International Islamic University Chittagong

BACHELOR OF SCIENCE

IN

ELECTRONIC AND TELECOMMUNICATION ENGINEERING

Traffic Sign Detection and Recognition Based On Convolution Neural Network

Submitted by

Md Mahatab Uddin

T181012

Supervised by

Md Ibrahim

Assistant Professor

Department of ETE, IIUC

Department of Electronic and Telecommunications Engineering

International Islamic University Chittagong

Kumira, Sitakunda, Chattogram – 4318

November, 2022

CANDIDATE DECLARATION

It is hereby declared that the work presented in this thesis has not been submitted elsewhere for the award of any degree or diploma, does not contain any unlawful statement.

_____

Md Mahatab Uddin

T-181012

DEDICATION

This thesis work is dedicated to all of our honorable teachers and parents. Their good wishes and assistance in attaining our goals.

Acknowledgements

In the name of Allah, the Most Forgiving, the Most Merciful. All thanks and glory belongs to Allah (SWT), who has provided us with a wealth of chances and showered us with His mercy and guidance throughout life. And may Allah grant His Prophet Muhammad (pbuh) peace and blessings, who is a source of wisdom and inspiration in our life. We express our sincere gratitude and indebtedness to our thesis supervisor of the Department of Electronic and Telecommunications Engineering, IIUC **Md Ibrahim**, for his initiative in this field of research, valuable guidance, encouragement throughout the time of research. We are also thankful to **Engr. Abdul Gafur**, Convener of our Thesis Committee for his dedication and sacrifice. We also want to sincerely thank all of our teachers for their dedication to us during the course of our academic careers. We also think back on how much our parents have helped us throughout our lives. Additionally, we thank our friends and well-wishers for helping us finish this thesis work, whether directly or indirectly.

Abstract

Modern automated driver assistance systems that display safety information heavily rely on Traffic Sign Recognition and Detection. It is a system that enables users to instantly identify traffic signs, usually in films but occasionally only in still images. Road accidents are caused by improper interpretation of traffic signs. Moreover hundreds of people could be killed if a driver misidentifies a traffic sign in hazardous conditions like heavy rain, cloudy weather, or sleepiness. As a result, the appropriate identification of traffic signs has become a required research issue. Convolutional neural networks were employed in this study to accurately detect and classify the traffic signs. Five Keras models of CNN have been proposed, and their output has been compared. Dealing with picture noise, such as ads, parked cars, pedestrians, and other moving things or background objects that make recognition considerably more challenging, is the key difficulty of this research. The investigation has been impacted not only by the objects but also by a number of environmental factors as light reflection, precipitation, fog, etc. We have assembled our own data-set in order to undertake this research. We wandered the streets of Chittagong and took images of the traffic signs because there is no benchmark data set for Bangladesh accessible. This model provides a 98% accuracy for **39,200** photos. Numerous studies have been conducted in this area, but ours stands out since it is based on data that we have independently acquired from Bangladesh.

Table of Contents

List of Abbreviations

ADAS - Advanced Driver Assistance System

ADES - Automatic Driver Evaluation System

ANN - Artificial Neural Net

AR - Augmented Reality

CNN - Convolitional Neural Network

CSV - Comma Separated Values

CV A - Computer Vision Applications

DSS - Driver Support System

EV S - Explaine Varience Score

F C - Fully Connected

GT SRB - German Traffic Sign Recognition Benchmark

HOG - Histogram of Oriented Gradient

KL - Karhunen-Ioeve

MAE - Mean Absolute Error

MLP - Multilayer Perceptron

MSE - Mean Squared Error

NN - Neural Network

P LSA - Probabilistic Latent Semantic Analysis

R – CNN - Regional Convolutional Neural Network

ResNet - Residual Network

ROI - Region of Interest

SURF - Speeded Up Robust Features

SV M - Support Vector Machine

T S - Traffic Sign

T SR - Traffic Sign Recognition

Chapter 1

Introduction:

The signs on the sides of the roadways that we frequently encounter are traffic signs or road signs. Both cars and pedestrians are intended to get advice from these signs. King Peter II of Portugal ordered the installation of the earliest traffic sign in Lisbon, Portugal, in 1668 to control traffic at the time. Since then, there has been an increase in traffic. Many nations have updated their traffic signs to use more straightforward visual formats. Recognizing these indicators gradually become necessary. The significance of TSR (traffic sign recognition) increased inevitably as technology advanced. TSR is a technology that allows for traffic control and guidance with the goal of enhancing road safety.

1.1 Inspiration

One of the most important elements of modern-day smooth traffic flow is the traffic sign. By communicating traffic conditions and encouraging drivers to follow the regulations of the road, it also promotes road safety. For the development of automated, intelligent driver support systems, intelligent vehicle technology, and road maintenance, TS recognition and detection are crucial. This sensing technology offers the potential to alert drivers to poor road conditions and lower traffic accidents. As a result, today's intelligent driving technologies come in a wide variety and need for the ideal detecting technique. Due to numerous weather factors, such as - wet or overcast weather, night environment, etc., detection of the TS could be challenging at times. This technology controls the real-time positioning and content of the road sign. This makes the task challenging and highly accurate.CNN is utilized to provide a higher and more precise detecting method. Self-driving cars have gained popularity in recent years, which is only made possible by CNN. An outstanding estimate of vehicle detection based on CNN is provided by a study by Kaijing Shi, Hong Bao, and Nan Ma . These strategies imply many ways to speed up training and testing while also boosting detection precision [24]. Such research studies serve as our thesis's main inspiration.

1.2 Objective

This thesis attempts to forecast a Convolutional Neural Network-based TS recognition system. The collection of real-time photographs of the TS from Bangladesh's perspective under varied weather circumstances is our primary goal. We created a data collection from the digital photos after gathering the images. We later used that data set to create training and pretraining models.

We also want to use a variety of pre-trained Convolutional Neural Network (CNN) models. We will compare several algorithms based on how well they forecast the appearance of traffic signs. Finally, we investigated the causes of performance differences amongst models when evaluated on the same dataset.

1.3 Thesis Outline

In-depth analysis of numerous diverse strategies and techniques related to traffic sign identification is offered in this work. In order to identify TS in real-time for our thesis, we developed a novel strategy and methodology. We provided a quick overview of the recognition of traffic signs in chapter 1. In chapter 2, some earlier research projects and their methods were covered, while in chapter 3, CNN and its feature extraction methodology were highlighted. In addition, in chapter 4 we covered the proposed model and its application. The result analysis was covered in Chapter 5. The research was completed, and a brief overview of the future was provided, in Chapter 6.

Chapter 2

Literature Review

TSR is now a crucial component of contemporary study since it lays the path for a safer transportation system. The presence of diverse environmental factors has always made it difficult to detect and correctly identify traffic signs. Numerous auto manufactures have primarily used this technology. Different approaches have been tried to accurately identify TS. Over the past ten years, TSR systems have drastically changed. A TSR system's primary objective is to extract the ROI that includes the TS. The HSV color model is utilized to extract color information from the photographs, according to research by Prashenjit Dhar and Zainal Abedin. So, utilizing ROI, the image's features may be retrieved. CNN then classifies the photographs once more.

