Effect of Classifiers in Traffic Signs Classification using Convolutional Neural Network

BY

Syed Khizar Abbas

NUST201463277MRCMS64214F

Supervised By

Dr. Mian Ilyas Ahmad

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENT FOR THE DEGREE OF

MASTER OF SCIENCE

in

Systems Engineering

RESEARCH CENTER FOR MODELING & SIMULATION (RCMS)

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY (NUST),

ISLAMABAD

August 2018

Dedication

I dedicate this effort to all those who have assisted me in any possible way to become what I am today. Their sacrifices seeded my success especially my parents who showed their devoted attention and to faculty members who inspired me all the way.

Certificate of originality

I hereby declare that the work presented in the following thesis titled as "*Effect of Classifiers in Traffic Signs Classification using Convolutional Neural Network"* is my own effort, except where otherwise acknowledged, and that the thesis is my own composition. All the secondary data has been cited properly in dissertation report and accordingly sources have been mentioned in references.

Syed Khizar Abbas

Acknowledgments

First and foremost, I would like to express my gratitude to Allah Almighty who gave me an opportunity to explore my natural abilities and blessing me with the courage to complete this desired task in the required time frame.

All praise for **Almighty ALLAH** Who is the ultimate source of all knowledge. Almighty Allah has made me reach this present pedestal of knowledge with quality of doing something novel, stimulating and path bearing. All respects are for Holy Prophet Hazrat Muhammad (PBUH) who is the symbol of guidance and fountain of knowledge.

I earnestly thank to my supervisor **Dr. Mian llyas Ahmad**, for his keen interest, invaluable guidance, encouragement and continuous support during my research work. I am grateful for his thought provoking and illuminating discussions, sound advices, encouragement, and valuable suggestions. He enabled me not only to tackle the problems more meaningfully on the subject but also provided an easy access to work seriously & sincerely to quest after my objectives. I want to thank him for providing me such a scientific knowledge which will help all of the humanity in a long run. I am thankful to my GEC committee members and other faculty members of RCMS who have been very kind enough to extend their help at various phases of this research, whenever I approached them, and I do hereby acknowledge all of them. I thank, **Engr. Sikander Hayat Mirza, Dr. Salma Sherbaz** and **Dr. Ammar Mushtaq** for their valuable suggestions and concise comments on some of the research papers of the thesis. I am very thankful to Engr. Usman and Sir Hassan for providing us the facilities and research conducive environment at RCMS.

Table of Contents

List of Tables

List of Figures

Abstract

We consider the problem of traffic sign classification through two different neural network architectures. In particular, images of traffic signs with different quality are utilized as an input to a specific neural network and the network predict their corresponding classes. There are different architectures of neural networks and in general, they perform well for different problems. One such group of architectures is the convolution neural network (CNN) that is considered as an efficient tool for image classification. In this thesis, we used a CNN classifier called AlexNet that has a specific architecture and is well used for image classification. The AlexNet predicts a label for the object in the image along with the probabilities of other object labels. Since it is possible to replace the final layers of Alexnet with some other classifier, we consider the use of support vector machine (SVM) for image classification together with the features identified from AlexNet. The performance of the two frameworks, AlexNet with built-in SoftMax classifier and AlexNet with SVM, are compared for different values of dropouts in the fully connected layers of AlexNet. German Traffic Signs Recognition Benchmark (GTSRB) is used for training and validation of the network and it is observed that we can reach to the accuracy of 94.78% with SoftMax and 97% with SVM classifier for the selected data and classes.

Chapter 1

Introduction

Systems Engineering is an interdisciplinary field of engineering that involves the design, analysis and management of different engineering systems throughout their life cycle. Modeling and simulation is becoming important tool for all stages of engineering, including the requirement analysis, design, development, test and verification processes. Here a system or its component is represented in the form of mathematical model and different operations are performed on the model that are observed through computer simulations to check the performance or behavior of that system. This allows us to analyze the system before its physical existence.

Mathematical models are often based on natural laws of physics. However, there are many systems for which the internal relationship between the systems variables is not clear for which we must identify the model through input-output data of the system. This link the field of modeling and simulation with data mining or machine learning. Machine learning is the study or application of the algorithms that can learn (build model) from data and make predictions.

In last few years, we have seen incredible increase in the use of machine learning algorithms in various range of applications. Some of the applications are helpful in the simplification of daily life problems such as image recognition, understanding of data mining, understanding of speech and security task.

In this thesis, we used machine learning algorithms for the problem of traffic signs classification. Convolution neural network-based algorithms AlexNet is used for learning as they are considered efficient feature extractor and classifiers in the literature [20]. The data set for training the algorithm involves different traffic sign images and their class labels.

The collection of information from real-world traffic system is an important component in many applications, such as self-driving car/driverless car, traffic plotting and traffic surveillance. With the arrival of some dataset such as the German Traffic Sign Recognition Benchmark (GTSRB) [1], number of excellent results are produced related to the recognition of European traffic signs in the literature [2], [3], [4].

Lately, the development of deep learning in computer vision has attracted the researchers. Regardless of the excellent performance achieved by CNN, exploring, understanding and considering the internal working principle of CNN remains the most interesting problems to researchers. In recent work, visualization of CNN model and recognition task has been carried out [5], [6], [7]. By inspiration of these works we used the CNN model for image classification especially for traffic signs.

1.1 Problem statement

The excess of vehicles on the roads are causing road congestion and discomfort to drivers. Due to these reasons, a driver may not be as focused as should be and this lead him to miss some of the traffic signs. To resolve this issue, a method should be designed which can assist the driver in detecting these traffic signs. But the first step in implementing this technology is to read the signs and classify them in their respective classes. The other reasons that can affect the sight of the driver are fog, mist and smoke. These factors should be taken under consideration when designing such systems. In this thesis, we analyzed the effects of different type of images i.e. normal, foggy, misty, attenuated, on different classifiers like Support Vector Machine (SVM) and SoftMax.

1.2 Motivation

Self-driving technology is the most debated technology in the automobile industry in recent years. Many companies are producing many autonomous features to their production cars. The motivation behind this technology is to improve car safety and efficiency. Leading engineer of Google's self-driving car wrote that the goal of developing automated car is to "prevent peoples from road accident". The hardware and software of the cars is improving day by day but still there is need of improvement for cost effectiveness and efficiency of traffic signs classification.

1.3 Contribution

Due to machine learning, there is an increase of artificially intelligent things in the market to perform the tasks automatically without intervention of human beings. This automation of task has made human life easier. There are many challenges being faced by the researchers in this regard. One of the biggest challenge for this car is to understand the traffic signals and act accordingly. My contribution in this thesis, is to analyze the effects of classifiers on the basis of different drop outs. Dropouts is the process of reducing the connections between fully connected layers of a neural network that can avoid overfitting and reduce training time.

1.4 Thesis Layout

The remainder of the thesis is presented as follows. In chapter 2, we briefly present the background theory of convolution neural network and its use as image classifier. In Chapter 3, we discuss the use of a specific CNN called AlexNet for image classification and observe the use of support vector machine with AlexNet for classification. Results for these image classifications are shown in Chapter 4. Finally, in Chapter 5, we present the conclusion and future work.

Chapter 2

Literature Review

In this chapter, we will illustrate the background theory associated with the methods discussed in the following chapters. In particular, we discuss the details of convolutional neural network for the problem of image classification. We begin with the discussion of some relevant concepts, such as machine learning, neural network and deep learning. We also discuss the use of convolutional neural network with two different classifiers.

2.1 Machine learning

A branch of computer science that applies algorithms on a data set. Due to this, some portion of human involvement is replaced by the learnable algorithms [8]. One common classification of machine learning algorithms is the supervised and unsupervised learning algorithms [9].

The algorithms in supervised learning identify the parameters of a known pattern for the unknown data set. While in case of unsupervised learning, a complete unknown pattern has to be identified for known data set. Unlike supervised learning, unsupervised learning does not perform well in the start, but when the parameters are tuned properly its performance starts improving. The example for the unsupervised learning is the clustering [9].

Training and testing are two phases implemented in both supervised and unsupervised learning. However, in supervised learning we already know the ground truths of our input data set. While in unsupervised learning we apply different machine learning approaches like clustering and frequency tables etc. unsupervised learning is a bit complex in a sense as we have to later judge up to what extent our clusters are well balanced. Both approaches vary situation to situation. Since in supervised learning we already know the ground truth upon which we tweak our weights using backpropagation. In unsupervised learning, we initially form random cluster or group and relate each row or point of our input data to a specific cluster or frequency column. Then we observe and judge that up to what extent we form clusters. If our random clusters perform well and or precise enough to reveal handsome information of out input point, then we test unknown points about their group or cluster and we know about this new point. Some outliers also mislead or let down our clusters. Different approaches are applied to detect these outliers e.g. distance from the boundary points of a cluster.

Overfitting is also a big threat on prediction and it can be mitigated using the different approaches as dropouts are used in deep learning. Often over fitted models outperform on training data but during prediction it behaves badly and deteriorate the model performance. Solution to overfitting in some model is cross validation and K- fold validation.

Evaluating the performance of machine learning algorithms, depends on their accuracy of the model on test data. However, none of the model performs 100 % so always some error remains, and we have to tolerate and chooses that model which performs better than all others [9].

2.2 Neural networks

Neural network is a programmable model which enables a computer to learn from observation data. Basic name of neural networks at start was artificial neural network. Its name neural is inspired from the neuron of human brain [16], [17]. In past, research comprises of threshold logic unit by Warren McCulloch and Walter Pitts in 1943 were carried out and the concept of perceptron were researched by Frank Rosenblatt in 1957 [11]. The neurons of the artificial neural network are mathematical functions which implemented on computers in series. Guidance is provided in the research of artificial neural networks by the advancement in engineering and mathematical fields rather than biology, guidance in the form of mathematical equations and research papers [8]. Here is a model of artificial neuron.

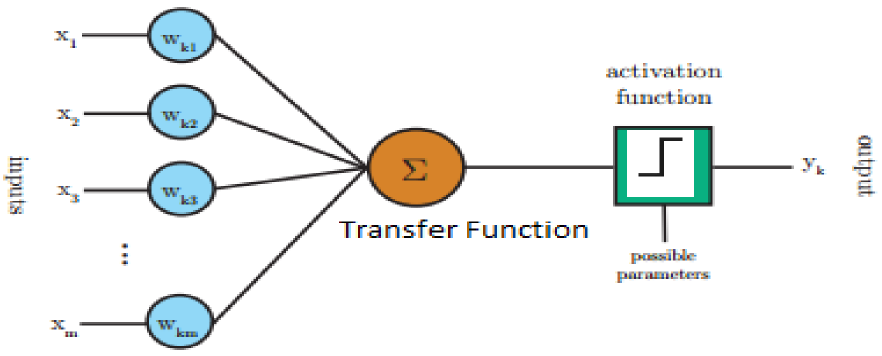


Figure 2.1 : A model of artificial neuron

The model of artificial neuron in the figure 2.1 is based on McCulloch- Pitts model. Here are some parameters and their names, shown in the table.

