# Road Sign Classification using Deep Learning

**Ayush Dhar**
**Virginia Tech**
**Blacksburg, Virginia**
adhar@vt.edu

**Darshit Patel**
**Virginia Tech**
**Blacksburg, Virginia**
darsh2198@vt.edu

**Zihao Cai**
**Virginia Tech**
**Blacksburg, Virginia**
zihao225@vt.edu

## Abstract

One of the key aspects for safe driving of autonomous vehicles (AVs) is detecting correct road signs. In this project, we are reporting the use of established Neural Networks, called AlexNet and GoogLeNet which are CNN models for the road sign classification problem. The problem is a single-image multi-class classification problem and hence the German Traffic Signs Recognition Benchmark (GTSRB) dataset was used for training and evaluation of the model. An accuracy of 96.84% was obtained for AlexNet, while an accuracy of 95.94% was obtained for GoogLeNet on the test data.

## 1 Introduction

Reactive Motion Planning for AVs requires very complex algorithms and high computational power to safely complete a journey. Active research is going on for a complete vision-based autonomous driving system which increases the computational load on the vision system. Hence it becomes extremely important to make different modules, like road-sign classification, less computationally expensive while maintaining high accuracy. A lot of research is going on to obtain a highly accurate light-weight Neural-Network for such multi-class classification problems. The GTSRB dataset is widely used for the evaluation in research [1]. Saadna Behloul reviewed and compared different models used for this problem of classification and reported the accuracies and efficiencies of them [2]. This project assesses the performance of AlexNet and GoogLeNet with required modifications on the GTSRB dataset.

## 2 Methodology

### 2.1 Dataset and Pre-Processing

The German Traffic Sign Recognition Benchmark (GTSRB) dataset was used for training the Convolutional Neural Network. The GTSRB dataset was created to evaluate the performance of state-of-the-art machine learning algorithms for classification problems [1]. The dataset consists of 39209 training images and 12630 test images. The images are classified into 43 different categories and each category consists of 200-3000 images in it including prohibitory signs, danger signs, and mandatory signs.. All the images are of different sizes ranging from 25x25x3 up to 266x232x3. The sample images for each class are shown on the left in Figure 1.

The traffic signs in this dataset are captured from various perspectives, under different lighting conditions, different shadows, color damages, and so on. This makes the detection problem more difficult and the human accuracy reported for the classification of the test set was 98.84%[1]. This indicates that the dataset consists of images that are difficult for even humans to classify. Image on the right in Figure 1 shows an example of image where the lighting was not very good. Usually a CNN model requires all the images that it is going to be trained on be of the same size. Hence, all the

Preprint. Work in Progress.

Figure 1: Sample of images from the dataset

images were resized to 32x32x3 before feeding them to the AlexNet and GoogLeNet models. This image size was selected to convert the images to 3-D arrays faster and reduce the training time.

### 2.2 Architecture

#### AlexNet

AlexNet is the first popularized CNN architecture in computer vision developed by Alex Krizhevsky, Geoffrey Hinton, and Ilya Sutskever.[3] It is the champion of ImageNet ILSVRC challenge in 2012 and significantly outperformed the second runner-up. The AlexNet has a similar architecture to LeNet, which was proposed by Yann LeCun in 1998, but it is deeper and bigger.

The original AlexNet architecture used its first convolutional layer filters to be the 224×224×3 input image which for our purposes we changed to be 32x32x3 instead to work better with the GTSRB dataset selected by us but with the same 96 kernels of size $5 \times 5$.

The second convolutional layer takes as input the (response-normalized and pooled) output of the first convolutional layer and filters it with 256 kernels of size $3 \times 3$. The third, fourth, and fifth convolutional layers are connected to one another without any intervening pooling or normalization layers. The third convolutional layer has 384 kernels of size $3 \times 3$ connected to the (normalized, pooled) outputs of the second convolutional layer.

The fourth convolutional layer has 384 kernels of size $3 \times 3$ , and the fifth convolutional layer has 256 kernels of size $3 \times 3$. The output of this layer was normalized by a pool size of $2 \times 2$ and a stride of 2 pixels and subsequently flattened to a vector form.

After a couple of dense layers with intermediate dropouts the output of the last fully-connected layer is fed to a 43-way softmax dense layer which produces a distribution over the 43 class labels.