Author [1] Traffic Sign Detection and Recognition Based on Convolutional Neural Network By, [Ying Sun, Pingshu Ge, Dequan Liu]. This article proposes a deep learning-based method for recognizing traffic signs that focuses mostly on circular traffic signs. This approach efficiently detects and recognizes traffic signs by utilizing picture preprocessing, traffic sign detection, recognition, and categorization.

Author [13] Traffic Signs Detection and Recognition System using Deep Learning By, [Pavly Salah Zaki, Marco Magdy William, Bolis Karam Soliman, Kerolos Gamal Alexsan, Maher Mansour, Magdy El-Moursy, Kerolos Khalil]. They presented a quick and efficient technique to find and categorize traffic indicators in this study. These are this paper's primary contributions:

• The F-RCNN Inception v2 model has been able to produce accurate, dependable, and quick results even in challenging real-world driving conditions by using transfer learning and a fully convolutional network.

• Tiny-YOLOv2 is a very quick model with a respectable level of accuracy, but YOLOv2 or YOLOv3 should be used in its place if a higher level of accuracy is required.

• If a powerful GPU is accessible, accuracy gains can be made by adding a considerable amount more training data and training the models for a longer period of time.

Author [24] Traffic sign recognition using convolutional neural networks By, [Kaoutar Sefrioui Boujemaa, Afaf Bouhoute, Karim Boubouh and Ismail Berrada]. The analysis of two successful and effective methods for identifying and detecting road signs was reported in this paper. The Fast R-CNN is so much faster than the C-CNN approach, and it is also invariant to changes in illumination, according to the experimental results obtained after testing both methods on the German Traffic Sign Detection & Recognition datasets. The C-CNN technique, however, is invariant to scale and viewing angle while being sluggish and weather-sensitive.

Author [15] Recognition of Traffic Sings by Artificial Neural Network By, [Dan Ghica, Si Wei Lu and Xiaobu Yuan]. In this paper, a validation subnetwork for artificial neural networks is proposed, which enhances the functionality of neural architectures. When provided with inputs that reflected improper patterns, the backpropagation network (BP) and the ARTl network both failed to function properly. The ARTl had better behavior, but adding new classes after nearly every erroneous pattern pre-sentation was a challenge to resolve. After the validation subnetwork was adopted, the network was trained, and the weights were established, allowing us to construct the actual network with a fixed set of weights. The system is quick enough to enable real-time processing of pictures taken by a camera mounted on a car.

Author [6] Traffic sign recognition based on deep learning By, [Yanzhao Zhu & Wei Qi Yan]. Using the dataset of our traffic signs, this study intends to test the precision and speed of TSR. In order to assess the performance of the YOLOv5 algorithm, the authors used the most recent version in the series. Additionally, they determine which model is best for the TSR between the SSD and YOLOv5 versions. They use a customized dataset of 2,182 traffic sign photos from eight classes in their experiment, which uses a customized set of traffic sign images.

Chapter 3

CNN and Inclusion of Features

3.1 Convolutional Neural Network (CNN)

Multilayer perceptrons are renormalized to form CNN. It is an expert in detecting images in 2D matrices. Our CNN method is used to a 2D pixel matrix that makes up these images. CNN uses the brain as its primary incentive to identify or categorize the image. Simple cells and complex cells make up the visual region of the human brain. In contrast to complex cells, which incorporate regional features from a limited spatial neighborhood, simple cells assist in feature discovery. Translationally invariant characteristics benefit from spatial pooling. Humans blend all of the various regional elements that they detect with their eyes to categorize the image. This is how a fresh image appears to us. To accurately categorize photos, CNN uses this method.

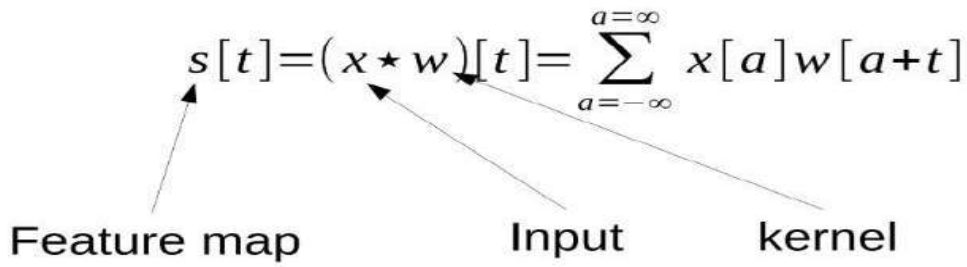


Figure 3.1: Convolution function

A convolution is used by a convolutional neural network. In mathematics, a function that is created by integrating two other functions is referred to as a convolution. It explains how one function can change another function's structure. For instance, if we consider an input I and an argument, the kernel K will generate an output that will convey how the shape of one is modified by the other. We discover picture "x". The image is made up of a 2D array of pixels with various RGB color channels. "W" stands for the kernel (feature detector) through which we obtain the output following the application of the feature map. To determine how similar two signals are, a feature map is utilized in mathematics. A feature detector or filter is used to detect the image's edges. The primary responsibility of identifying an image's edges is accomplished by the convolution procedure.

The convolution function is typically thought to always be zero. As we can see from the equation below, this leads to the implementation of infinite summation as a sum over a limited number of array members.

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n) \qquad (3.1)$$

The convolutional function for a 2D array may be seen in the equation, where I stands for the 2D array and K for Kernal. We can rewrite equation 3.1 using a cross correlation function to make the convolution implementation in machine learning more straightforward. Nearly every neural network makes use of a cross correlation function. Convolutions are inherently unstable, hence the equation 3.1 can be expressed as the equation 3.2.

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n) \qquad (3.2)$$

In equation 3.2, the cross correlation function is displayed. The range of possible values for m and n is utilized to stabilize the algorithm. As a result, variation is reduced.

3.1.1 How CNN Works

CNN is viewed as the mainstreamed form of completely interconnected networks. We already know that CNN draws its organizational inspiration from the brain. Each neuron in a layer is coupled to every neuron in the layer above it. A neuron, however, is fully independent inside a single layer and has no connections with any other neurons within that layer. As a result, the network becomes fully connected at every layer, which also makes CNN susceptible to data overfitting. The output layer is identified as the final FC (completely connected). Additionally, CNN uses a hierarchical structure to regularize data. It combines simpler patterns to create the complex patterns. Images are inputted into CNN for processing and classification. Depending on the image resolution, computers interpret images as arrays of pixels. Height, width, and dimension will be used to separate the resolutions.

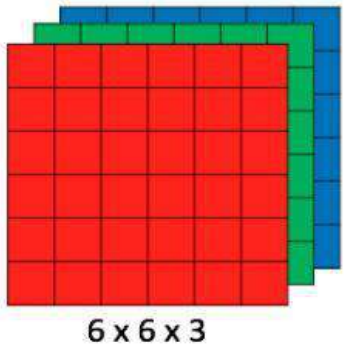


Figure 3.2: Array of RGB matrix

Figure 3.2 shows an illustration of an RGB h x h x d (h = height, w = width, and d = dimension) matrix array. The values in this case are 6 x 6 x 3. Now, for instance, the neurons will determine 224x224x3 weights for a picture of size 224x224. As the RGB color channel is defined by the dimension 3, it will stay at 3. The fundamental function of CNN is to run each input image through a set of convolutional layers with filters. In doing so, CNN is able to shrink the image without losing any information. Kernals are the layers that include filters. To train and test different CNN models, the images are fed through kernals.