Parameters	**Names**
_|¯	Transfer function
M	Weight
x_i	Inputs
y_k	Outputs

Table 2.1 : Parameters of artificial neuron

The *kth* neuron receives inputs from the input parameters x_j. The neuron also has an extra parameter, which represents the bias input and almost its value is either +1 or -1. Weight parameters are multiplied to the inputs and then summed into a single output. Then resulted sum are fed to the activation function as an input, y_k result is produced at output.

$$y_k = \phi\ (S_k) = \phi\left(\Sigma_{j=1}^{m}\ w_{kj}x_j\right) \tag{2.1}$$

Equation 2.1 shows the sigmoid function. During training the neural network weights are updated and select to get the desired pattern from inputs.

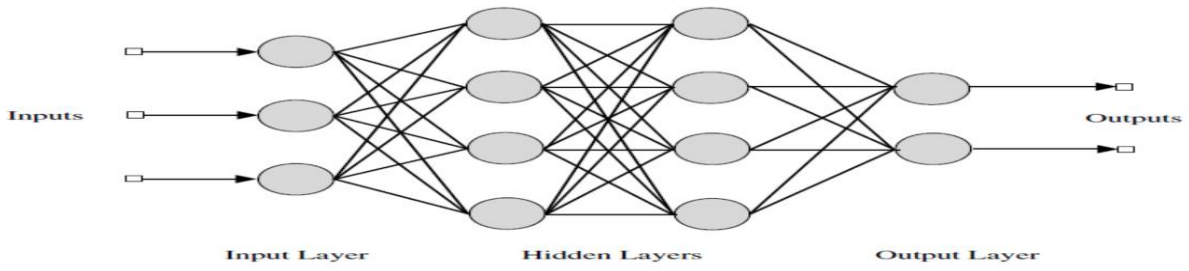


Figure 2.2 : Neural network with multilayer

In figure 2.2 Neural network with multilayer is shown. It is usually used to undertake complicated task. Multilayer network consists of three different types of layers: an input layer, hidden layer and an output layer [9].

2.2.1 Back Propagation in neural network

Training process in neural network works by selecting the appropriate weights for all neurons connected in layers through back propagation algorithm [8], [9].

In the first part of algorithm, input matrix is applied to neural network, and dummy weights are assigned for initializing the process. After the completion of process with these weights, output is compared to desired value. If the error is produced, then back propagation is started. In backpropagation gradient of loss function needs to be calculated. Basically, value of gradient is the error value. Chain rule of derivative is used to solve the loss function of neurons of hidden layer. General form activation function equation is

$$\phi(v) = \frac{1}{1+exp(-v)} \tag{2.2}$$

$$\phi'(v) = \phi(v)[1 - \phi(v)] \tag{2.3}$$

At $\phi(v)$, initial value of weights are calculated. After assigning the weights, process starts, and output is calculated which is compared with the desired results. If error is produced, then again adjust the weights by the given neural network learning rule.

$$w(n+1) = w(n) + a * w(n+1) + \eta * \delta(n) * y \tag{2.4}$$

In equation 2.4,

a is the mobility factor.

η is the training parameter or learning function.

w(n+1) is the new weight.

2.2.2 Deep learning

Deep learning is extension of artificial neural network which previously contain one hidden layer while deep learning involves multiple hidden layers. Furthermore, it uses different activation functions like tanh, ReLu and sigmoid along with multiple update optimizers like sgd, momentum, ada grade and ada delta. Some deep learning architectures involves convolution operations and are best in image processing.

There are different architectures of neural networks, that work well on different problems. In the following we discuss a specific architecture of neural network that in particular perform well for image classification.

2.3 Convolutional neural networks

CNN is the architecture of neural network that performs well in image classification of the traffic signs for the automation of driverless vehicles [14], and its performance is impressive. It consists of three main layers, that is convolutional layer, pooling layer and fully connected layer.

2.3.1 Motivation towards CNN

The problem-solving capacity of traditional neural network is limited because images contains large amount of information and we need to extract the meaningful information from the image.

A simple image like monochromatic consists of 720×480 i.e., (345600) pixels. If we apply single pixel on input of the fully connected neural network, then each neuron in the network requires 345600 weights. If we have full HD image, 2000×1080 then 2160000 weights are required. If the images are in the form of polychrome, then the number of weights is multiplied with the number of channels in a single image. Due to increase in the image size, the number of parameters also increases and large network forms. This large network cause overfitting and slows down the performance [9]. It is very difficult to train neurons one by one to recognize same pattern in lower and upper portion of the image. Traditional fully connected neural network unable to do such type of recognition [15].

2.3.2 Structure of convolutional neural network

Structure of convolutional neural network consist of two main blocks that are feature extraction and classification block. The detail of the structure is given below. Our main objectice in thesis is to Analyze the Effect of Classifiers in Traffic Signs Classification using Convolutional Neural Network. Our work structure looks like Figure 2.3.

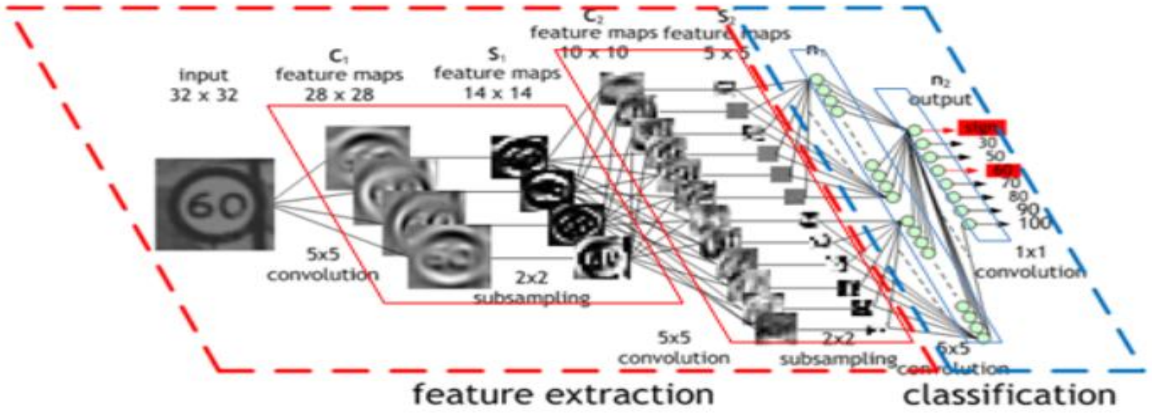


Figure 2.3 : Structural flow of traffic signs Classification

Now we move to describe the architecture of convolutional neural network in details and then discuss what type of work already carried out in traffic signs classification and what is our contribution.

The architecture of CNNs is made up of distinct layers that are usually arranged in multiple stages [18]. The basic layers include convolutional layers, nonlinearities, pooling layers and fully connected layers. In the first level stage typically learns simple visual features like edges or color blobs. The second stage then combines the previous level's features, e.g. learning corners as combinations of edges. Adding more stages results in more complex high-level features, such as faces, depending on data and application. Convolutional layers consist of multiple filters that are defined by their weights. The layer defines the number of filters and their kernel size, the stride in which they are applied and the amount of padding to handle image borders. Example filters learned in the first convolutional layer is shown in Figure 2.4. The convolved output of a filter is called a feature map and a convolutional layer with *n* filters creates *n* feature maps, which is the input for the next layer. For backpropagation, the gradient of the convolution is required, which is the forward-pass convolution with weights flipped along each axis.

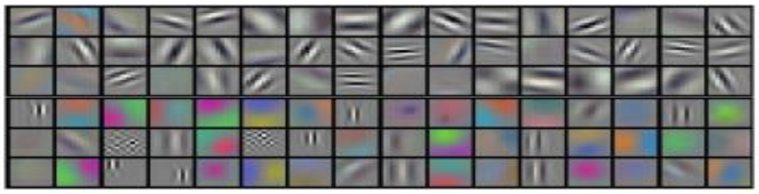


Figure 2.4 : First convolutional layer in Alex Net

Pooling layers reduce feature map resolutions and thereby the sensitivity to shift and distortions, as exact feature location is discarded and only relative and approximate location information remains. Max-pooling selects the maximum output from a local receptive field and is applied in a sliding-window fashion similar to convolutions [19]. Figure 2.5 shows the result of max-pooling. To obtain the gradient it is necessary to store the original location of the selected maximum value since maximum operations act like a routing mechanism in neural networks. An additional benefit of pooling layers is reduced memory cost. For example, 2×2 max-pooling results in output feature maps with half the input's width and height.

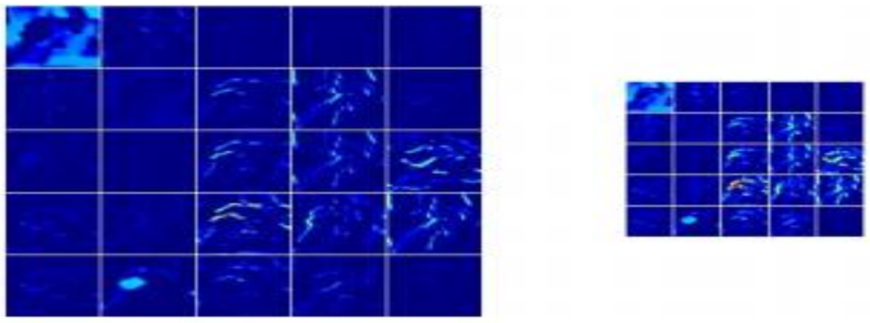


Figure 2.5 : Max pooling feature map

Fully connected layer works separately to hidden layers of a standard neural network. They can be used at the end of a CNN after several stages to compute arbitrary features and output scores (cf. universal approximations) [20].
One method that is applicable to CNNs is transfer learning, where a network is pre-trained with a dataset and fine-tuned to another dataset afterwards [21]. In this case, a network is pre-trained ideally on a large dataset to learn robust filters and features that are generalizable to new data and then trained with application-specific data benefiting from features identical or like the pre-trained ones.

Convolutional neural network characteristically consists of non-linear activation functions. These non-linear activation functions are called rectified linear function. In convolutional neural network rectified linear unit layer is often a separate layer between the convolutional layer and pooling layer. In some systems as [22] normalization layer act as separate layer, which is used for making dimension have same impact. If the proposed network is used for classification, then SoftMax layer is added to the network [9].

2.4 Classification

Convolutional neural networks are very successful deep networks, for image recognition and classification problems. Now we move towards the discussion on related work which is already carried out in image classification and discuss many techniques that were used in the past, then we will discuss our main objective, that is analysis of different classifiers and compare their results.

In image classification, Krizhevsky *et al.* used two techniques to improve the classification results. First, by using the image translator he augmented the training data set and then he used new efficient dropout technique for the regularization [23]. More lately, many researcher new architectures are being analyzed, which are different from former models like LeNet. Szegedy *et al.* [24], his work is based on Google which means that he used new model rather than LeNet. His network consists a lot of convolutional and pooling layers and many number filters with different size, running in parallel.

The Visual Geometry Group of the Oxford university came to play on the same problem image classification which also known as VGGNet. The features which learned by using the VGGNet, are very accurate for not only the image classification, but also can be used for the other purposes like image localization, subdivision of the images, and for textual description for the given image [22].

In 2015, a newly technique batch normalization was developed by Ioffe and Szegedy [25]. He proved that deep neural networks can be made stronger, thus rapidly the training of large data set take place. Batch normalization also addresses the changing of activation functions in the input layer to other layers throughout the training procedure. It normalizes the input layer during the training process of mini-batch.