Figure 2 shows the architecture of the AlexNet we used in this project.

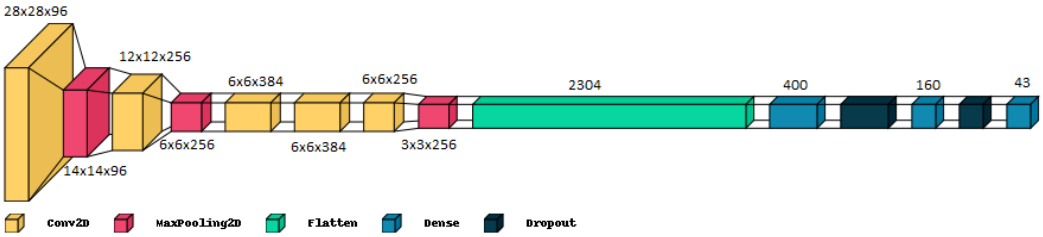


Figure 2: Illustration of the AlexNet architecture that was implemented

**GoogLeNet**

GoogLeNet was first developed by C. Szegedy et al. [4] as a part of the ImageNet ILSVRC challenge in 2014 and it eventually won the challenge. That architecture outperformed the previous best architecture, AlexNet with better accuracy and fewer parameters.

In order to prevent overfitting, the number of parametric layers were reduced from 22 to 14. GoogLeNet is one of the inception networks where inception modules are used. Inception modules consist of six convolutional layers and a max pooling layer connected as shown in Figure 3. The kernel size for conv2D layers f1, f2, f4, and f6 is $1 \times 1$, for layer f3 is $3 \times 3$, and for layer f5 is $5 \times 5$. The depth of these convolutional layers were different for different inception modules and are summarised in table 1. All the convolution layers used ReLu as the activation function. These layers are connected by a concatenate layer which is connected to the next layer/module.

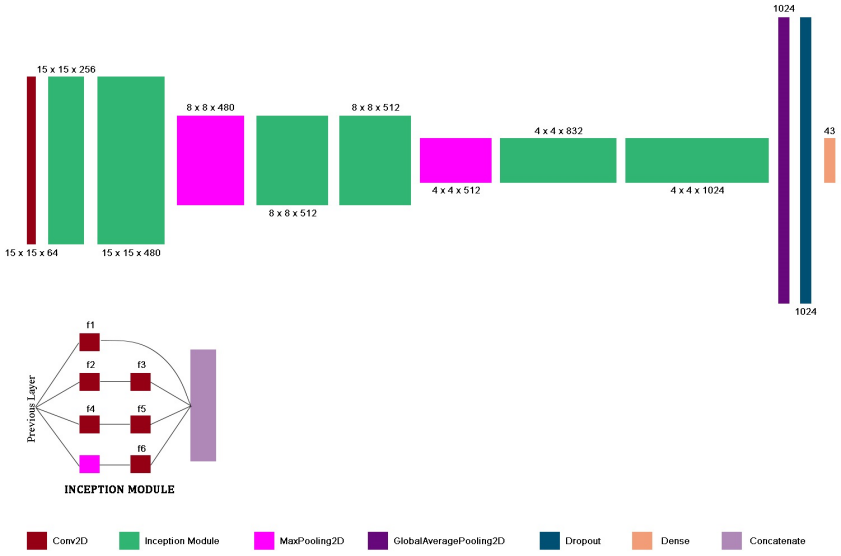


Figure 3: Illustration of the GoogLeNet architecture that was implemented

The original architecture used nine inception modules, while our model uses six inception modules. $32 \times 32 \times 3$ sized image is the input to the first conv2D layer with 64 kernels of size $3 \times 3$ and a stride of 2 pixels. The output of this layer is fed to the first inception block. The ouput of first inception block is directly fed to the second inception block without any pooling. The output of this block is normalized by a pool size of $3 \times 3$ and a stride of 2 pixels.

The next two inception blocks are again connected without any pooling and the output of the 4th inception block is again normalized by a pool size of $3 \times 3$ and a stride of 2 pixels. The feature maps obtained after the 6th inception block were vectorized using a GlobalAveragePooling layer to get a 1024 sized vector. After passing through the dropout layer, the output is fed to a 43-way softmax dense layer which produces a distribution over the 43 class labels similar to the AlexNet architecture.