3.1.2 An Architectural Overview of CNN

Multiple layers make up a convolutional network. To create a comprehensive convolutional architecture, the layers are stacked one on top of the other. This network's layers each use a differentiable function to rebuild the volume of activations from one layer to the next. The 3D input volume to the 3D output volume is converted by each layer using an easy-to-use API. This is done by utilizing the differential function. Convolutional layer, pooling layer, and FC layer are typically the three types of layers utilized to form convolutional architecture. The initial image is changed into the final class grade by adding these layers. There's a chance that some layers have parameters or not. The FC layer and the convolution layer typically have settings. These variables serve as the input's weights and biases in addition to being used to activate the input volume. The ReLu and Pooling layers, on the other hand, strive to implement a fixed function. in order to train the parameters from the earlier levels. The convolution layer, which is dependent on the labels of each image in the training set, uses the gradient descent technique to determine the class results.

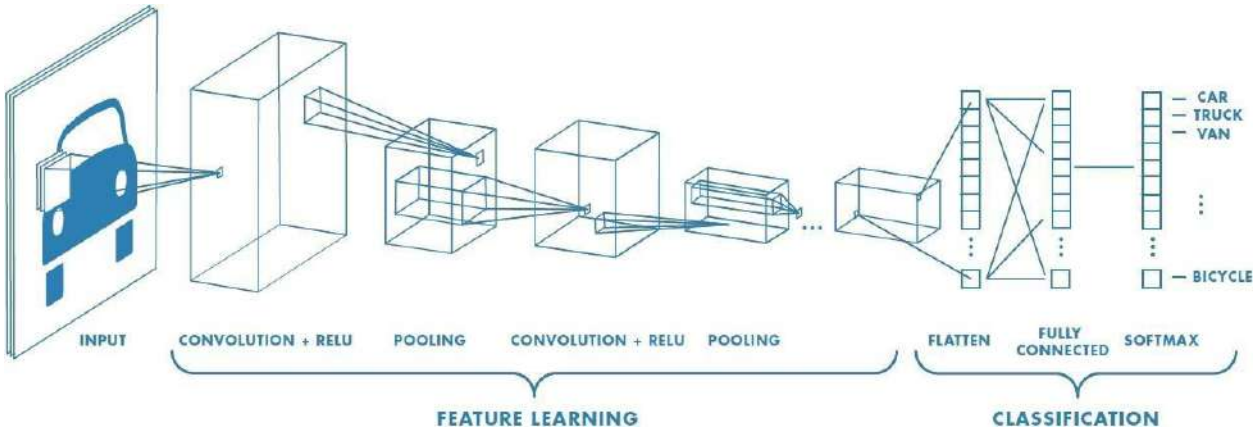


Figure 3.3: Different layers of CNN

We can see the many levels of CNN in excellent detail from figure 3.3. These layers, together with Kernels, Pooling, FC, and applying the Softmax function to classify an object, are applied to each input image. A probabilistic value between 0 and 1 is produced in this way. From processing an input image through classifying the images based on their values, CNN's entire workflow is seen in figure 3.3. Below are descriptions of CNN's many layers:

3.1.3 Input Layer:

The image is read by this layer. The input layer has nothing to learn as a result. It merely offers the input image's form. The input contains an image's 9 raw pixel values. For instance, the input for a 64x64 image would be 64x64x3. Here, there are three RGB color channels and 64 pixels of height and width. Therefore, this layer has no parameters to learn.

3.1.4 Convolutional Layer:

This layer uses CNN. Given that it handles the majority of the intricate computation, it serves as the foundation of a CNN. CNN learns in this layer. Therefore, we do have parameters that are weight matrices. A set of learnable filters are made up of these parameters. We must multiply the height, width, and take the filters into consideration in order to determine the parameters. The bias term for each filter is another important factor to take into account. Consequently, a convolutional layer would have the following number of parameters:

$$((m * n) + 1) * k) \qquad (3.3)$$

M is the width of the filter's form in equation 3.1. the number of filters, k, and the shape of the filter's height, n "k" feature maps are thought to be the output of a convolutional layer. Every single filter had the 1 added as the bias term.

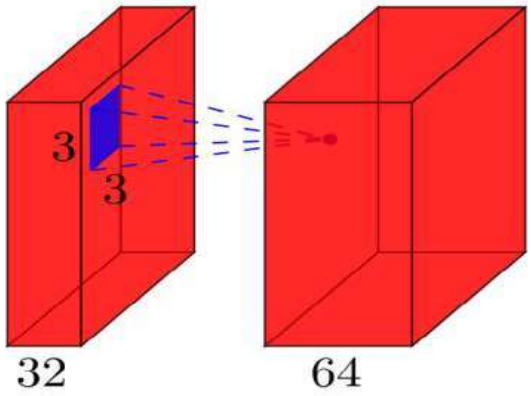


Figure 3.4: Input and output of a Conv layer

We can observe from figure 3.4 that the input contains I=32 feature maps. The filter size is n=3 and m=3, and the outputs are k=64 feature maps. It is essential to remember that the input has three dimensions, thus we must employ a three-by-three filter. Therefore, 64 separate 3*3*32 filters make up the first convolution layer's output.

3.1.5 Pooling Layer:

Because it only calculates a fixed function of the given input, no parameters are announced in this layer. This layer aids in reducing image dimension size in order to reduce the number of parameters. As a result, overfitting is avoided. A pooling layer is typically found between convolutional layers. A pooling layer is also translationally insensitive. This means that if only a small portion of the input is changed, the pooled output remains unchanged. This pooling layer feature aids in detecting common characteristics in the input image. For example, the edges or colors of an image.