Before the starting of the decade, the problem of traffic sign recognition extensively researched in machine learning that were only based on the human hand tuned features for both detection and classification of traffic signs. After that, trend was changed, training of classifiers was based on the already extracted features such as Bayes classifier, support vector machine(SVM) [26]. In the starting of the decade, convolutional neural network was first used for the traffic sign recognition problem, along with the large dataset which were publicly available, and it was the set of labelled traffic signs data. In 2011, by using of GTSRB data set [1] first contest for traffic sign classification was carried out at International Conference on Neural Networks (IJCNN).

2.4.1 Support Vector Machine Classifier (SVM)

Support vector machine classifier is a supervised learning model which consists of learnable algorithms that analyzes the dataset for classification. First time support vector machine classifiers were used in [27] and have been prolonged for regression problems in [28] and in preference learning [29],[30]. The primarily form of SVMs is a binary classifier in which output of a learnable function is either positive or negative. A pairwise coupling method is used for implementing the multiclass classification problems by merging of multiple binary classifiers [31],[32].

In this thesis, our main objective is to analyze the different classifiers by using the convolutional neural network. First, we select the dataset, architecture, and classifiers. After that we generate the confusion matrix and calculate the evaluating parameters for the classifiers.

In this thesis, our main purpose is to demonstrate the consistent advantages of replacing the SoftMax classification layer with the support vector machine classifier (SVM). In past [34] SVMs shows the significant results on the popular dataset like MNIST, CIFAR-10, and the ICML 2013 competition of face recognition challenge. In our work, AlexNet architecture producing the images features and the SVMs by using these features train the network and then test the network by test features.

Mathematically, SVM classifier is defined by separating the hyperplane between the two different classes. This method of classification is highly accurate and enormously fast which is best choice for large training data. Basically, SVM is a binary classifier that separates two different classes by the help of support vectors. Here we demonstrate our work in which binary classifier in cascaded (first output will be the input of second classifier) manner is used.

For example, our data is in the form of (x_1, y_1), …. (x_n, y_n). Each pair in the data set represents the labels -1 or +1; in which -1 represents the negative samples and +1 represents the positive samples. A decision function called hyperplane is used to separate these classes. Hyperplane can be defined as a set of point x satisfying:

$$u \cdot x + b = 0 \tag{3.7}$$

where u is the normal vector to the hyperplane, and b is bias input.

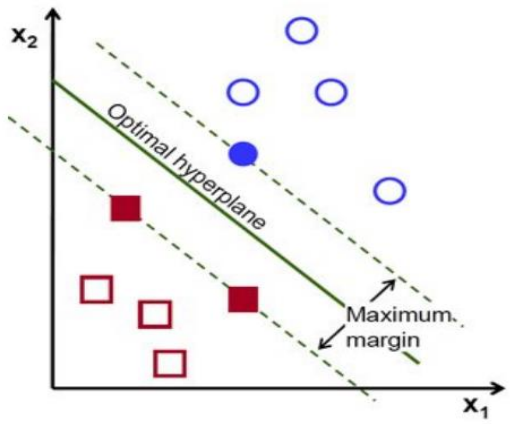


Figure 2.6 : Optimized hyperplane

Figure 3.4 shows that how the squares and circles are separated with the help of different lines. Two parallel hyperplanes are required to separate our data. The distance between the hyperplanes are much important to distinguish two different classes. We can define hyperplanes like:

$$u \cdot x + b = 1 \tag{3.8}$$

$$u \cdot x + b = -1 \tag{3.9}$$

The distance between hyperplanes is $\frac{2}{|u|}$. We need to maximize the distance for best classification. So, we minimize $|u|$. In Support vector machine, distance named as margin. Important point in finding hyperplane is avoiding the data points from falling in the margin. In this thesis, we use constraints to make sure that data points are not fallen into the margin and they are in correct side of the margin.

$$u.x_i + b \geq 1 \qquad \text{if} \quad y_i = 1 \tag{3.10}$$

$$u.x_i + b \leq -1 \qquad \text{if} \quad y_i = -1 \tag{3.11}$$

Constraints are combined in the form of:

$$y_i(u.x_i + b) \geq 1 \qquad \text{for all} \quad 1 \leq i \leq n \tag{3.12}$$

After joining the equation, we get the following problem:

Minimize $||u||$ subject to $y_i(u.x_i + b) \geq 1$, for i = 1, . . ., n (3.13)

When our data is linearly separable above constraint provide best optimized result in classification. But when our data are not linearly separable, loss function introduced. It means that greatest margin can be achieved by allowing classifier to misclassify some data points. For calculating fault tolerance, we introduced slack variable:

$$y_i(u.x_i + b) \geq 1 - \xi_i \qquad (3.14)$$

Where $\xi_i \geq 1 \quad for \ i = 1, \ldots ., n$ (3.15)

If the value of slack variables is between 0 and 1, its mean that data points are not correctly classified and are in the margin (margin error). If the slack variable greater than 1 some data points are misclassified. Then new item is introduced called soft margin and it is denoted as C. After using soft margin optimization problem becomes:

Minimize $\frac{1}{2}||u||^2 + C\,\Sigma_i\ \xi_i$ subject to $y_i(u.x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \ \text{ for } i = 1, \ldots, n$ (3.16)

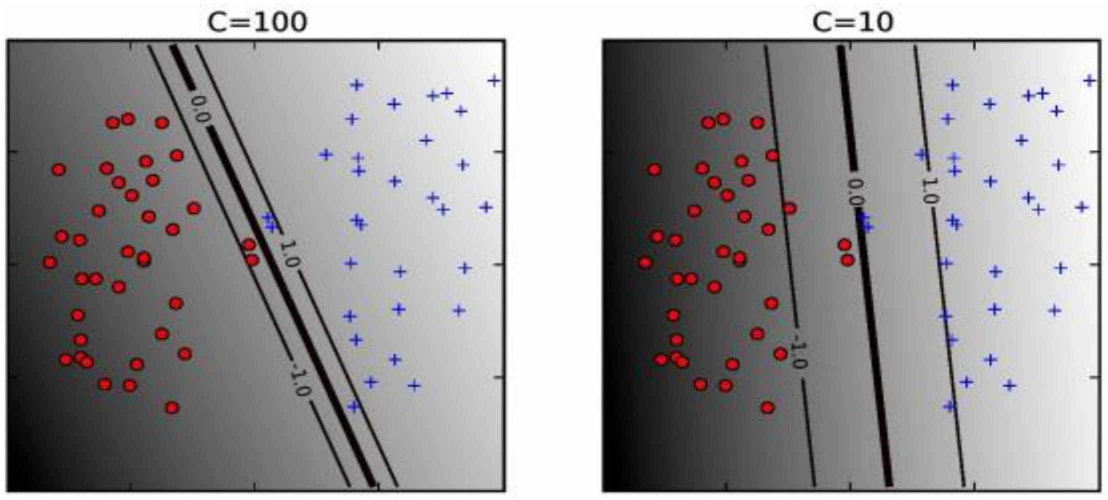


Figure 2.7 : Effect of soft margin

Cortes and Vapnik presented above formulation in [35]. Soft margin constant effect the decision boundary. As shown in figure 3.5, when we select the large value of C, it produces large disadvantage to margin errors. When C has the smaller value some data points becomes margin errors.

2.4.2 SoftMax Classifier

SoftMax Classifier is a general form of binary logistic regression. In mathematics, it is called as a normalized exponential function [9]. It gives normalized class probabilities in which efficient information are present to interpret each class. In SoftMax classifier, mapping function remains the same which is $\mathrm{f(u_i; W) = Wu_i}$ but now we understand, that scores of unnormalized log probabilities are produced for each class. To normalize the log probability, we use cross-entropy loss instead of hinge loss because SoftMax classifier gives only probabilities. Cross entropy loss has the following form

$$L_i = -\log\left(\frac{e^{f_{\mathrm{v}_i}}}{\sum_j e^{f_j}}\right) \tag{3.4}$$

Where f_j is the j-th element of the vector of class score f. Full loss for the data set are calculated through the mean of L_i. $f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}}$ is the SoftMax function. It takes a vector of real value's score in term of z and squashes it to a vector of values between zero and one and their sum must be equal to one. Cross entropy between true distribution t and an estimated distribution e is defined as

$$C(t, e) = -\sum_u t(u)\, log\, e\,(u) \tag{3.5}$$

The main purpose of SoftMax is to minimize the cross entropy between the estimated values of probabilities $\left(\mathrm{e} = \frac{e^{f_{\mathrm{v}i}}}{\sum_j e^{f_j}}\right)$ with the values of true distribution of class scores.

$$\boldsymbol{t(v_i|u_i;W)} = \frac{\boldsymbol{e^{f_{v^i}}}}{\sum_j \boldsymbol{e^{f_j}}} \tag{3.6}$$

From the above expression, it can be explained as the normalized probability is allotted to correct label $v_i$with provision of the images u_i and the parameter W as we explain above that the SoftMax classifier interpret the score of the output vector f. Exponential of above equation provide the unnormalized probabilities and the division changes to normalized score.

2.5 Chapter summary

In this chapter, we covered the main idea of artificial neural network, its background and its advancement to the famous deep neural network called convolutional neural network. Motivation for the convolutional neural network, its background and efficiency for classification especially in traffic sign classification and regarding the traffic sign recognition literature is discussed briefly here.

Chapter 3

Research Methodology

This work relates to image processing and computer vision, so we employed state of the art deep learning model which is convolutional neural network and its pretrained model AlexNet. We altered AlexNet according to our data set and also applied some enhancement in learning rates along with increasing the dropouts to exploit lower computational cost. In rest of the chapter, we briefly discussed the architecture of the AlexNet and flow of inputs till outputs.

3.1 AlexNet architecture

AlexNet is the newest architecture of computer vision that is used for image recognition and classification. It was named after working of Alex Krizhevsky with IIya Sutskever under the supervision of Geoffrey Hinton, they applied this architecture on international ILSVRC-2012 competition, data set of ImageNet are featured.

