Traffic Sign Detection

Khyathi Maddala
Computer Science and Engineering Dept.
Amrita University Amritapuri
Kollam, India
email : maddalakhyathi@gmail.com

Vishwanath M
Computer Science and Engineering Dept.
Amrita University Amritapuri
Kollam, India
email : vishwanathsai2001@gmail.com

Spandana Devisetty
Computer Science and Engineering Dept.
Amrita University Amritapuri
Kollam, India
email: spandana273@gmail.com

Sujanya Reddy Tirumala
Computer Science and Engineering Dept.
Amrita University Amritapuri
Kollam, India
email : reddysujanya@gmail.com

***Abstract*—For several years, much research has focused on the importance of traffic sign recognition systems, which have played a very important role in road safety. Researchers have exploited the techniques of machine learning, deep learning, and image processing to carry out their research successfully. The new and recent research on road sign classification and recognition systems is the result of the use of deep learning-based architectures such as the convolutional neural network (CNN) architectures. In this research work, the goal was to achieve a CNN model that is lightweight and easily implemented for an embedded application and with excellent classification accuracy. We choose to work with an Alexnet model for the classification of road signs. We trained our model network on the German Traffic Sign Recognition Benchmark (GTSRB) database.**

I. Introduction

Road safety is attracting the attention of many researchers around the world since it is indispensable in protecting human life. Driver assistance systems have played a very important role. For several years now, systems for the detection, classification, and recognition of road signs have become a very important research topic for researchers. From one research project to another, the authors have tried to improve the accuracy and recognition rate of these systems. To achieve these improvements, some researchers have turned to deep learning models.

Amongst these models that have been successful in the field of object detection and image classification are CNN. CNN's methods are similar to those of traditional supervised learning methods: they receive input images, detect the features of each of them, and then train a classifier on them. However, the features are learned automatically. The CNN do all the tedious work of feature extraction and description themselves: during the training phase, the classification error is minimized in order to optimize the classifier parameters and the features. In addition, the specific architecture of the network makes it possible to extract features of different complexities, from the simplest to the most sophisticated. Object detection and image classification are part of machine vision and machine learning problems. Road sign classification is a difficult task in machine vision since it requires a lot of computational effort and a correct, consistent, and accurate classification algorithm. CNN can solve such problems, thanks to the availability of their precise and simple architecture. Since 2010, these architectures were numerous thanks to the flagship computer vision competition ILSVRC (ImageNet Large Scale Visual Recognition Challenge) which aimed at correctly locating and classifying objects and scenes in images. For example, during this annual competition in 2012, the AlexNet architecture was created, which is a convolutional neural network. Since the creation of this architecture, it was invented in several image detection and classification tasks: face detection, facial emotion detection , garbage detection , counterfeit image detection, and localization .

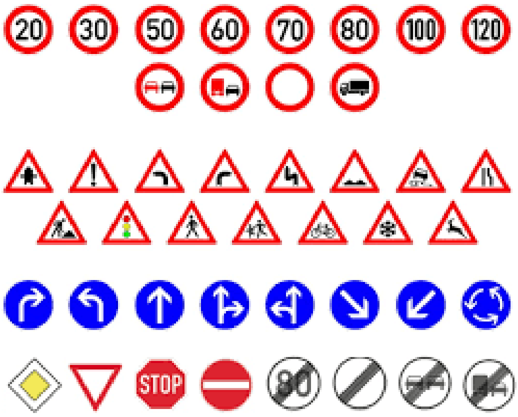


Fig. 1. Traffic signs

. Alexnet is a very famous architecture in the field of object detection and image classification. The traditional Alexnet consists of 11 layers including 5 convolutional layers, 3 subsampling layers, and 2 fully connected layer followed by an output layer. Due to the lightness of this network, the training

Identify applicable funding agency here. If none, delete this.

time is reduced, as well as the number of parameters, which makes the classification task easier for a machine.

II. Related Work

Recent research on road sign recognition systems uses deep learning models based on CNN. In order to recognize many classes of road signs, as, for example, in the GTSRB database which contains 43 classes, it is necessary to extract as many important features as possible from a road sign. CNN has the advantage of hidden feature extraction processing, allowing parallel processing through a parallel structure and real time operation . Another block of road sign classification and type recognition is through the CNN. It is used to judge whether the candidate region is a traffic sign and what type of sign it is. In another work, the use of CNN eliminated the manual work of feature extraction and provided resistance to spatial variations, the system was tested on GTSRB, and the accuracy rate using CNN reached 97.6 percent.