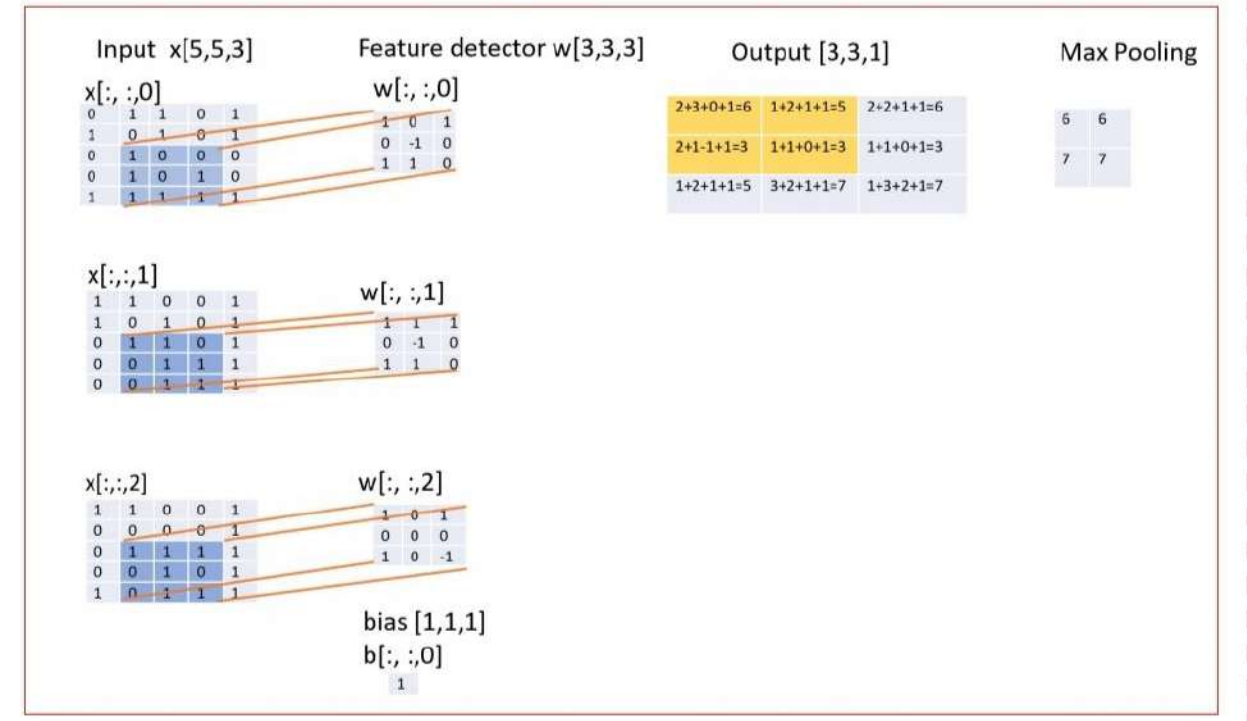


Figure 3.5: Input and output of a Conv layer

3.1.6 ReLu Layer:

A non-linear operation is the Rectified Linear Unit (ReLU). The term reLu stands for rectified linear unit. This concept is used in the convolutional network to introduce nonlinearity. Because real-world data necessitates nonnegative-linear values for the convolutional network, ReLU converts all negative numbers to zeros. Equation 3.2 depicts the output of this layer.

$$f(x) = max(0, x) \qquad (3.4)$$

The ReLU function takes any integer value 'x' and returns zero if the integer value is negative, as shown in equation 3.4. Other activation functions, such as tanh or sigmoid functions, can be used to add nonlinearity to the network in addition to the ReLU function. However, according to experiments and research, ReLU works best for CNN.

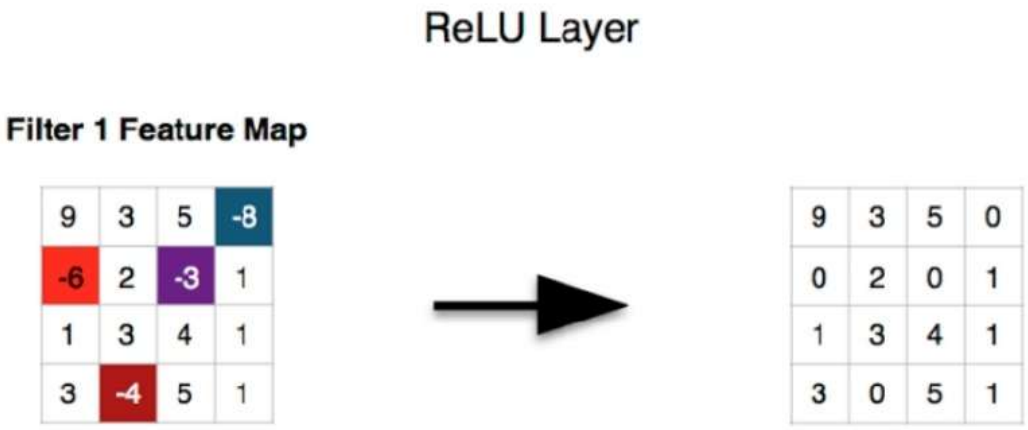


Figure 3.6: ReLU layer

3.1.7 Fully-Connected Layer:

CNN's fully connected layers are, as the name implies, fully connected to one another. This layer is in the final stage of CNN and is completely connected to the previous layers. It depicts the feature vector for the input.

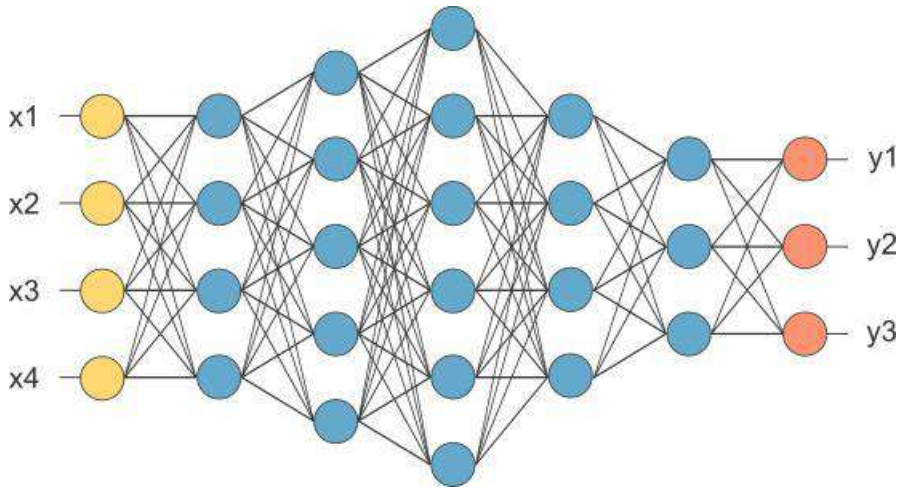


Figure 3.5: Fully connected layer

Neurons in a fully connected layer have all previous layer activation connections. Matrix multiplication with a basis offset id is used to activate the connections. When the network is

trained, the feature vectors control the loss. This feature vector is also used for classification, regression, and feeding input into other layers.

3.1.8 Output Layer:

The final layer of a CNN is the output layer. This layer approximates each class based on the given input image. This layer employs the softmax activation function, which maps the final dense layer and generates vector output that is summed into a single output. It will indicate whether each element belongs to one or more classes.

3.2 Data set Description:

An essential component of excellent research is a benchmark image dataset. We must use a data collection containing Bangladeshi traffic signs because our study is based on the road signs in Bangladesh, where we are doing it. The benchmark dataset, however, is not readily available regarding Bangladeshi traffic signs. Our own data set had to be gathered as a result. We moved to gather visuals of the traffic signs, I went around many important streets in Chittagong.

For a specific road sign, we discovered that several specific signs were not uniform. Pedestrian-crossing signs, for instance, were inconsistently placed, as we Different Chittagong streets were represented, and the frame's contrastive shapes and separate fonts were many image types as well.

Fig 3.6: Collected dataset

Fig 3.7: Collected Dataset

3.1 Class values

- Crossing the street.
- Turning to the left.
- Removal of hydraulic horn.
- Omission of bus stop.
- Transport Hub.
- No rickshaw.
- Blank.
- 30 KM sign.
- 40 KM sign.
- No parking.
- U-turn.
- Mosque.
- Rail-line.
- No crossing.
- One way.
- Both ways.