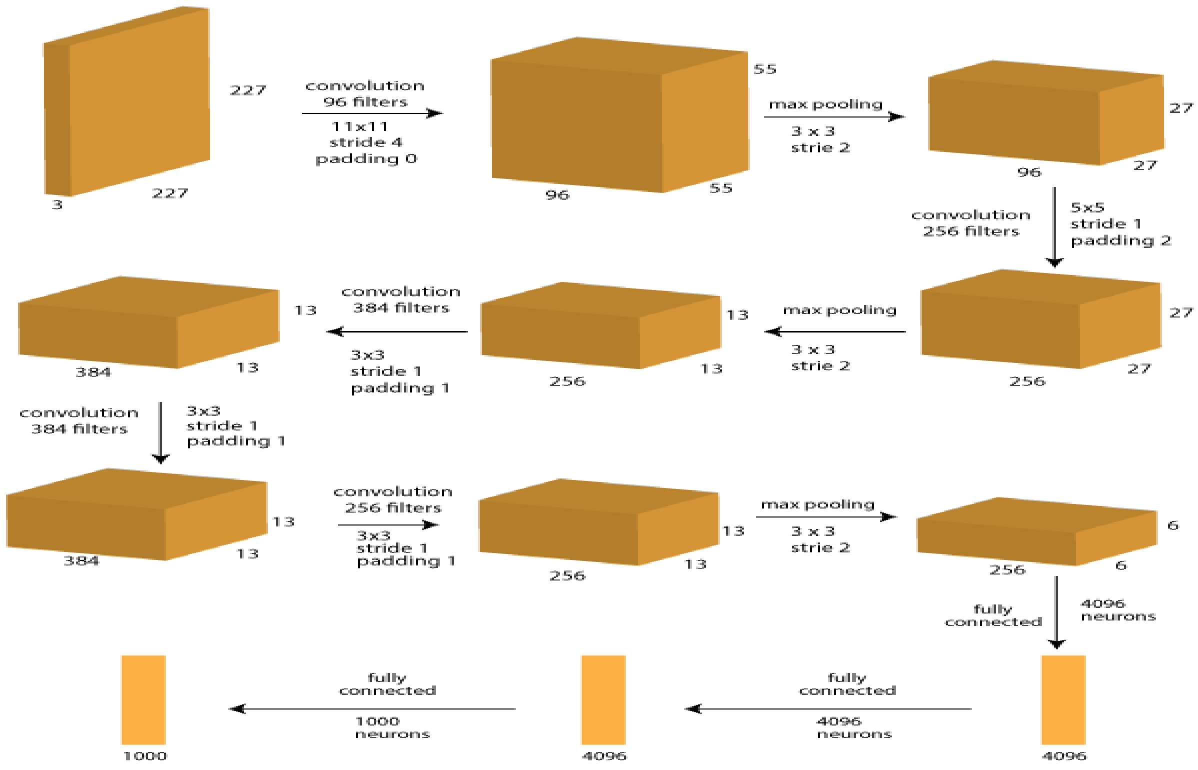


Figure 3.1 : Processing within the AlexNet architecture

In figure 3.1 complete processing of images are shown. Now we discuss the details, how network process the traffic image. It is interconnected network of different layers; main layers are convolutional layer, max pooling layer and fully connected layer.

3.1.1 Convolutional layer

As the name of convolutional layer, it is the main component of the convolutional neural network, and its working principles are inspired by the small processing cells in mammals' visual cortex called hyper complex cells (Hubel and Wiesel, 1962 [33]).

In convolutional neural network, convolutional layer is the main building block, most of the computation of the image feature occurred in this layer. The convolutional layer consists of learnable filters like 3x3, 5x5, but in Alex Net, 11x11 filter is used. In first step which is forward bias, we slide each filter across the width and height of input image matrix completely and then we convolved (dot product) one by one portion of the image matrix with the filter, then combined results of all portion of the image matrix, resulted matrix called activation map, are feed to next layer.

Here we discuss the size of the output volume of the convolutional neural network layer. Depth, stride and zero-padding are the three parameters that select the size of output of convolutional layer. The depth of output volume is that whose value is set before the learning process starts. The values of other parameters are derived from the training. In our proposed method three depth that are Red, Green and Blue are used. Stride is a number, with which we slide the filter. And when we select the stride as 1 we move the filter by one jump. In our work, we use 1 and 2 strides throughout the implementation. By using these strides, we achieved the smaller output spatially. And the next important parameter is padding, it is a mechanism in which we pad the zeros in border areas of the matrix. To save the network from loss of information during learning. We used the zero-padding which is hyper parameter. It also helps to control the output volume. We compute the output volume of neurons from the function of input size (W), the size of convolutional neural network layer (F), the stride (S) and the amount of zero-padding (P) which applied on the border of the matrix to save the information. Following formula is used to calculate the neurons, that are required for the network.

$$\frac{(W-F+2P)}{S+1} \tag{3.1}$$

3.1.2 Nonlinearities

After computing the convolutional neural network layer output, every CNN layer passed through the non-linearity function denoted as

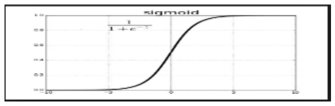
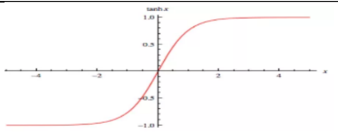
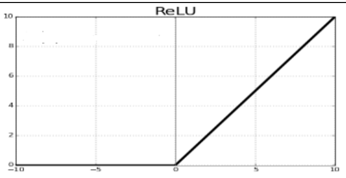
Activations	*Formulas*	*Diagrams*
Sigmoid	$Y_s = \frac{1}{1+e^{-x_s}}$	
Tanh	$Y_s = tanh(x_s)$	
ReLu	$Y_s = max(0, x_s)$	

Table 3.1 : Different activation Functions

Non-linearity function layer has no parameters. In literature, many non-linearity functions are available, we are using rectified linear unit (ReLU) layer.

At the beginning of the neural network, sigmoid activation function was used, that was called hyperbolic tangent. Now a days trend has been changed from artificial neural networks to deep networks, so that large number of input (both positive and negative) causes the huge saturation to the sigmoid unit. In saturation mode, the derivative $f'(x)$ is very close to zero and this limitation is known as vanishing gradient. This slow down the training process in the deep neural networks.

To overcome the difficulty in slowing down of training, we used newly techniques rectified linear unit (ReLU), it accelerates the training process in deep neural network [20]. It does not saturate the positive values in the inputs, it also addresses the vanishing gradients problem. Moreover, ReLUs are very computationally economical and endorses sparse activations. Due to these reasons, ReLUs are most popular choice for deep neural networks in recent years. In our work, every negative value coming from the input value to ReLUs are changed into zero.

3.1.3 Pooling layers

In this layer, we down sampled three-dimensional information of the input data, and then result of these layers are added between the different stages of convolutions. Main purpose of resizing the input data to minimize the number of parameters, computational cost and computational time in the network. Pooling layer operates on every depth slice of the input and resizes the input data, by using the MAX operations. In literature, most common form of pooling layer used with the 2x2 filter and 2x2 stride that is also used in same proportion in our proposed work. During pooling the depth dimensions remain the same. Here are some parameters and equations that are used in pooling process.

- volume of size $W1 \times H1 \times D1W1 \times H1 \times D1$
- Two hyper parameters:
  - their spatial extent F,
  - the stride S,

Produces a volume of size $W2 \times H2 \times D2$ where:

- $W2=(W1-F)/S+1$
- $H2=(H1-F)/S+1$
- $D2=D1$

In this circumstance, maximum activation of small part in the input is achieved by the down sampling during max pooling operation. Mainly in max-pooling 2x2 filter with stride 2 are used.

Our proposed work also carried out. Here is the small portion of input patch, we applied max-pooling with stride 2 and find the output as a down sampled value.

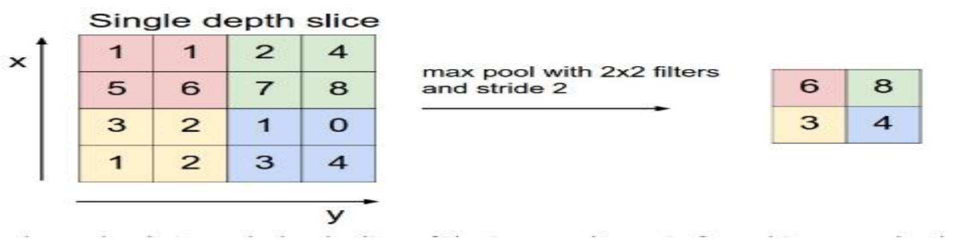


Figure 3.2 : Max-pooling position operation

3.1.4 Drop out layer

Deep neural network is a powerful machine learning system, containing the large number of parameters. So that overfitting become a serious problem due to large number of parameters. It is generally believed that large network is slow to use, it become difficult to deal with the overfitting of the model with the large number of prediction at the test time. Drop out is a key idea to prevent the network from overfitting. The neurons in the neural network are temporarily discarded by the probability P and the formula for this probability is given below:

$$p = \left(p_a = \frac{1}{u}\right) \tag{3.2}$$

3.1.5 Fully Connected layer

Generally, fully connected layer is employed at the end of convolutional neural network. In this layer, neurons have full connections to all activations of the earlier layer, as seen in regular neural networks and calculation of score of the all classes applied to the network are carried out, resulting in the size of [1x1x11], where each of the 11 numbers correspond to score of one class. In our work, we used 11 classes of traffic signs.

3.2 Training of AlexNet with Extended Dropouts

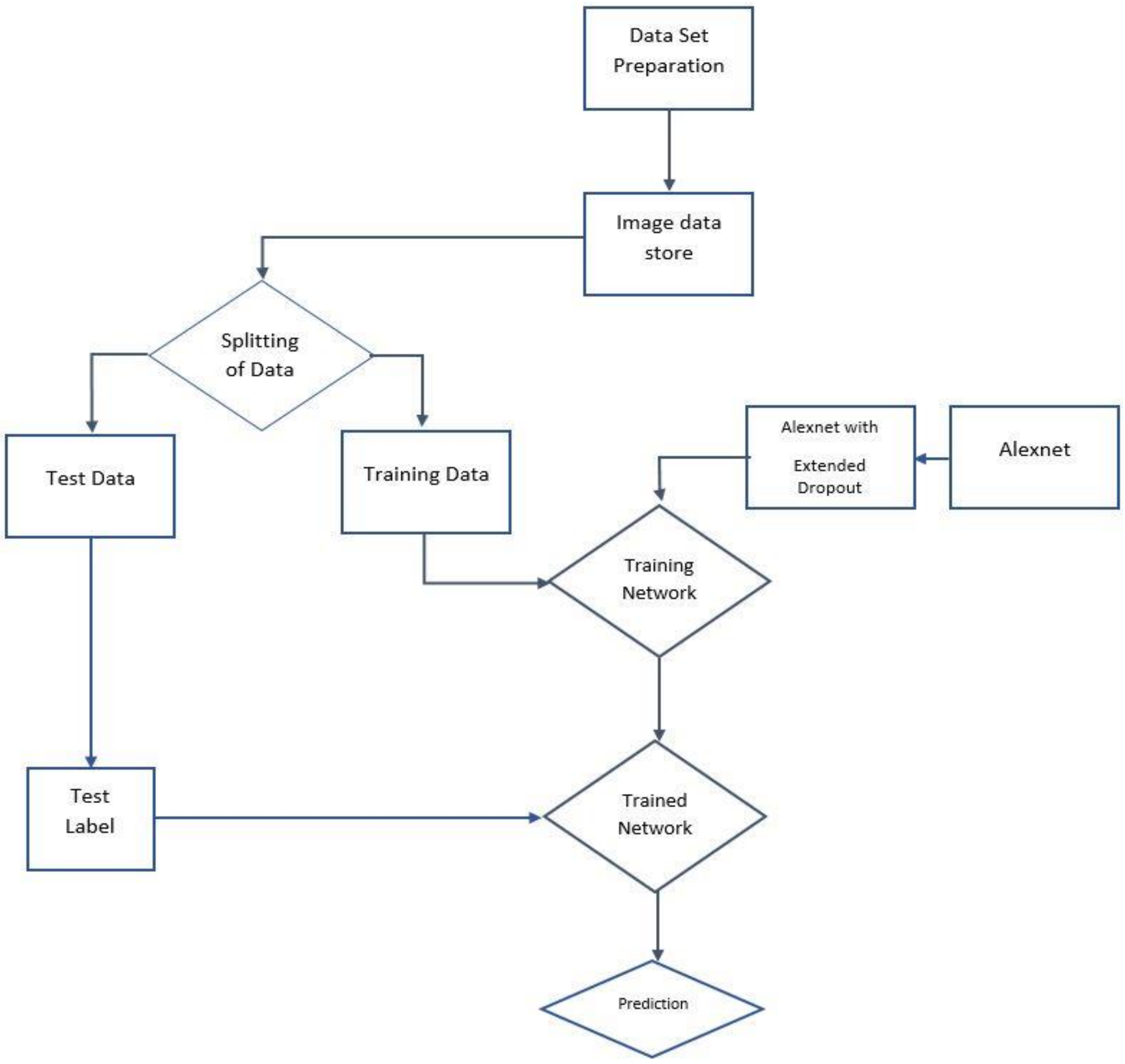


Figure 3.3 : Flow chart AlexNet with extended dropouts

Flow chart in Figure 3.3 shows that first we prepare the data set which means that all images are transformed in same size. And then feed it to image data store then it is flowed to next phase where it is split into two folds, training data and testing data. I have picked a pretrained model of AlexNet and altered its drop outs further to achieve lower computation without sacrificing accuracy. Then the training data is fed to this altered AlexNet model and train it. At last step, i finally got the weights to employ on test data and got the prediction.

