used another model called VGGNET for comparison of both to see which predicts outputs correctly with maximum efficiency.

After coding and executing theses model class now our data set is ready for training.

6. MODEL TRAINING AND EVALUATION

In the module we do training and evaluation.

In this we are going to train our models from the obtained normalized images from the module 2. Now we are going to train our model using a pipeline and run the training data.

As discussed earlier we have a set our epoch to 30. after completion of each epoch we shuffle the training set. After training of each epoch we calculate the loss and accuracy of validation set. After completion of 30 epochs we save our model. From this we can state that if we get low accuracy on the validation and training set we can clearly state that underfitting and vice versa (i.e.) high accuracy states that overfitting.

After LeNet-5 training output using sample code for LeNet-5 training:

From the above training of the LeNet-5 model we are achieving the maximum accuracy of 94.5 % over the validation set on running over 30 epochs .Now we well train the VGGNet model and evaluate its maximum accuracy.

After VGGNet training using VGGNet sample code:

From this Model we are able to achieve the maximum accuracy of 99.3%. So, we are going to use this model for predicting testing dataset.

Pipeline architecture for VGGNet :

Figure13 is the pipeline architecture of the VGGNet running 30 epochs with validation accuracy as 99.3 % and testing accuracy as 97.6 %

7. TESTING THE MODEL WITH TESTING DATASET

Now we use the training set which was divided in the module 1 we use that to predict the accuracy of our model. We are been able to achieve the test accuracy around 97.6%. which means the signs are almost predicted correctly.

As From the testing accuracy we noticed that our model fails in some cases we will represent the confusion matrix for which images our model is failing. Confusion matrix graph is shown in figure 12.

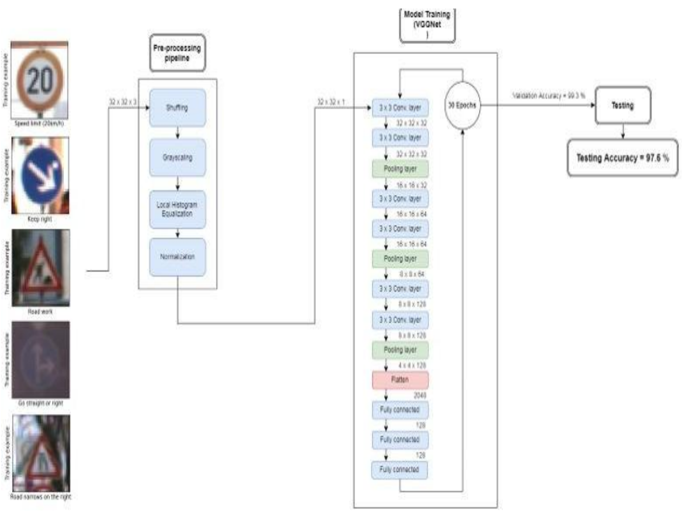

Fig -10: Pipeline Architecture Diagram for VGGNet

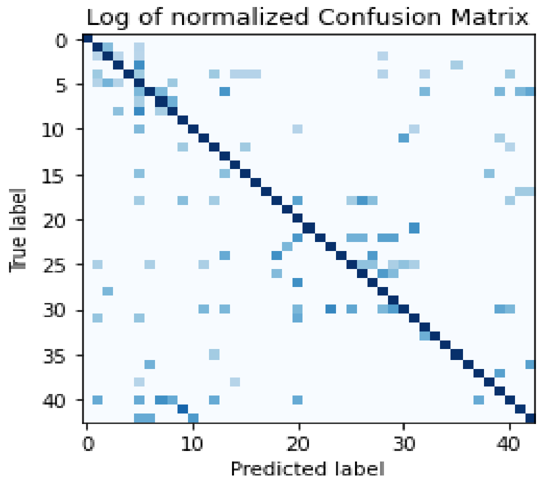

Chart-5: Confusion Matrix Graph

From the above graph we have a look at a few clusters withinside the confusion matrix above. It seems that the speed limits traffic sign images are every now and then are mis-labelled. Similarly, The Traffic signals with triangular form are misclassified amongst themselves. We can similarly enhance at the version using hierarchical CNNs to first become aware of broader groups (like speed limit images) after which have CNNs to classify finer features (including the real speed limit).

8. TESTING THE MODEL ON NEW IMAGES

We have Final module in our project testing the model using 5 random set of images.