- Hospital.
- Petrol pump.
- Speed breaker.
- Side road.
- No left turn.

In the future, we intend to collect more datasets in a variety of weather conditions, such as rainy season, foggy winter, etc., as well as at various times of day to capture different lighting conditions. We also intend to expand our data collection outside of Chittagong city to obtain a more enriched and varied dataset of road signs for use in future advanced research.

3.3 Work flow:

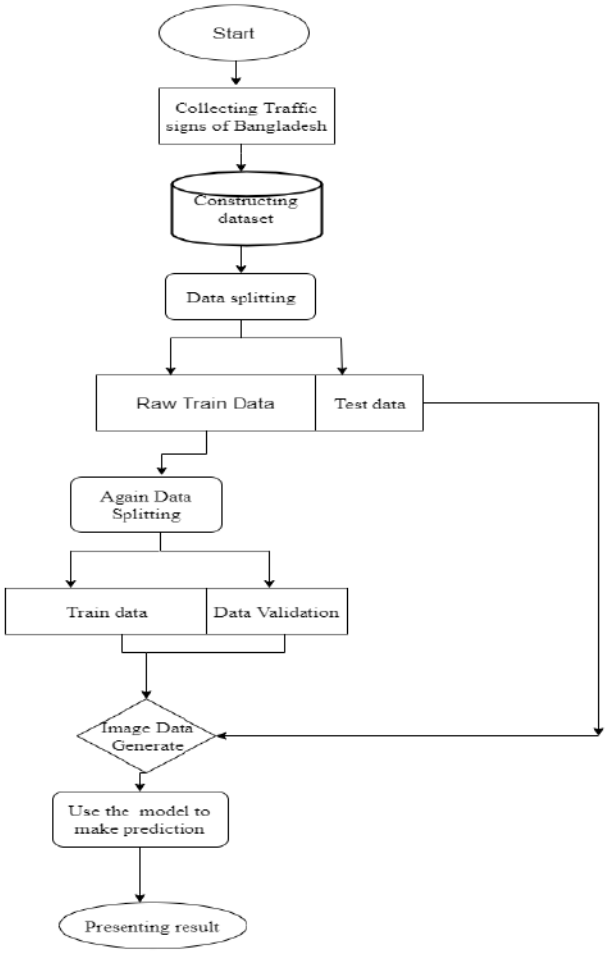


Figure 3.8: Flowchart of the Proposed Model

Our suggested model can identify Bangladeshi Trac symptoms. In order to implement the plan, we took images of the road signs in the city of Chittagong to gather our data set model. The entire work of the suggested model is shown in Figure 3.8. At First, we trained our model using the data (images). Our data was divided into two groups. Raw test data and training data are the sections. Once more, we divided the data into two halves for training.

3.4 Pre-processing

3.4.1 Techniques for Image Pre-Processing

First, we must process our data in order to implement our model. Image processing is required for this. Image processing is a method of producing a consistent set of data. Image processing aids in the improvement of detection systems by reducing variation across data sets, eliminating misinterpretation, and enhancing some valuable features. Image processing systems interpret images as 2D signals. Image processing is necessary for pre-processing the data set because it helps to extricate objects in an image, measure the objects in an image, and visualize the invisible objects in an image. Image processing is typically divided into two types: digital image processing and analog image processing. We will be using digital image processing for our research.

3.4.2 Image pre-processing with Keras

Keras is a very powerful deep learning library used for data augmentation and the construction of machine learning models. Keras' image data generator class generates batches of tensor image data that are augmented in real time. First, the images are converted to 224 x 224 pixels, which is the standard input format for CNN models. We used parameters such as horizontal IP, vertical IP, height shift range, width shift range, zoom range, ll mode, and so on.

3.5 Model Compilation

Following the pre-processing stage, we compiled our models. We left out the fully connected layer when compiling the models. It is necessary to do so because it requires a complete image to function, and a reduced version of an image may not perform properly. An active FC layer may result in over tting. As a result, the object will be incorrectly classified. When the image shape or input size is xed at 224x 224 x 3, the FC layer is prevented from activating.

CHAPTER 4

Model Implementation Proposal

4.1 Conv2D

Conv2D is a 2D Convolution Layer that produces a tensor of outputs by winding a convolution kernel with the layer's input.

-The number of filters from which convolutional layers will learn is a required Conv2D parameter.

-It is an integer value that also determines the number of convolution output filters.

4.2 Max Pooling 2D

Max pooling is a discretization process that uses samples. The goal is to reduce the dimensionality of an input representation (image, hidden-layer output matrix, etc.) and allow assumptions to be made about features contained in the binned sub-regions. This is done to aid over-fitting by providing an abstracted version of the representation. It also lowers computational costs by reducing the number of parameters to learn and provides basic translation invariance to the internal representation.

In the simplest case, the output value of the layer with input size (N, C, H, W) (N,C,H,W), output (N, C, H_{out}, W_{out})($N,C,Hout,Wout$) and kernel_size (kH, kW)(kH,kW) can be precisely described as:

$$out(Ni,Cj,h,w)=m=0,\ldots,kH-1\max n=0,\ldots,kW-1\max\text{input}(Ni,Cj,\text{stride}[0]\times h+m,\text{stride}[1]\times w+n)$$

4.3 Batch Normalization

Batch normalization is a technique for making artificial neural network training faster and more stable by normalizing the layers' inputs by re-centering and re-scaling. While the effect of batch normalization is clear, the reasons for its effectiveness are still being debated. It was thought to alleviate the problem of internal covariate shift, in which parameter initialization and changes in the distribution of the inputs of each layer affect the learning rate of the network. Some researchers have recently argued that batch normalization does not reduce internal covariate shift, but rather smooths the objective function, which improves performance. However, at initialization, batch normalization causes severe gradient explosion in deep networks, which is only alleviated by skip connections in residual networks. Others argue that batch normalization achieves length-direction decoupling and thus accelerates neural networks.

4.4 Flatten

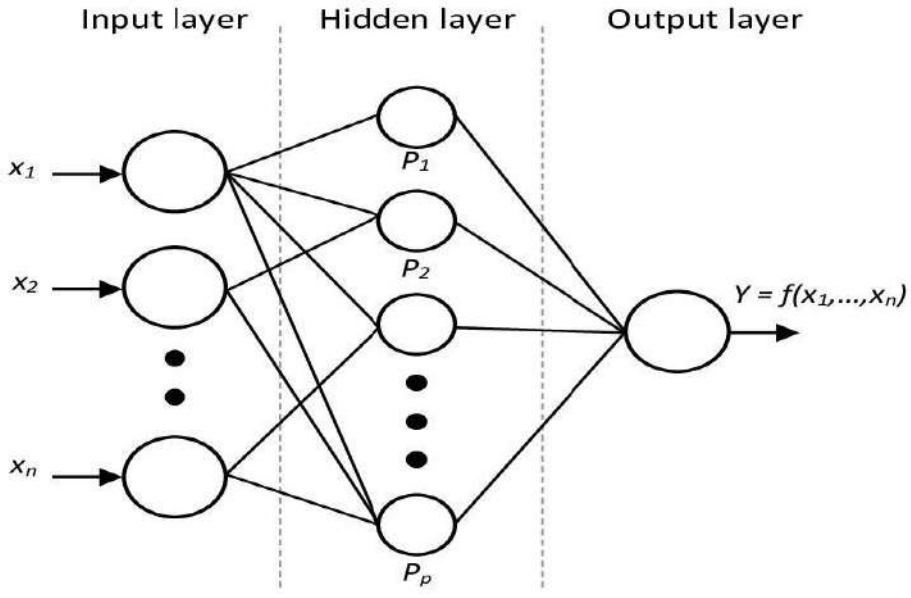
Flattening is a method for combining all of the 2-Dimensional arrays produced by pooled feature maps into a single long continuous linear vector. In order to classify the image, the flattened matrix is fed as input to the fully connected layer. By flattening all of the obtained 2-Dimensional arrays from the pooled feature maps, one very long continuous linear vector is created. The image is classified using the flattened matrix, which is fed into the fully linked layer. To simplify the multidimensional input, the flatten layer is commonly used in the transition from a multidimensional input to a fully linked layer. Configure the Layer differently depending on whether the Tensor Space Model loads a pre-trained model before initialization.