3.3 Training of AlexNet with Extended Drop outs with SVM

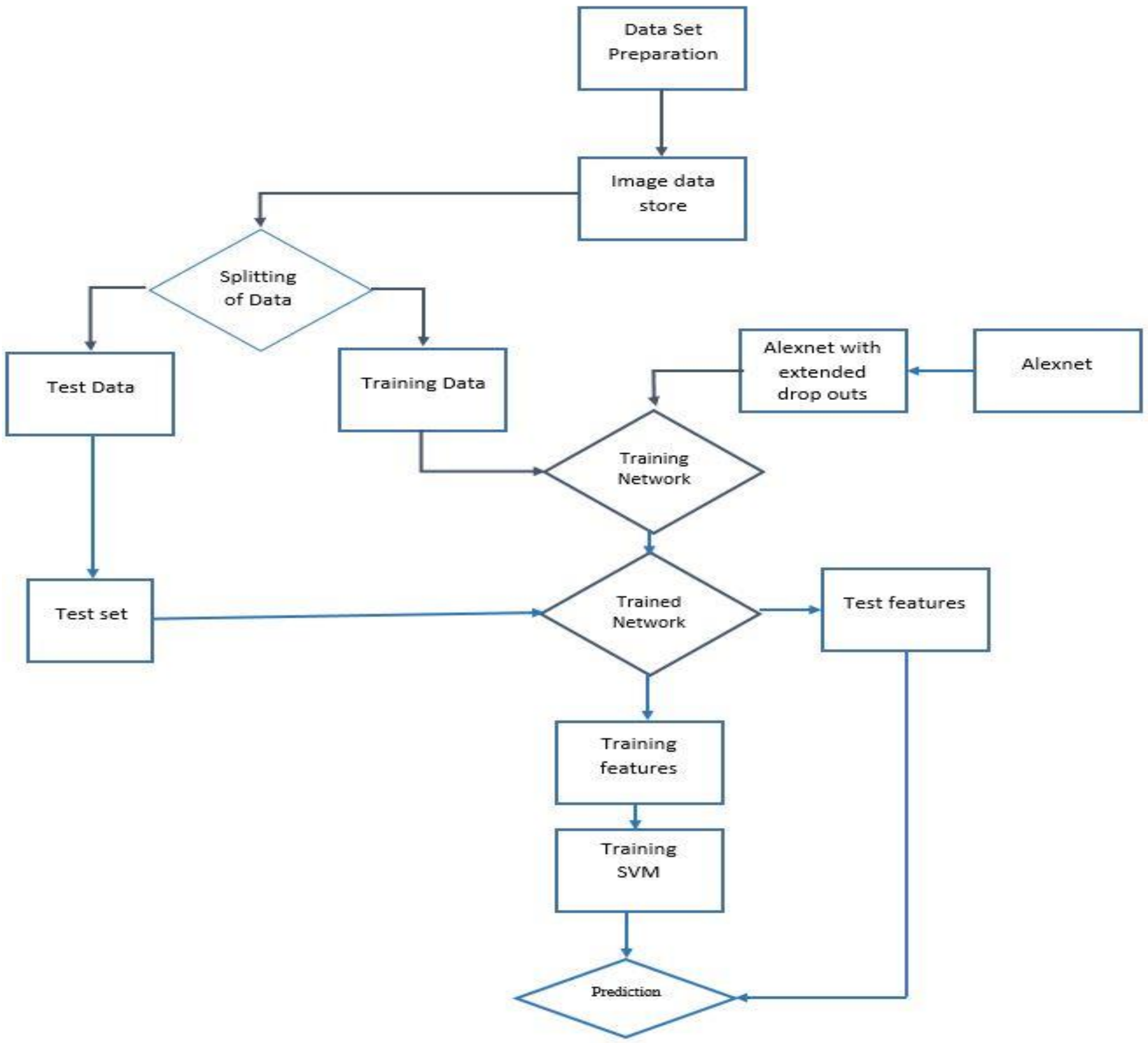


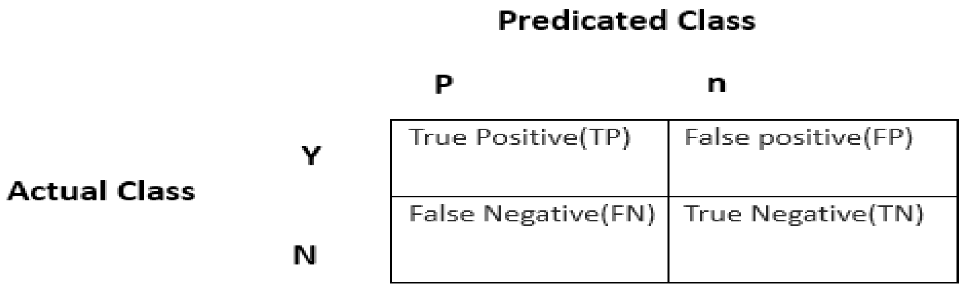
Figure 3.4 : Flow chart of AlexNet with SVM

The only difference in flow chart mentioned in Figure 3.4, is that we extracted certain features from trained model and feed these features for training of SVM classifier. After training, SVM is applied on test features which are also extracted using test set on trained model. SVM classifier produces the desired prediction.

3.4 Classifiers performance measure

In machine learning, a confusion matrix specially used to evaluate the classifiers in the problem of classification and it is also called as error matrix [36]. It is a specific table that is used to visualize the performance of our purposed method for the classification typically by using supervised learning which known as matching matrix or table. In this thesis, we generate the confusion matrixes at different drop outs and at different no. of iterations. we use (0.5) dropout which is default to analyze the classifiers like SoftMax and SVM after that similarly 0.6 and 0.65 dropout are used with increased no. of iterations to make the network cost-effective. In section 4.2.1 different parameters for evaluation the classifiers are discussed.

We started by seeing the problems by using two classes. Each actual value(I) is compared to the one member of the set (p, n) positive and negative of each class labels. A classifier compares from the actual value to the estimated one. There are some models of classification that are producing the continuous output and some models producing the output in distinct manners. To differentiate between actual and predicted classes we use the different labels like [yes, no].



		Predicated Class	
		P	n
Actual Class	Y	True Positive(TP)	False positive(FP)
	N	False Negative(FN)	True Negative(TN)

Figure 3.5 : Confusion matrix for single class

In figure 3.5 confusion matrix is shown it contain all the possible values of the actual and predicted classes that has four possible values TP (True positive), TN (True negative), FP (False positive) and FN (False negative).

True Positive rate or Recall:

If the positive cases were correctly identified, then it is called as true positive and it is calculated as

$$\text{True Positive rate} = \frac{\text{Positve values correctly classified}}{\text{Total Positive values}}$$

False positive rate:

If the positive cases that were incorrectly classified, then it is called False positive and it is calculated as

$$\text{False Positive rate} = \frac{\text{Positive values incorrectly classified}}{\text{Total negative values}}$$

True negative rate or specificity:

If the negative cases that were correctly classified, then it is called as True negative and calculated as

$$\text{True negative rate} = \frac{\text{True negative values}}{\text{True Positive} + \text{True negative}}$$

False negative:

If the negative cases that were incorrectly classified as negative, then it is called as False negative and it is calculated as

$$\text{False negative rate} = \frac{\text{False negative values}}{\text{False Positive} + \text{False negative}}$$

Accuracy:

Accuracy is the total no of predictions that were correct. It can be calculated by using the following equation.

$$\text{Accuracy} = \frac{\text{All True Positives}}{\text{Sum of Confusion Matrix}}$$

Precision

it is proportion of positive case that were positive are predicted.

Precision = TP/TP +FP

F 1 Score

It is used to calculate the test accuracy. It contains both recall and precision.

F1 Score = 2 *(Recall)(Precision)/ (Recall + precision)

Now we are showing the results that we calculated through the different dropouts and iterations.

3.5 Chapter summary

In this chapter, we covered the main methodology that is used to implement the desired problem. Our focus is to analyze SoftMax classifier which is CNN classifier and SVM classifier, we discussed how it is implemented and classified the traffic signs. Results and reasons for selection of SVM classifier are discussed in next chapter. We will show how it is better for traffic signs classification at the different rate of drop outs and no. of iterations.

Chapter 4

Result and Discussion

In this section, we briefly discuss the results of our purposed structure and observe the performance of SoftMax and SVM classifiers.

4.1 Data set

The German Traffic Sign Recognition Benchmark (GTSRB) dataset is freely available like others. It is divided into 43 classes [1]. In figure 4.1 representation of all 43 classes are shown. This dataset was initially published in 2011 during Traffic sign recognition competition in International Joint Conference on Neural Networks (IJCNN). We use 11 classes of the dataset to analyze the classifier results.

The classification of traffic signs meets several challenges, including lighting, atmospheric effects and physical damages of signs. In figure 4.2 the images that are considered as difficult to classify are shown. The resolution of some images is not clear, and it is highly possible that the drivers fails to classify the signs, while on move. As discussed in previous chapter, we used neural network classification to address this problem. The dataset is divided into a training set data and a test data for prediction classification. AlexNet architecture with dropout (0.5, 0.6, 0.65) as a feature extractor and by using these features we train the support vector machine classifier.