Another real-time embedded traffic sign recognition uses an efficient convolutional neural network for classification and a multiscale, deep-separable convolution operation for detection. This model has only 0.9 million parameters while achieving 98.6% of accuracy on GTSRB. What is more, in the latter paper, we tested their method using both traditional GoogleNet and Alexnet architectures. Based on the results (test time and test accuracy), we found that AlexNet was faster and more accurate than GoogleNet in this case.

III. Proposed Method

Recent research on traffic sign recognition systems has shown great interest in image quality and contrast. The use of image processing techniques improves the task of classification and the accuracy of the system.

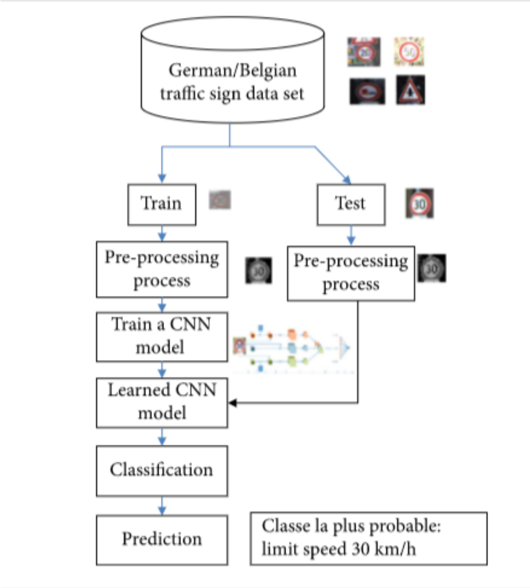


Fig. 2. Proposed Method

A. Preparing data for training

We defined a simple transformation that performs only two operations i.e., resize the images to 112 x 112 and convert them to PyTorch tensor. We initially split the training dataset into two: training and validation, with ratio 80:20. So I used 31392 images for training and the remaining 7849 images as validation samples. Then data loaders for both training and validation datasets were created. The distribution plot of both training and validation examples is shown in the figure. As can be observed, the distribution of validation examples closely follows that of the training set.

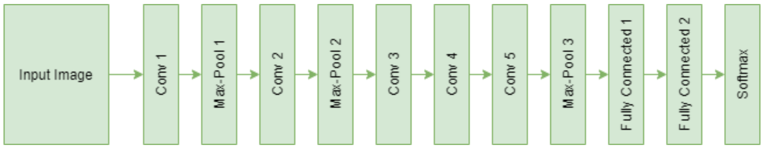


Fig. 3. AlexNet Architecture

B. CNN

Classification with artificial neural networks is a very popular approach to solve pattern recognition problems. A neural network is a mathematical model based on connected via each other neural units – artificial neurons – similarly to biological neural networks. Typically, neurons are organized in layers, and the connections are established between neurons from only adjacent layers. The input low-level feature vector is put into first layer and, moving from layer to layer, is transformed to the high-level features vector. The output layer neurons amount is equal to the number of classifying classes. Thus, the output vector is the vector of probabilities showing the possibility that the input vector belongs to a corresponding class. An artificial neuron implements the weighted adder, which output is described as follows where aij is jth neuron in the ith layer,

$$a_j^i = \sigma(\sum\nolimits_k a_k^{i-1} w^i j_k),$$

Fig. 4. weighted adder

wijk weight of a synapse, which connects the jth neuron in the ith layer with the kth neuron in the layer i-1. Widely used in regression, the logistic function is applied as an activation function. It is worth noting that the single artificial neuron performs the logistic regression function. The training process is to minimize the cost function with minimization methods based on the gradient decent also known as backpropagation. In classification problems, the most commonly used cost function is the cross entropy:

C. AlexNet

AlexNet was the first convolutional network which used GPU to boost performance.

$$H(p,q) = -\sum_{i} Y(i) \log y(i) .$$

Fig. 5. cross entropy

1. This architecture consists of 5 Convolutional layers, with the 1st, 2nd and 5th having Max-Pooling layers for proper feature extraction.

2. They are followed by 2 fully-connected layers (each with dropout) and a linear layer at the end for predictions.

3. AlexNet has overall 60 million parameters.

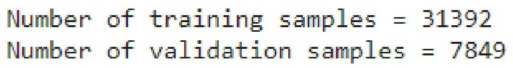
4. Operated with 3-channel images that were (112×112×3) in size.