4.5 Dense

A dense layer, also known as a fully connected layer, is a layer used in the neural network's final stages. This layer assists in changing the dimensionality of the preceding layer's output so that the model can easily define the relationship between the values of the data in which the model is working. In this article, we will go over the dense layer in depth, including its significance and function.

A dense layer in any neural network is one that is deeply connected to the layer before it, which means that the neurons in the layer are connected to every neuron in the layer before it. In artificial neural network networks, this is the most commonly used layer.

In a model, the neuron in the dense layer receives output from every neuron in the preceding layer, and the neurons in the dense layer perform matrix-vector multiplication. The row vector of the output from the preceding layers is equal to the column vector of the dense layer in matrix vector multiplication. The row vector must have the same number of columns as the column vector, according to the general rule of matrix-vector multiplication.



4.6 Inception V3

In the development stages of the CNN classifier, the Inception network brought an important breakthrough. Before the concept of Inception, what CNNs did, is to just stack convolution layers deeper and deeper for better performance. On the other hand, the Inception network was complex. It pushed performance in terms of both speed and accuracy. the continuous modification of this network leads to the discovery of several versions of the network. For this research, we will be using Inception version 3.

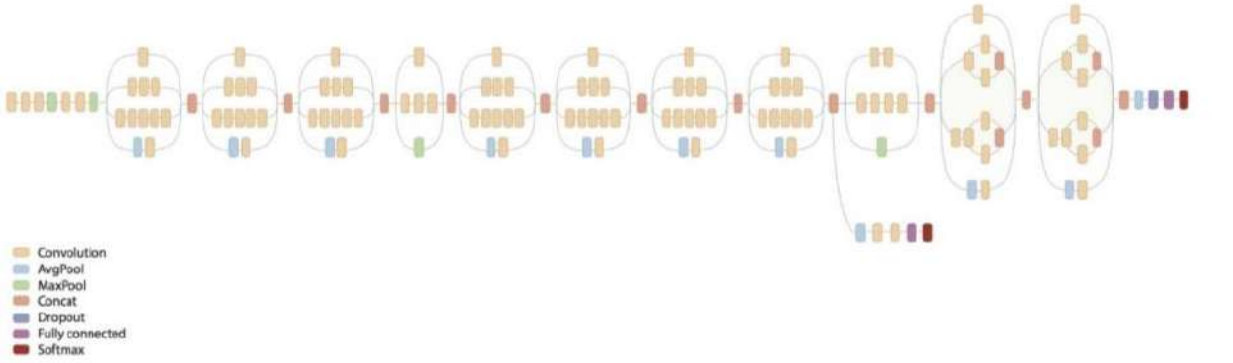


Figure 4.1: Inception V3 architecture

Inception V3 is a CNN model that is trained especially for image classification.

4.7 Resnet 50

Deep Residual net(ResNet) is one of the most groundbreaking discoveries in deep learning. In ResNet, several layers are stacked which helps to train hundreds or even thousands of layers. Even though training this huge amount of data ResNet gives still produces compelling performance. Due to this fascinating performance,many CVA(computer vision applications) have been encouraged. Especially the object detection and face recognition work remarkably well in this model. Since the

discovery of ResNet in 2015, many changes have been made to its architecture for better results[27].

We know that a shallow network appears to function effectively. We built a much deeper network using the weights from the shallow network, and anytime there is a gap,Identity mapping is what we do. This is just the building process. ResNet's central concept is to create a deeper network from a shallow network by duplicating its weights and setting other levels of the deeper network to identity mapping. For training and testing problems in ResNet, conventional wisdom advises digging deeper. The layer does not go deeper by merely stacking the layers together. As the depth increases, the non-linearity also decreases, making deep networks challenging to train. But this seems to be a problem since it generally doesn't seem to work.
To get over this problem, the ResNet paper introduced “Skip connections” which is basically identity mapping or replication. This provides an alternative path for the gradient flow and makes training possible. The number of parameters is also sometimes increase with the kind of approach as you go deeper, you have more layers.

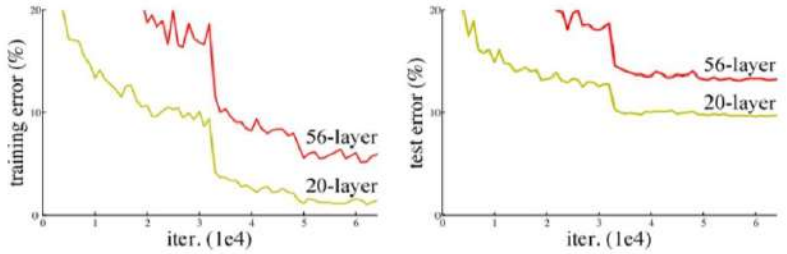


Figure 4.3: Increment of network depth resulting worse performance

4.8 Inception-Resnet V2

The Inception V3 architecture has achieved an exquisite performance with a surprisingly low computational cost. The recent discovery of residual networks that has a more traditional architecture also performed exceptionally well in a challenging environment. This model of CNN is pretrained in ImageNet dataset which has more than a million of images. Somehow its performance was identical to the Inception-v3 model. Inception-ResNet is the study of two ideas combined together.

Inception-Resnet v2 is simply the combination of the Inception and the Residual Network structure.In this structure, multiple sized convolution filters are merged with the residual network. It is designed in such a way to bypass the degradation problem that occurs for the deep structure. This model is used to ease the difficulties of training a deep neural network. Instead of learning unrefereed functions, this model only learns from the layers that have the reference to the layer input. These networks are easy to optimize. On the other hand, the Inception model is exceedingly prone to change. A pure Inception variant without residual connections gives out the same recognition result as Inception-ResNet-v2. In fact, sometimes a Residual-Inception model

outperforms the similar Inception models without residual connections such as- Inception V3 and Inception V4 [19].