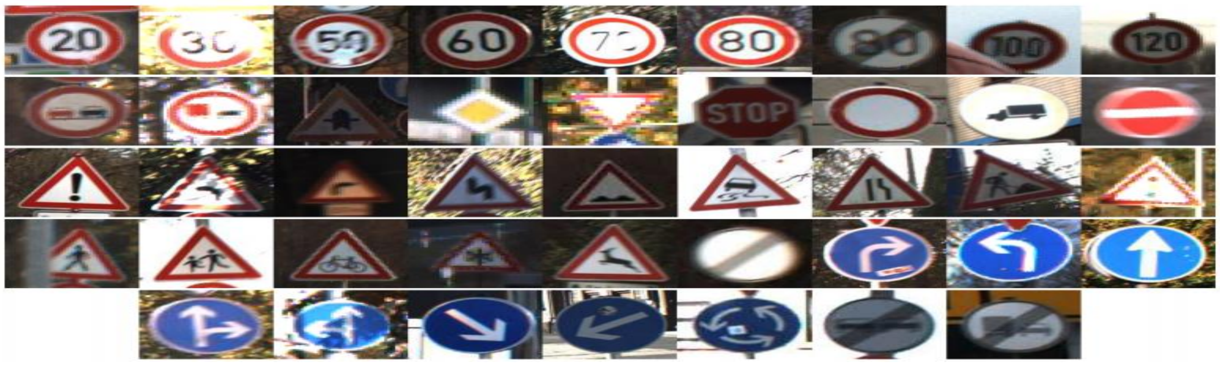


Figure 4.1 : Representation of all classes

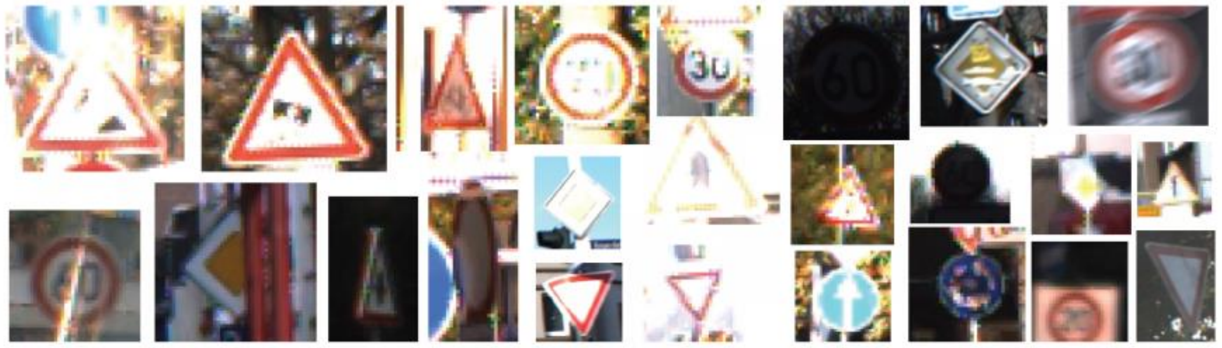


Figure 4.2 : Images difficult to classify

The data set was in raw form and therefore we have to arrange them in a specific folder and these folders are labeled appropriately. To conduct the tests, labeled dataset is used. In which 70% of the dataset is used for training purpose and the rest 30 % of dataset is used for test set. In table 4.1 complete labeled dataset is represented. In this thesis, we used 11 classes to analyze the classifiers on different dropouts.

S.no	Class name	Total images	S.no	Class name	Total Images
1	Keep right	88	7	construction	31
2	speed limit 100	41	8	Give way	83
3	speed limit 50	81	9	stop	32
4	speed limit 60	30	10	danger	38
5	speed limit 70	68	11	Priority road	85
6	speed limit 80	53	12		

Table 4.1 : Traffic Signs Data Set

4.2 Classification with AlexNet Architecture

After dataset labeling, we used transfer learning of AlexNet. In which we transfer the layers of network and update the layers with our requirement. In this thesis, our focus is to analyze the classifiers with variable range of dropouts. Drops outs are the parameters to avoid the overfitting and minimizing the network cost. We purposed that if we increased the dropouts then the accuracy of the whole system suffers. Because drops outs contain the useful information, to overcome this difficulty we increased the no. of iteration, due to increased iteration more back propagation in the convolutional neural network takes place until the desired results meets. We represent our results in two forms, first we generate the confusion matrix form the trained classifier then we construct the table which consist of key parameters are used to evaluate the performance of the classifier.

4.2.1 Training with 0.5 dropout and 150 iterations

In table 4.2 we select the default setting of dropout which is 0.5. Then train the classifier by using default setting of dropout with 150 no. of iterations. After training above 4.1 table of confusion matrix are generated. Then the key parameters are calculated in table 4.3.

No. of Classes	Speed limit 60	Speed limit 70	Speed limit 80	Speed limit 100	Stop	Priority road	Danger	Construction	Give way	Keep right	Speed limit 50
Speed limit 60	0.778	0.222	0	0	0	0	0	0	0	0	0
Speed limit 70	0	1	0	0	0	0	0	0	0	0	0
Speed limit 80	0.0741	0	0.889	0.0370	0	0	0	0	0	0	0
Speed limit 100	0	0	0	0.9615	0	0	0	0	0.0385	0	0
Stop	0	0	0.370	0	0.9630	0	0	0	0	0	0
Priority road	0	0	0	0	0	0.9231	0	0	0	0.0769	0
Danger	0	0	0	0	0	0	0.8	0	0.06667	0.133	0
Construction	0	0	0	0	0	0	0	1	0	0	0
Give way	0	0	0	0	0	0	0	0	1	0	0
Keep right	0	0	0	0	0	0	0	0	0	1	0
Speed limit 50	0	0	0	0	0	0	0	0	0	0	1

Table 4.2 : Confusion matrix (0.5)

Name of Classes	True Positive	False Positive	False Negative	True Negative	Name of Classes	True Positive	False Positive	False Negative	True Negative
Speed limit 60	0.7778	0.0741	0.2222	2.2222	Danger	0.8000	0	0.2000	10
Speed limit 70	1	0.2222	0	10	Construction	1	0	0	10
Speed limit 80	0.8889	0.0370	0.1111	9.2593	Give way	1	0.1051	0	10
Speed limit 100	0.9615	0.0370	0.0385	10	Keep right	1	0.2103	0	10
Stop	0.9630	0	0.0370	10	Speed limit 50	1	0	0	10
Priority road	0.9231	0	0.0769	10					

Table 4.3 : Results for SoftMax 0.5 drop

Table 4.3 shows the results of SoftMax classifier. The overall accuracy and error rate of the classifier is calculated as below

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

$$Over\ all\ accuracy = \frac{All\ True\ Positives}{Total\ sum\ of\ Confusion\ matrix}$$

$$= \frac{10.3146}{10.9998} = \quad 0.9378$$

= 93.78%

$$Error\ \ rate = \frac{FN+FP}{TP+TN+FP+FN}$$

$$Over\ all\ error\ rate\ = \ 1 - \ Over\ all\ accuracy$$

=1-93.78 = 0.0621

=6.21%

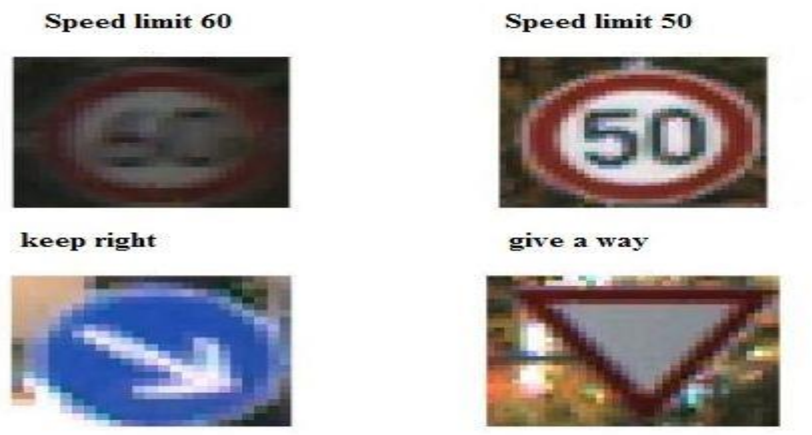


Figure 4.3 : Prediction by SoftMax at 0.5 drop out

Figure 4.3 shows the prediction made by SoftMax classifier. In which we see that speed limit 60, speed limit 50, keep right and giver a way correctly classified.

4.2.2 Training with 0.5 dropout and 200 iterations

Now we select the default setting of dropout which is 0.5 and increase the number of iterations then train the network with this setting and see the effects on classifier. Like the above, confusion table is generated. Below we calculate the overall accuracy and overall error rate. We see that accuracy increased.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Over\ all\ accuracy = \frac{All\ \ True\ \ Positives}{Total\ sum\ of\ Confusion\ matrix}$$

$= \frac{10.4275}{11.0016}$ = 0.9478

= 94.78%

$$Error\ \ rate = \frac{FN + FP}{TP + TN + FP + FN}$$

$$Over\ all\ error\ rate\ = \ 1 - \ Over\ all\ accuracy$$

= 1-0.9478 = 0.521

= 5.21%

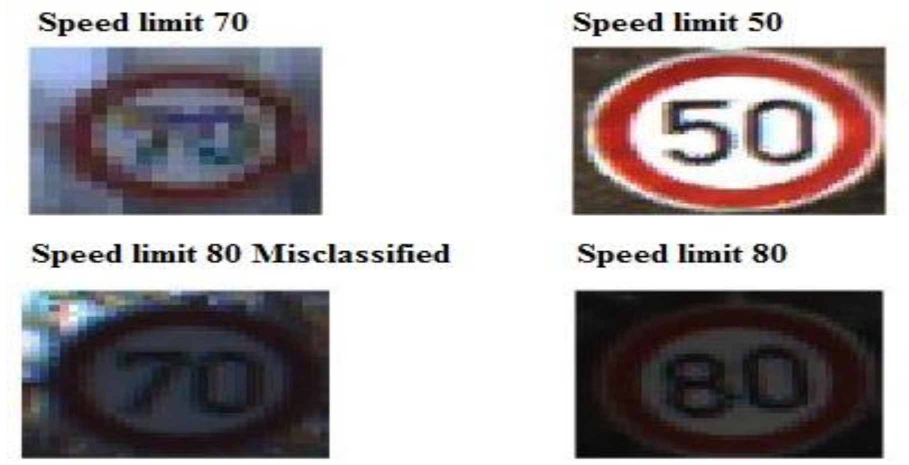


Figure 4.4 : Prediction by SoftMax

Figure 4.4 shows that speed limit 70, speed limit 50, speed limit 80 are correctly classified and speed limit 70 misclassified as speed limit 80.