Plotting new images:

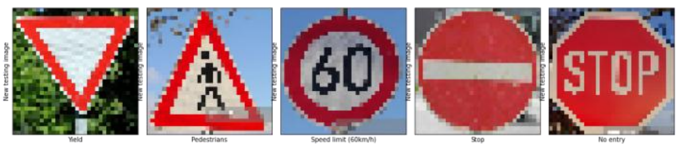

Fig -11: 5 random set of images

In these some of them are easy to predict like stop and No entry. Also this set contains some low clarity or blurred image like speed limit sign .Finally the signs in triangular shape "Pedestrians" sign , are also typical to predict because traffic signs with triangular shape are misclassified among themselves.

After that data preprocessing: We wrote the all data preprocessing functions differently that as 4 different functions. And we called all the functions in the same function called preprocess.

This line of code will preprocess the data images and gives the output as normalized images. We pass this images to the model for predicting the label.

Output of Predicting Model code:

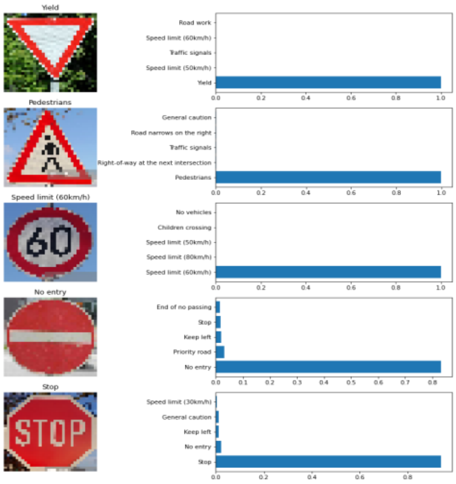

Fig -12: Predicting images

As we will note from the top five softmax probabilities, the model which we have created has very good accuracy(100%) while we input the simple test image traffic signs, like the "Stop" and the "No entry" signal, and even excessive predic- tion accuracy while predicting easy triangular symbol signs in a completely clean image, like the "Yield" signal.

Also, we have to notice that our model accuracy is slightly low while detecting the more complicated triangular signal in a "quite noisy" image, in the "Pedestrian" signal image, we have got a triangular signal with a form internal it and the copyrights of the photographs provides a few noises to the image, the trained model became able to expect the accurate class, however with 100% self-assurance. But it may be sometimes less when the image is very blur and noisier it may expect the actual class may be between (60,100).

And in the "Speed limit" signal, we will look at that the version appropriately expected that it a

"Speed limit" signal, however, became one way or the other burdened among the specific pace limits. However, it was able to assign the image to the correct id/label at the end. The VGGNet model version became capable of expecting the accuracy results for every of the five new take a look atimages.

Test Accuracy = 100.0%

After Pickling file using picking code, it results in three files as

- train.p
- valid.p
- test.p

These files are uploaded in aws bucket and got the cdns(i.e cloud front urls) to use them in google colab.the google colab allows us to download it that particular notebook each time we run them.

9. CONCLUSIONS

We have discussed how the deep learning can be utilized to order traffic signs with high precision, utilizing an assortment of pre-processing and regularization methods (for example dropout), and attempting distinctive model designs. We have fabricated profoundly configurable code and fostered an adaptable method of assessing numerous architecture. The model we designed arrived at near 97% precision on the test set, accomplishing 98% on the validation set.

REFERENCES

[1] Bangquan, X., Xiong, W. X. (2019). Real-time embedded traffic sign recognition using an efficient convolutional neural network. IEEE Access, 7, 53330-53346.

[2] Wong, A., Shafiee, M. J., Jules, M. S. (2018). Micronet: A highly compact deep convolutional neural network architecture for real-time embedded traffic sign classification. IEEE Access, 6, 59803-59810.

[3] Zhu, Z., Lu, J., Martin, R. R., Hu, S. (2017). An optimization approach for localization refinement of candidate traffic signs. IEEE Transactions on Intelligent Transportation Systems, 18(11), 3006-3016.

[4] Liu, C., Li, S., Chang, F., Wang, Y. (2019). Machine vision based traffic sign detection methods: Review, analyses and perspectives. IEEE Access, 7, 86578-86596.

[5] Liu, Z., Du, J., Tian, F., Wen, J. (2019). MR-CNN: A multi-scale region- based convolutional neural network for small traffic sign recognition. IEEE Access, 7, 57120-57128.

[6] Changzhen, X., Cong, W., Weixin, M., Yanmei, S. (2016, August). A traffic sign detection algorithm based on deep convolutional neural network. In 2016 IEEE International Conference on Signal and Image Processing (ICSIP) (pp. 676-679). IEEE.