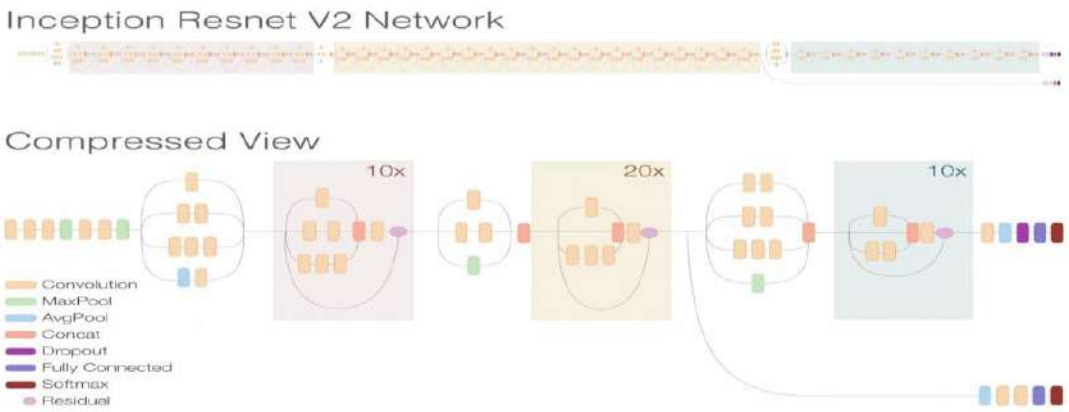
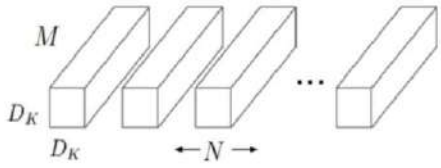


Figure 4.5: Inception-ResNet V2 architecture

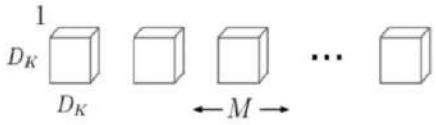
From figure 4.3, we can see that the top of the second Inception-ResNet V2 figure, the full Inception-ResNet V2 network is expanded. This network is considered deeper than the previous Inception V3 networks. The Inception-ResNet V2 architecture is more authentic than the previous networks. This model also requires approximately twice the memory as well as the computation time compared to Inception V3.

4.9 Mobilenet

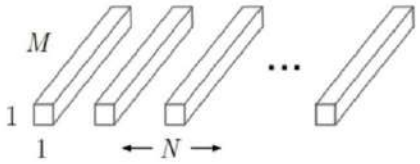
The MobileNet network is basically proposed by Google. The significance of using this network is the advantage of parameter number reduction in NN. This is one of the fastest models in terms of performance and a significantly lower spaceconsuming model. The MobileNet gives an opportunity to solve image classification and detection problems efficiently.



(a) Standard Convolution Filters



(b) Depthwise Convolutional Filters



(c) 1×1 Convolutional Filters called Pointwise Convolution in the context of Depthwise Separable Convolution

Figure 4.7: Depthwise separable convolution

4.10 Xception

Xception a model developed by Google which stands for “Extreme Inception”. This model represents an comparatively stronger version of Inception architecture. This model also uses a depthwise separable convolution and gives out further better even result than Inception V3. Xception presents a clarification of Inception models in CNN. It is an intermediate stepin-between regular convolution and the depthwise separable convolution operation which is followed by a pointwise convolution [25].

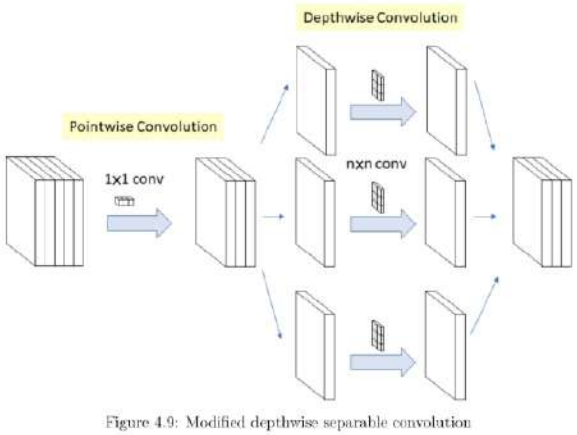


Figure 4.9: Modified depthwise separable convolution

CHAPTER 5

Result -:

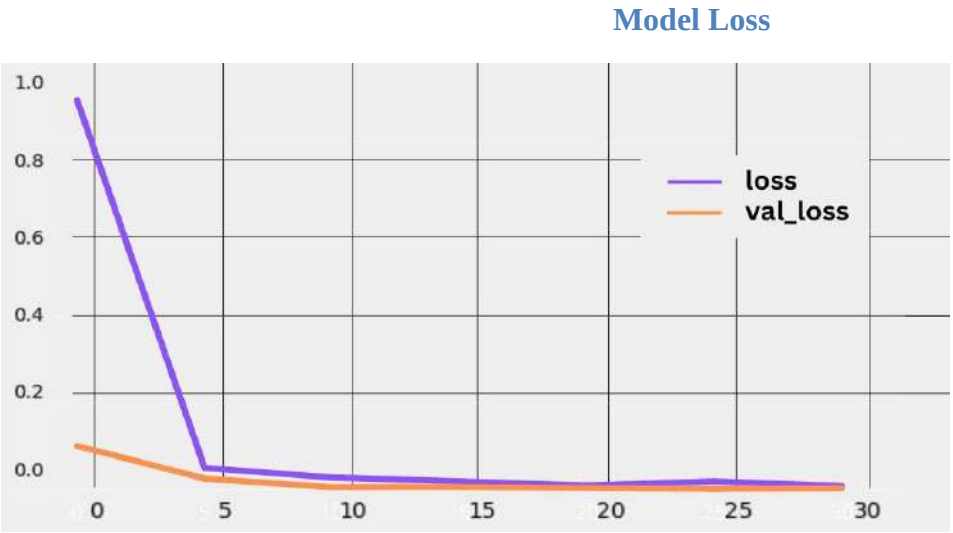


Figure-1

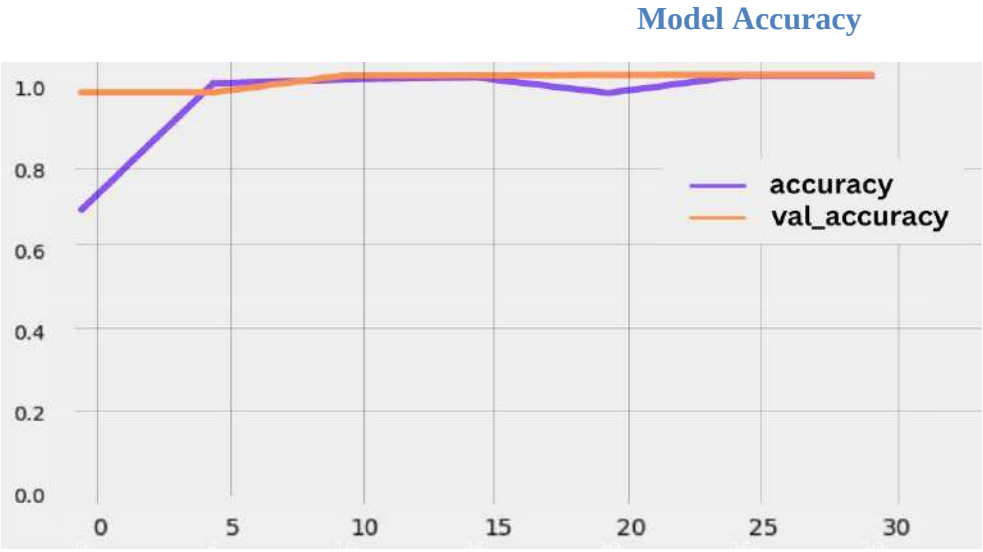


Figure-2

Epoch	Train Accuracy %	Val Accuracy %	Train Loss	Val Loss
1/30	70.01%	96.04%	1.1208	0.1252
5/30	98.03%	96.04%	0.0621	0.0312
10/30	98.92%	99.79%	0.0344	0.0068
15/30	99.34%	99.79%	0.0233	0.0069
20/30	95.90%	99.90%	0.0117	0.0039
25/30	99.69%	99.96%	0.0233	0.0017
30/30	99.69%	99.96%	0.0102	0.0031