4.2.3 Training with 0.6 dropout and 200 iterations

In this portion, we increase the drop outs with 0.6. And then train the network by applied 200 iterations as a setting of the network. Then we see the results that how much classifier is effected either effect is positive or not. We observe that the accuracy slightly reduced to approximately 94.17%. Same as above confusion table is created, from which we calculate over all accuracy and error rate.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Over\ all\ accuracy = \frac{All\ \ True\ \ Positives}{Total\ sum\ of\ Confusion\ matrix}$$

$$= \frac{10.264}{10.9596} \quad = \quad 0\ .936$$

$$= 93.6\%$$

$$Error\ \ rate = \frac{FN + FP}{TP + TN + FP + FN}$$

$$Over\ all\ error\ rate\ = \ 1 - \ Over\ all\ accuracy$$

$$= 1 - 0.936 = 0.063$$

$$= 6.3\ \%$$

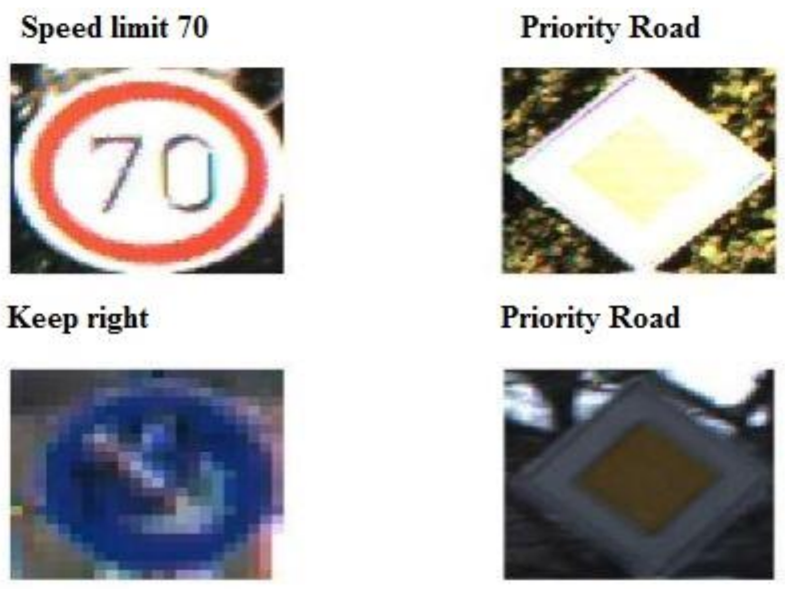


Figure 4.5 : Prediction at 0.6 dropouts

Figure 4.5 shows that speed limit 70, keep right and priority road are correctly classified.

4.2.4 Training with 0.65 dropout and 200 iterations

At this point, the number of iteration remains same as 200 and increase the drop out with 0.65. We trained the network with these setting and observed that results that are produced in default setting (0.5 drop out) with 150 iterations has less accurate than the result produced with 0.65 and 200 iterations setting. When we increased the both dropout and iterations not only accuracy improved but also the overfitting of model and cost of network is minimized.

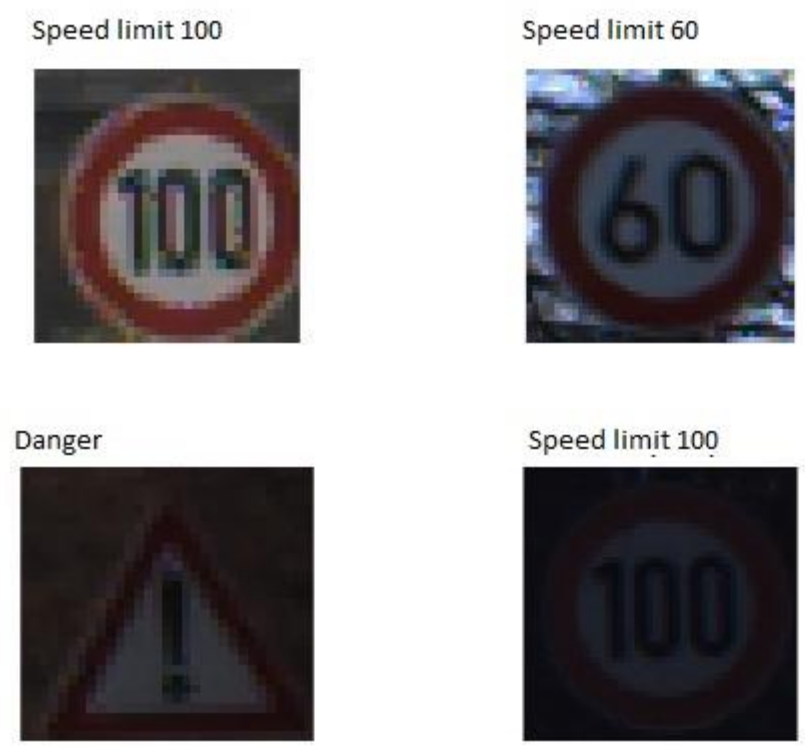


Figure 4.6 : prediction at 0.65 dropouts

Figure 4.6 shows that speed limit 60, speed limit 100 and danger traffic signs are correctly classified. We observed these predictions at 0.65 with 200 iterations.

Now we calculate overall accuracy and error rate. We observed that due to increase of dropouts slightly accuracy is reduced.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Over\ all\ accuracy = \frac{All\ \ True\ \ Positives}{Total\ sum\ of\ Confusion\ matrix}$$

= 0.9276 = 92.76 %

$$Error\ \ rate = \frac{FN+FP}{TP+TN+FP+FN}$$

$$Over\ all\ error\ rate\ = 1 - \ Over\ all\ accuracy$$

= 1-0.9276= 0.0724= 7.24 %

4.3 Classification with AlexNet Architecture and Support Vector Machine (SVM)

In this section, we used the newly pretrained network by transfer learning which is trained on one thousand of images and these no. of images are increased by using the data augmentation. Data augmentation uses the different angle of rotations to increase the images. In our case $20°$ of rotation is used to increase the number of images but these images are virtually located on computer. We used transfer learning with different dropouts, due to increase in drop out slightly accuracy is reduced to compensate the accuracy and reducing the network cost, we increase the number of iterations so that we achieved optimized accuracy. Support Vector Machine classifier (SVM) is trained on the features that are extracted by using the newly trained network. After training the SVM, we generate the confusion matrix by using SVM prediction for the test data. From the table of confusion matrix some evaluating parameters of SVM classifier are calculated. Now we are going to describe the results of SVM like above in two forms first one confusion matrix and second one is the parameters of confusion matrix

4.3.1 Training SVM by pretrained network with 0.5 drop out and 150 iterations

In this section, we used the pretrained network that has been trained on 0.5 drop out and 150 iterations. We input the image data to pretrained network and it extract the features of the input data, these features are used to train the SVM classifier. Below we describe the confusion matrix generated by SVM.

No. of Classes	Speed limit 60	Speed limit 70	Speed limit 80	Speed limit 100	Stop	Priority road	Danger	Construction	Give way	Keep right	Speed limit 50
Speed limit 60	0.9	0.1	0	0	0	0	0	0	0	0	0
Speed limit 70	0	1	0	0	0	0	0	0	0	0	0
Speed limit 80	0	0	1	0	0	0	0	0	0	0	0
Speed limit 100	0	0	0	0.963	0	0	0	0	0.370	0	0
Stop	0	0	0	0	1	0	0	0	0	0	0
Priority road	0	0	0	0	0	0.9231	0	0	0	0.0769	0
Danger	0	0	0	0	0	0	1	0	0	0	0
Construction	0	0	0	0	0	0	0	1	0	0	0
Give way	0	0	0	0	0	0	0.05	0	0.95	0	0
Keep right	0	0	0	0	0	0	0	0	0	1	0
Speed limit 50	0	0	0	0	0	0	0	0	0	0	1

Table 4.4 : Confusion Matrix SVM (0.5)

No. of Classes	True Positive	False Positive	False Negative	True Negative
Speed limit 60	0.9000	0	0.1000	1
Speed limit 70	1	0.1000	0	10
Speed limit 80	1	0	0	10
Speed limit 100	0.9630	0	0.0370	10
Stop	1	0	0	10
Priority road	0.9231	0	0.0769	10
Danger	1	0.0500	0	10
Construction	1	0	0	10
Give way	0.9500	0.0370	0.0500	10
Keep right	1	0.0769	0	10
Speed limit 50	1	0	0	10

Table 4.5 : Results for SVM with 0.5 drop out

Table 4.4 shows the confusion matrix that is formulated by SVM and Figure 4.5 shows the parameters that are calculated to evaluate the SVM classifier. We observed that SVM performance is better than the SoftMax. The overall accuracy and error rate of the classifier is calculated as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Over\ all\ accuracy = \frac{All\ \ True\ \ Positives}{Total\ sum\ of\ Confusion\ matrix}$$

= 0.976

= 97.6 %

$$Error\ \ rate = \frac{FN+FP}{TP+TN+FP+FN}$$

$$Over\ all\ error\ rate\ = 1 - \ Over\ all\ accuracy$$

= 1- 0.976

= 0.024

= 2.4 %

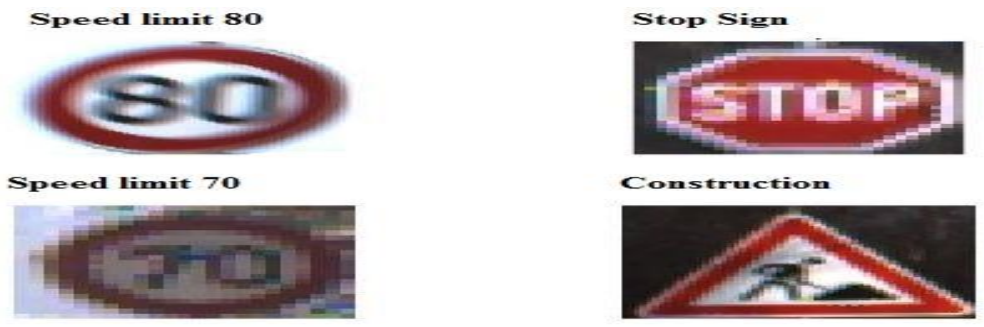


Figure 4.7 : Prediction by SVM at 0.5 drop

4.3.2 Training SVM by pretrained network with 0.5 drop out and 200 iterations

In this section, we used the previously trained network, that was trained on 0.5 drop out and 200 iterations. By using this network, we train the SVM classifier with rich features that is extracted from the pretrained network. After that same as above, create the confusion matrix and then calculate the evaluating parameters for classifiers.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Over\ all\ accuracy = \frac{All\ \ True\ \ Positives}{Total\ sum\ of\ Confusion\ matrix}$$

$$= \frac{10.68}{11} = 0.97$$

$$= 97\ \%$$

$$Error\ \ rate = \frac{FN+FP}{TP+TN+FP+FN}$$

$$Over\ all\ error\ rate\ = 1 - \ Over\ all\ accuracy$$

$$= \quad 1 - 0.97$$

$$= 3\ \%$$

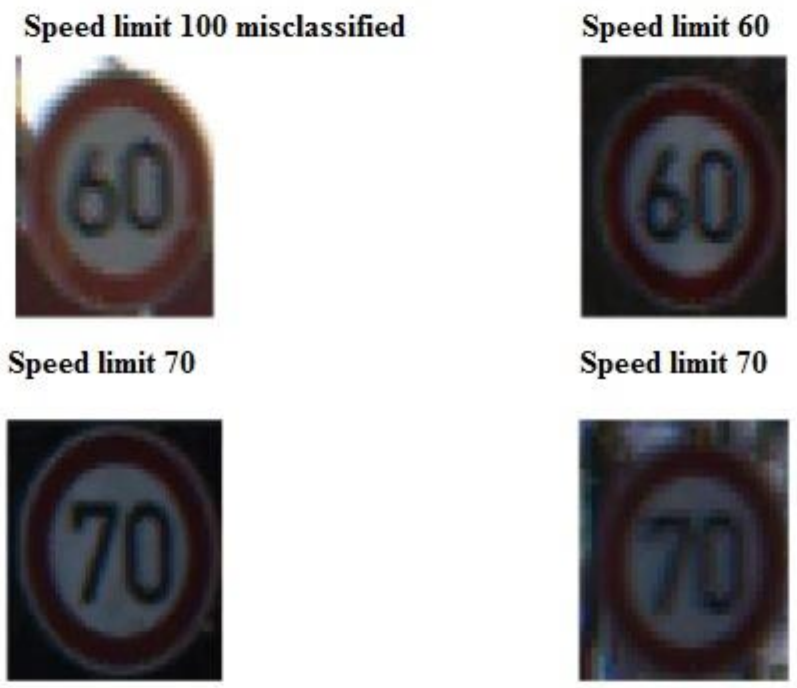


Figure 4.8 : Prediction by SVM

4.3.3 Training SVM by pretrained network with 0.6 drop out and 200 iterations

In this section, we use the network that was trained with the 0.6 drop out and 200 iterations setting. After that same as above we extract the features of the images and trained the SVM. Then create the confusion matrix by trained SVM. The overall accuracy and error rate is calculated as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Over\ all\ accuracy = \frac{All\ \ True\ \ Positives}{Total\ sum\ of\ Confusion\ matrix}$$