[7] C. Wang, "Research and Application of Traffic Sign Detection and Recog- nition Based on Deep Learning," 2018 International Conference on Robots Intelligent System (ICRIS), Changsha, 2018, pp. 150-152.

[8] Ravindran, R., Santora, M. J., Faied, M., Fanaei, M. (2019, December). Traffic Sign Identification Using Deep Learning. In 2019 International Conference on Computational Science and Computational Intelligence (CSCI) (pp. 318-323). IEEE.

[9] Han, Y., Oruklu, E. (2017, August). Traffic sign recognition based on the nvidia jetson tx1 embedded

system using convolutional neural networks. In 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS) (pp. 184-187). IEEE.

[10] Stallkamp, J., Schlipsing, M., Salmen, J., Igel, C. (2011, July). The German traffic sign recognition benchmark: a multi-class classification competition. In The 2011 international joint conference on neural networks (pp. 1453-1460). IEEE.

[11] Houben, S., Stallkamp, J., Salmen, J., Schlipsing, M., Igel, C. (2013, Au- gust). Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark. In The 2013 international joint conference on neural networks (IJCNN) (pp. 1-8). Ieee.

[12] Yang, Y., Luo, H., Xu, H., Wu, F. (2015). Towards real-time traffic sign de- tection and classification. IEEE Transactions on Intelligent transportation systems, 17(7), 2022-2031.

[13] Li, K., Lan, W. (2011, August). Traffic indication symbols recognition with shape context. In 2011 6th International Conference on Computer Science Education (ICCSE) (pp. 852-855). IEEE.

[14] Sugiharto, A., Harjoko, A. (2016, October). Traffic sign detection based on HOG and PHOG using binary SVM and k-NN. In 2016 3rd Interna- tional Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE) (pp. 317-321). IEEE..

[15] Kamal, U., Tonmoy, T. I., Das, S., Hasan, M. K. (2019). Automatic traffic sign detection and recognition using SegU-net and a modified tversky loss function with L1-constraint. IEEE Transactions on Intelligent Transportation Systems, 21(4), 1467-1479.

[16] Nunn, C., Kummert, A., Muller-Schneiders, S. (2008, June). A novel region of interest selection approach for traffic sign recognition based on 3d modelling. In 2008 IEEE Intelligent Vehicles Symposium (pp. 654- 659). IEEE.

[17] Luo, H., Yang, Y., Tong, B., Wu, F., Fan, B. (2017). Traffic sign recog- nition using a multi-task convolutional neural network. IEEE Transactions on Intelligent Transportation Systems, 19(4), 1100-1111.

[18] Greenhalgh, J., Mirmehdi, M. (2012). Real-time detection and recognition of road traffic signs. IEEE Transactions on Intelligent Transportation Systems, 13(4), 1498-1506.

[19] Cengil, E., Cinar, A. (2018, September). A Deep Learning Based Ap- proach to Lung Cancer Identification. In 2018 International Conference on Artificial Intelligence and Data Processing (IDAP) (pp. 1-5). IEEE.

[20] Chou, L. D., Tseng, C. W., Lai, M. S., Chen, W. Y., Chen, K. C., Yen, C. K., ... Chiu, Y. H. (2018, October). Classification of malicious traffic using tensorflow machine learning. In 2018 International Conference on Information and Communication Technology Convergence (ICTC) (pp. 186-190). IEEE.

[21] Choi, S. Y., Jeong, H. J., Park, K. S., Ha, Y. G. (2019, February). Efficient driving scene image creation using deep neural network. In 2019 IEEE international conference on big data and smart computing (BigComp) (pp. 1-4). IEEE.

[22] Paclík, P., Novovicová, J., Duin, R. P. (2011). Building road-sign classi- fiers using a trainable similarity measure. IEEE Transactions on Intelligent Transportation Systems, 7(3), 309-321.

[23] Supreeth, H. S. G., Patil, C. M. (2016, March). An approach towards efficient detection and recognition of traffic signs in videos using neural networks. In 2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET) (pp. 456-459). IEEE.

[24] Ohara, H., Nishikawa, I., Miki, S., Yabuki, N. (2012, November). Detec- tion and recognition of road signs using simple layered neural networks. In Proceedings of the 9th International Conference on Neural Information Processing, 2002. ICONIP'02. (Vol. 2, pp. 626-630). IEEE.

[25] Biswas, R., Fleyeh, H., Mostakim, M. (2014, January). Detection and classification of speed limit traffic signs. In 2014 World Congress on Computer Applications and Information Systems (WCCAIS) (pp. 1-6). IEEE.