Table: Result

5.1 Result Analysis

Right now, we're showcasing our model performances. Our dataset has previously been discussed. To evaluate our model, we used graphical representation.

For detecting and recognizing the traffic signs in an effective manner, we have used three CNN layers. We had improved our deep learning-based system to accurately classify traffic signs from a large dataset. To obtain accuracy from a CNN network with many layers placed step by step, we need to improve.

The above figure is the tranning and validation Loss and Accuracy denoted. We trained for 30 epoch. We see in the above figure validation loss is .0039 and and tranning loss is nearly 0.0117 for 20 epoch. For 30 epoch trainning accuracy was 99.69% and validation accuracy was 99.69% . Our Overall Model accuracy was 99.6%

CHAPTER 6

Conclusion

Every year, millions of people are killed in road accidents around the world. Road accidents are increasing at an alarming rate in Bangladesh. The majority of accidents are caused by a failure to recognize traffic signs. Traffic sign recognition has become mandatory in order to improve that state. The main goal of this thesis is to propose a traffic sign detection system from the standpoint of Bangladesh. While conducting this research, we discovered some serious issues with our country's traffic signs. On our own data, we compared three layers of CNN. We conducted a comparison study between these models and discovered that Mobile Net has the highest accuracy for our data-set, which is approximately 99.6%.

6.1 Future Work

Much research has been conducted on image detection and traffic sign detection using CNN and other neural networks. However, if we look closely, we can see that Tesla, Volvo, and a few other vehicle manufacturers are attempting to implement proper image and traffic sign detecting technologies that will eventually lead us to a fully autonomous transportation system. Again, we gathered our own data sets based on our weather conditions for our work. As a result, there is a huge opportunity to improve and solve any problems that may arise in the future automated transportation system.

References

[1] Ying Sun, Pingshu Ge, Dequan Liu, Traffic Sign Detection and Recognition Based on Convolutional Neural Network, IEEE, 2021

[2] G. Loy and N. Barnes, \Fast shape-based road sign detection for a driver assistance system", in 2018 IEEE

[3] B. Alefs, G. Eschemann, H. Ramoser, and C. Beleznai, \Road sign detection from edge orientation histograms", in 2007 IEEE Intelligent Vehicles Symposium, IEEE,2019.

[4] L. D. Lopez and O. Fuentes, \Color-based road sign detection and tracking", in Image Analysis and Recognition, M. Kamel and A. Campilho, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2021

[5] L.-W. Tsai, J.-W. Hsieh, C.-H. Chuang, Y.-J. Tseng, K.-C. Fan, and C.-C. Lee, \Road sign detection using eigen colour", IET computer vision, vol. 2, no. 3

[6] [Yanzhao Zhu & Wei Qi Yan], Traffic sign recognition based on deep learning, IEEE, 2021

[7] T. T. Le, S. T. Tran, S. Mita, and T. D. Nguyen, \Real time trffiC sign detection using color and shape-based features", in Intelligent Information and Database Systems, N. T. Nguyen, M. T. Le, and J. _Swiatek, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2018

[8] D. Scherer, A. M • uller, and S. Behnke, \Evaluation of pooling operations in convolutional architectures for object recognition", in International conference on arti_cial neural networks, Springer, 2019

[9] D. Cire_san, U. Meier, J. Masci, and J. Schmidhuber, \A committee of neural networks for tra_c sign classi_cation", in The 2011 international joint conference on neural networks, IEEE, 2021

[10] K. Kaplan, C. Kurtul, and H. L. Ak_In, \Real-time traffic sign detection and classification method for intelligent vehicles", in 2017 IEEE

[11] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, \Detection of traffic signs in real-world images: The german tra_c sign detection benchmark", in The 2017 International Joint Conference on Neural Networks (IJCNN)

[12] S. El Margae, B. Sanae, A. K. Mounir, and F. Youssef, \Tra_c sign recognition based on multi-block lbp features using svm with normalization", in 2019

9th international conference on intelligent systemS

[13] Pavly Salah Zaki, Marco Magdy William, Bolis Karam Soliman, Kerolos Gamal Alexsan, Maher Mansour, Magdy El-Moursy,
Kerolos Khalil, Traffic Signs Detection and Recognition System
using Deep Learning, IEEE, 2017

[14] M. Haloi, \A novel plsa based traffic signs classi_cation system", CoRR, vol. abs/1503.06643, 2017

[15] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recog-nition, 2017

[16] Rongqiang Qian, Bailing Zhang, Yong Yue, Zhao Wang, and F. Coenen, \Robust chinese tra_c sign detection and recognition with deep convolutional neural network", in 2018

[17] B. Xu, N. Wang, T. Chen, and M. Li, \Empirical evaluation of recti_ed activations in convolutional network", arXiv preprint arXiv:1505.00853, 2015.

[18] R. Qian, Y. Yue, F. Coenen, and B. Zhang, \Tra_c sign recognition with convolutional neural network based on max pooling positions", in 2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowl-edge Discovery (ICNC-FSKD), IEEE, 2016, pp. 578{582.

[19] C. Szegedy, S. Io_e, and V. Vanhoucke, \Inception-v4, inception-resnet and the impact of residual connections on learning", CoRR, vol. abs/1602.07261, 2016. arXiv: 1602.07261. [Online]. Available: http://arxiv.org/abs/1602.07261.

[20] C. Szegedy, V. Vanhoucke, S. Io_e, J. Shlens, and Z. Wojna, \Rethinking the inception architecture for computer vision", in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2818{2826.

[21] K. Xie, S. Ge, Q. Ye, and Z. Luo, \Tra_c sign recognition based on attributere _nement cascaded convolutional neural networks", in Paci_c rim conference on multimedia, Springer, 2016, pp. 201{210.

[22] L. Abdi and A. Meddeb, \Deep learning tra_c sign detection, recognition and augmentation", in Proceedings of the Symposium on Applied Computing, ACM, 2017, pp. 131{136.

[23] H. H. Aghdam, E. J. Heravi, and D. Puig, \A practical and highly optimized convolutional neural network for classifying tra_c signs in real-time", Inter-national Journal of Computer Vision, vol. 122, no. 2, pp. 246{269, 2017.