$$= \frac{10.6}{10.9996} \ \ = 0.963$$

= 96.3 %

$$Error\ \ rate = \frac{FN+FP}{TP+TN+FP+FN}$$

$$Over\ all\ error\ rate\ = 1 - \ Over\ all\ accuracy$$

= 1 – 0.963

= 0.037

= 3.7 %

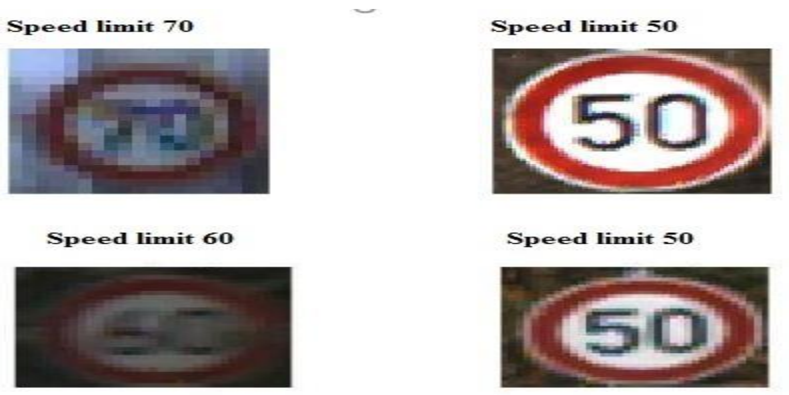


Figure 4.9 : Prediction by SVM at 0.6 drop

4.3.4 Training SVM by pretrained network with 0.65 drop out and 200 iterations

In this section, we used the trained network, that was trained on the 0.65 drop out and 200 iterations. Same as above it is used to extract the features for training the SVM. Then calculate the overall accuracy and error rate as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Over\ all\ accuracy = \frac{All\ \ True\ \ Positives}{Total\ sum\ of\ Confusion\ matrix}$$

$$= 0.973 = 97.3\ \%$$

$$Error\ \ rate = \frac{FN+FP}{TP+TN+FP+FN}$$

$$Over\ all\ error\ rate\ = 1 -\ Over\ all\ accuracy$$

$$= 1 - 0.973 = 0.027$$

$$= 2.7\ \%$$

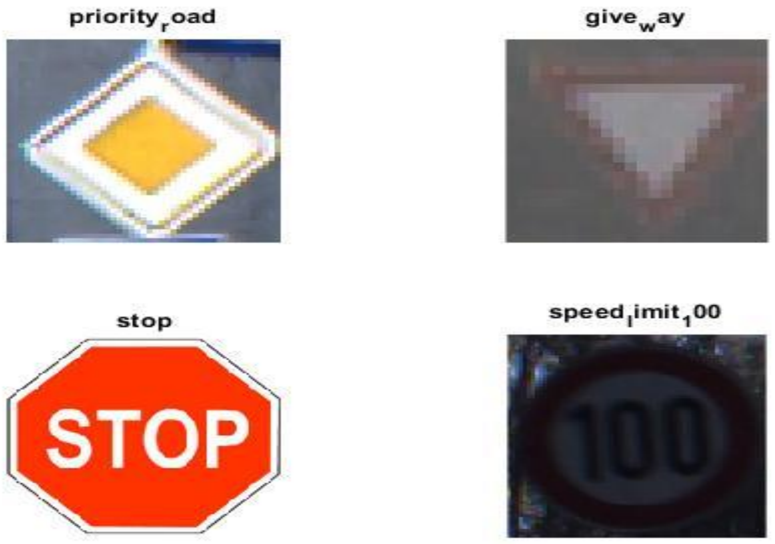


Figure 4.10 : prediction at 0.65 dropouts

Figure 4.10 shows that priority road, stop, speed limit 100 and give a way traffic signs are correctly classified. We observed that if we increase the dropouts, accuracy is slightly reduced, which is acceptable.

4.4 Summary

In this chapter, discussion of results on our proposed methodology are carried out. We observed that due to variation of dropouts, slightly reduction on accuracy is carried out that is easily compensated by increasing the dropouts. Remaining part of thesis, consist of conclusion and Bibliography.

Chapter 5

Conclusions and future work

In recent years, research in deep learning is growing extremely fast in various kind of applications like classification, regression etc. In past few years, development of deep learning in both academic and industry has increased and products which are related to deep learning appeared in market. Since more than 85 % of the data populated across the internet belongs to visual data, enormous and high-performance computing enabled processing and decision making on hand held devices using fog. It has been proven empirically in our research that altering the existing pretrained model of AlexNet performed quite well despite reduction in computation, achieving increase in dropouts. Auto driver cars are getting popularity, but they require high speed internet communication for uploading the video data on cloud and downloading the decisions performed on cloud. Since in our research we have classified traffic signs with a better accuracy. This research will surely add some value to this area. In near future, it is predicted that products number and their impact will increase enormously. Our assumptions are true because as we increase dropouts then we also increase iteration so that we compensate the overall efficiency of the classifier. So, our proposed method produced not only the efficient results but also reduced the network cost. In future optimization can be made on smallest filters to enhance the learning.

Bibliography

1. Stallkamp, Johannes, Marc Schlipsing, Jan Salmen, and Christian Igel. "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition." *Neural networks* (2012).
2. Houben, Sebastian, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. "Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark." In *Neural Networks (IJCNN), The 2013 International Joint Conference on* IEEE, 2013.
3. Cireşan, Dan, Ueli Meier, Jonathan Masci, and Jürgen Schmidhuber. "A committee of neural networks for traffic sign classification." In *Neural Networks (IJCNN), The 2011 International Joint Conference on* IEEE, 2011.
4. Sermanet, P. and Y. LeCun. *Traffic sign recognition with multi-scale convolutional networks*. in *Neural Networks (IJCNN), The 2011 International Joint Conference on*. 2011: IEEE.
5. Simon, M. and E. Rodner. *Neural activation constellations: Unsupervised part model discovery with convolutional networks*. in *Proceedings of the IEEE International Conference on Computer Vision*. 2015.
6. Simonyan, K., A. Vedaldi, and A. Zisserman, *Deep inside convolutional networks: Visualising image classification models and saliency maps.* arXiv preprint arXiv:1312.6034, 2013.
7. Fleet, David, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, eds. *Computer Vision--ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings*. Vol. 8689. Springer, 2014.
8. Goodfellow, I., Y. Bengio, and A. Courville, *Deep Learning.(2016).* Book in preparation for MIT Press. URL: http://www. deeplearningbook. org, 2016.
9. Bishop, C., *Bishop cm: Pattern recognition and machine learning. springer.* Journal of Electronic Imaging, 2006. **16**(4): p. 140-155.
10. Bengio, Y. *Deep learning of representations for unsupervised and transfer learning*. in *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*. 2012.
11. Rojas, R., *Neural Networks-A Systematic Introduction Springer-Verlag.* New York, 1996.
12. Long, L.N. and A. Gupta, *Scalable massively parallel artificial neural networks.* Journal of Aerospace Computing, Information, and Communication, 2008. **5**(1): p. 3-15.
13. Girshick, R., *Fast r-cnn.* arXiv preprint arXiv:1504.08083, 2015.
14. Szeliski, R., *Computer vision: algorithms and applications*. 2010: Springer Science & Business Media.
15. LeCun, Yann, Bernhard Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne Hubbard, and Lawrence D. Jackel. "Backpropagation applied to handwritten zip code recognition." *Neural computation* 1, no. 4 (1989): 541-551.
16. Fukushima, K., *Neocognitron: A hierarchical neural network capable of visual pattern recognition.* Neural networks, 1988. **1**(2): p. 119-130.
17. Hubel, D.H. and T.N. Wiesel, *Receptive fields and functional architecture of monkey striate cortex.* The Journal of physiology, 1968. **195**(1): p. 215-243.
18. LeCun, Y., K. Kavukcuoglu, and C. Farabet. *Convolutional networks and applications in vision*. in *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*. 2010: IEEE.

19. Goodfellow IJ, Warde-Farley D, Mirza M, Courville A, Bengio Y. Maxout networks. arXiv preprint arXiv:1302.4389. 2013 Feb 18.

20. Krizhevsky, A., I. Sutskever, and G.E. Hinton. *Imagenet classification with deep convolutional neural networks*. in *Advances in neural information processing systems*. 2012.

21. Yosinski, Jason, Jeff Clune, Yoshua Bengio, and Hod Lipson. "How transferable are features in deep neural networks?" In *Advances in neural information processing systems*, pp. 3320-3328. 2014.

22. Simonyan, K. and A. Zisserman, *Very deep convolutional networks for large-scale image recognition.* arXiv preprint arXiv:1409.1556, 2014.

23. Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: a simple way to prevent neural networks from overfitting." *The Journal of Machine Learning Research* 15, no. 1 (2014): 1929-1958.

24. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A. Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition 2015 (pp. 1-9).

25. Ioffe, S. and C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift.* arXiv preprint arXiv:1502.03167, 2015.

26. Escalera, Arturo de la, Luis Moreno, Miguel A. Salichs, and José M. Armingol. "Road traffic sign detection and classification." (1997).

27. Burges, C.J., *A tutorial on support vector machines for pattern recognition.* Data mining and knowledge discovery, 1998. **2**(2): p. 121-167.

28. Yu, H. *SVM selective sampling for ranking with application to data retrieval*. in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. 2005: ACM.

29. Herbrich, R., *Large margin rank boundaries for ordinal regression.* Advances in large margin classifiers, 2000: p. 115-132.

30. Yu, H. and S. Kim, *Svm tutorial—classification, regression and ranking*, in *Handbook of Natural computing*. 2012, Springer. p. 479-506.

31. Hastie, T. and R. Tibshirani. *Classification by pairwise coupling*. in *Advances in neural information processing systems*. 1998.

32. Friedman, J., *Another approach to polychotomous classification*. 1996, Technical report, Department of Statistics, Stanford University.

33. Hubel, D.H. and T.N. Wiesel, *Receptive fields, binocular interaction and functional architecture in the cat's visual cortex.* The Journal of physiology, 1962. **160**(1): p. 106-154.

34. Tang, Y., *Deep learning using linear support vector machines.* arXiv preprint arXiv:1306.0239, 2013.

35. Cortes, C. and V. Vapnik, *Support-vector networks.* Machine learning, 1995. **20**(3): p. 273-297.