5. Used ReLU activations in Convolutions.

6. Used (2×2) kernels for max pooling.

7. Used (3×3) kernels for convolutions.

8. Classified images into one of 43 classes.



```
Number of training samples = 31392
Number of validation samples = 7849
```

Fig. 6. Data

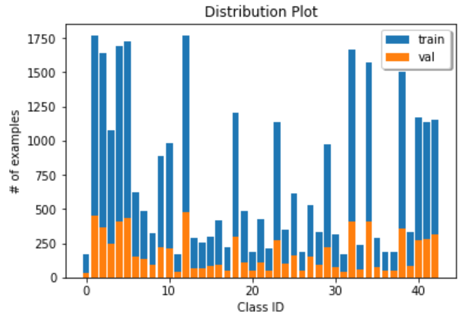
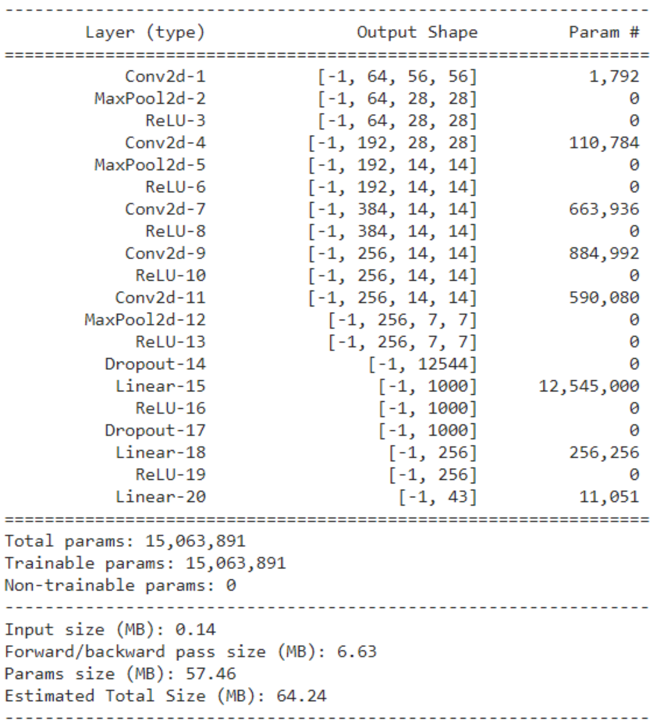


Fig. 7. Distribution plot of training and validation

D. Implementation

Performances	Accuracy
Epoch 5 [train/val]	93.5/92.4
Epoch 6 [train/val]	96.2/95.0
Epoch 7 [train/val]	98.94/97.78

TABLE I
PERFORMANCE



```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1           [-1, 64, 56, 56]           1,792
         MaxPool2d-2           [-1, 64, 28, 28]               0
              ReLU-3           [-1, 64, 28, 28]               0
            Conv2d-4          [-1, 192, 28, 28]         110,784
         MaxPool2d-5          [-1, 192, 14, 14]               0
              ReLU-6          [-1, 192, 14, 14]               0
            Conv2d-7          [-1, 384, 14, 14]         663,936
              ReLU-8          [-1, 384, 14, 14]               0
            Conv2d-9          [-1, 256, 14, 14]         884,992
             ReLU-10          [-1, 256, 14, 14]               0
           Conv2d-11          [-1, 256, 14, 14]         590,080
        MaxPool2d-12            [-1, 256, 7, 7]               0
             ReLU-13            [-1, 256, 7, 7]               0
          Dropout-14                [-1, 12544]               0
           Linear-15                 [-1, 1000]      12,545,000
             ReLU-16                 [-1, 1000]               0
          Dropout-17                 [-1, 1000]               0
           Linear-18                  [-1, 256]         256,256
             ReLU-19                  [-1, 256]               0
           Linear-20                   [-1, 43]          11,051
================================================================
Total params: 15,063,891
Trainable params: 15,063,891
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.14
Forward/backward pass size (MB): 6.63
Params size (MB): 57.46
Estimated Total Size (MB): 64.24
----------------------------------------------------------------
```

Fig. 8. Summary

CNN-training parameters	Value
Optimizer	Adam
Learning rate	0.001
Activation function	ReLU
Input image size	112x112x3

TABLE II
HYPER-PARAMETERS

IV. DATASET

- The German Traffic Sign Benchmark is a multi-class, single-image classification challenge held at the International Joint Conference on Neural Networks (IJCNN) 2011. Our benchmark has the following properties:
- The provided data are RGB traffic sign images of various sizes, cropped out of photos taken in real life scenarios.
- Single-image, multi-class classification problem
- More than 40 classes
- More than 50,000 images in total
- Large, lifelike database

V. RESULTS

GTSRB is a very famous international database of road signs. In order to train and test the model, some research projects used this database. GTSRB contains 43 kinds of road signs, training and test images taken under real conditions, as shown in Figure 5, a total of more than 50,000 images. The

CNN Architecture	Accuracy
AlexNet	97.99
GoogleNet	94.2%
VGG	94.9%

TABLE III
RESULTS

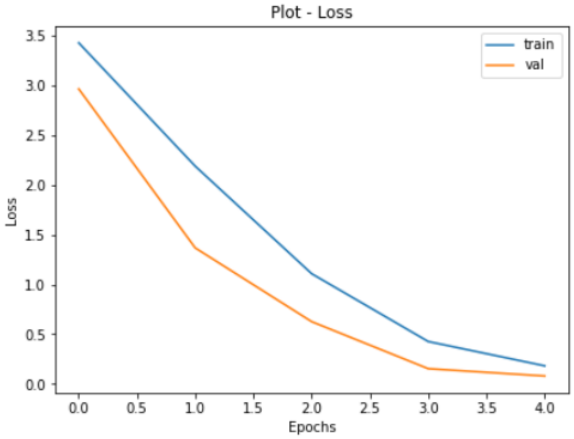


Fig. 9. Loss

database contains road sign images of different sizes, and the number of images in different categories is different, which will lead to the imbalance of the data set, thus affecting the accuracy of classification.

In our work, to avoid this problem, we use the data augmentation technique to increase the number of training images since the CNN becomes more efficient with a huge database. Also, by increasing the number of data, we will get a variation of exposures and more points of view for the same image which ensures better prediction. The data augmentation technique is applied on the training images. In order to train and evaluate our improved model, we made this distribution of data: 20 percent for the test and 80 percent for the training, and from this 80 percent of training, we chose 20 percent for the validation. The training data is used to train the model, the validation data allows us to supervise the performance of the model while training it (a reduced version of the test data), and the test data is used to evaluate the model.

In order to optimize the parameters, a loss function had to be set. These losses often measure the quadratic or absolute error between the output generated by the model and the desired output. In our work, we use cross-entropy which is designed specifically for classification problems. It minimizes the distance between two probability distributions: predicted and actual. As a glide slope technique, we use the Adam optimizer because of the good performance found when used with the ReLU activation function.

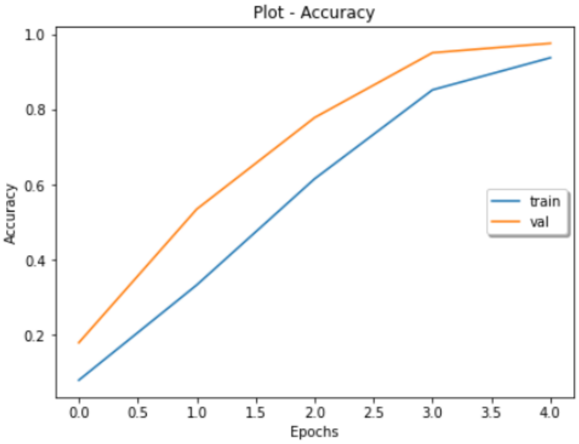


Fig. 10. Accuracy

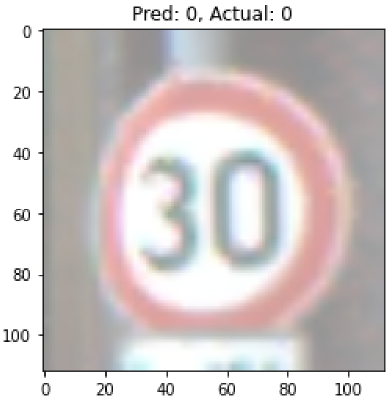


Fig. 11. Output Image

VI. CONCLUSION

The objective of road sign classification is to develop a system capable of automatically assigning a class to a road sign image. The applications of automatic classification of road signs are numerous. We obtained an accuracy of 97.99 percent and a reduced number of trained parameters compared to the depth of our model. Lightness allows us to try our model with an embedded application that uses the webcam. In this case, the classification is also very accurate.

VII. FEED FORWARD NEURAL NETWORKS

A. Image Pre-Processing

- For image preprocessing, we here converted the RGB images to Grayscale images. The purpose of converting is to extract the features in more efficient manner.
- RGB color space is sensitive to light changing. The process of converting RGB to Grayscale is shown in the following equations: The simple average method i.e.

$$Grayscale = (R+G+B)/3 = R/3+G/3+B/3 \tag{1}$$

doesn't work soo well as expected because human eyeballs have different sensitivity to different lights,so they should have different weights in the distribution.

- The weighted method:

$$Grayscale = 0.299R + 0.587G + 0.114B \qquad (2)$$

This approach is used by MATLAB,Pillow and Open CV. In our model we used Open CV.

- After converting into Grayscale images we have resized all the images to 32*32(Height*Width). The deep learning model architecture require that our images are the same size and our raw collected images may vary in size.

B. Architecture

Feed-forward neural networks are the most popular and most widely used models in many practical applications. They are known by many different names, such as 'multilayer perceptrons' (MLP). A feed-forward neural network is a biologically inspired classification algorithm. It consists of a number of simple neuron-like processing units, organized in layers and every unit in a layer is connected with all the units in the previous layer. These connections are not all equal, as each connection may have a different strength or weight. The weights on these connections encode the knowledge of a network. Often the units in a neural network are called nodes.

Data enters at the input and passes through the network, layer by layer, until it arrives at the output. The input layer consists of just the inputs to the network. Then follows a hidden layer, which consists of any number of neurons, or hidden units placed in parallel. Each neuron performs a weighted summation of the inputs , which then passes a transfer/activation function, also called the neuron function. During normal operation there is no feedback between layers. article [utf8]inputenc graphicx

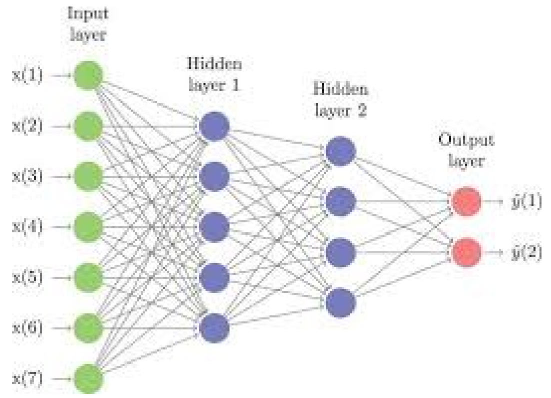


Fig. 12. Architecture

C. Training

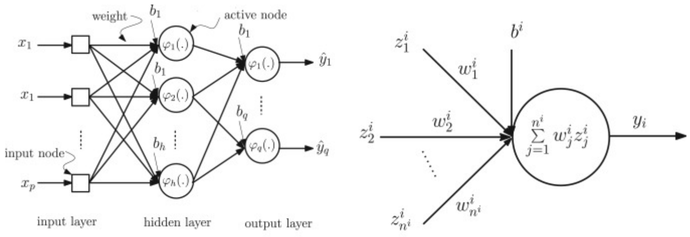
The networks was trained using the Gradient Descent optimization algorithm (eq 4) with an initial learning rate of 0.1, Batch size should be preferably greater than the number of class (43) and hence we choose 100 as the batch size. We use the cross-entropy (eq 3) loss as the objective function. Entropy H(x) can be calculated for a random variable with a set of x in X discrete states discrete states and their probability P(x) as follows:

$$H(X) = -sumxinXP(x)log(P(x) \qquad (3)$$

At every iteration we update our model parameters:

$$params = paramslearningrateparamsgradients \qquad (4)$$

Learning rate determines how fast the algorithm learns. Too small and the algorithm learns too slowly, too large and the algorithm learns too fast resulting in instabilities.



(a) Three-layer feedforward neural network

(b) Node of the network

Fig. 13. FeedForwardNetwork

D. Results

In training process there are more than 70 epochs. The accuracy with Sigmoid activation function is 62 percent. The accuracy with tanh activation function is 63 percent. The accuracy with ReLu activation function is 65 percent. The accurcy with LeakyRelu activation function is 68 percent.

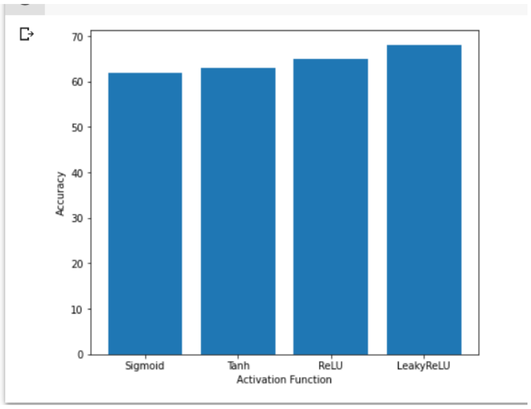


Fig. 14. Accuracy Comparison

E. Conclusion

We started with the simple model architecture using Sigmoid activation function which had a moderate fit on the data. So we moved on to the next activation function tanh, which gave a better fit compared to sigmoid. Then we tried to increase the accuracy with ReLu activation function. At last, the best fit for the model which gave highest accuracy and better performance compared with all other models is LeakyReLu.