Figure 3 shows the architecture of the inception network we used in this project.

Table 1: Number of kernels for Conv2D layers in the inception modules.

| **Inception Module** | **f1** | **f2** | **f3** | **f4** | **f5** | **f6** |
|---|---|---|---|---|---|---|
| IM-1 | 64 | 96 | 128 | 16 | 32 | 32 |
| IM-2 | 128 | 128 | 192 | 32 | 96 | 64 |
| IM-3 | 192 | 96 | 208 | 16 | 48 | 64 |
| IM-4 | 162 | 112 | 224 | 24 | 64 | 64 |
| IM-5 | 256 | 160 | 320 | 32 | 128 | 128 |
| IM-6 | 384 | 192 | 384 | 48 | 128 | 128 |

In order to implement, train and evaluate the models we used python along the keras, numpy and sci-kit learn library. The AlexNet was trained using the Adam optimizer with a learning rate of 0.001 over 50 epochs while the GoogLeNet was also trained using the Adam optimizer but with a learning rate of 0.0001 over 50 epochs.

## 3 Evaluation and Results

We trained the AlexNet model and GoogLeNet model with **31367** traffic sign training images and **7842** traffic sign validating images from GTSRB dataset. The models were evaluated using **12630** testing images to obtain the accuracy of the the testing data. The models were compared using the F-1 score for each architecture. The F1-Score is the norm of the classifier model to compare it with other models. F1-Score takes 1.0 as the best score and 0.0 as the worst score as the performance of the model during the computation.

### 3.1 Performance of AlexNet

During the training process, the accuracy started converging very quickly at the 4th epoch, and the loss of the model also started decreasing immediately after 4th epoch. At the 1st epoch, the training accuracy was 23.38% and loss is 2.6090. At the 2nd epoch, the accuracy increased to 62.84% while the loss decreased to the 1.0924. After the 4th epoch with training accuracy of 91.6%. the accuracy gradually converged to the 99% training accuracy and the loss converged to 0.02 by the end of 50th epoch.

On the test dataset, the accuracy of the AlexNet model came out to be **96.84%** with an F-1 score of **0.97** on the classification of the traffic signs.

Table 2: Classification Report of AlextNet Accuracy on Test Data.

| | **Precision** | **Recall** | **F1-Score** | **Support** |
|---|---|---|---|---|
| Accuracy | - | - | 0.97 | 12630 |
| marco avg | 0.95 | 0.96 | 0.95 | 12630 |
| weighted avg | 0.97 | 0.97 | 0.97 | 12630 |

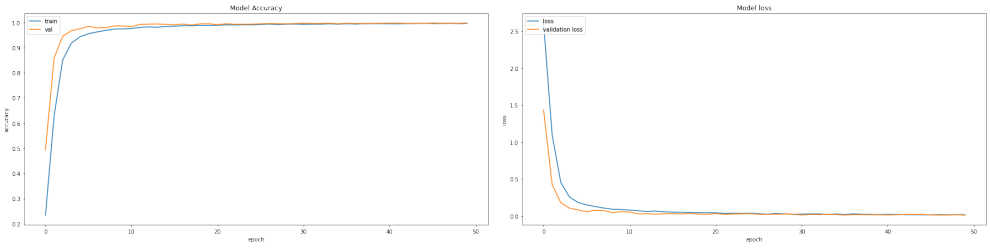


Figure 4: Model Accuracy(left) and Model Loss(right) of the AlexNet on Training data

Table 3: Classification Report of GoogLeNet Accuracy on Test Data.

| | **Precision** | **Recall** | **F1-Score** | **Support** |
|---|---|---|---|---|
| Accuracy | - | - | 0.96 | 12630 |
| marco avg | 0.94 | 0.94 | 0.94 | 12630 |
| weighted avg | 0.96 | 0.96 | 0.96 | 12630 |

### 3.2 Performance of GoogLeNet

The rate of convergence for GoogLeNet was slower as compared to the AlexNet architecture as observed through figure 5. At the 1st epoch, a training accuracy of **56.49%** and a loss of **1.2891** was observed. The training started converging at the 17th epoch, with the training accuracy finally converging to **97%** and the loss to **0.12**.

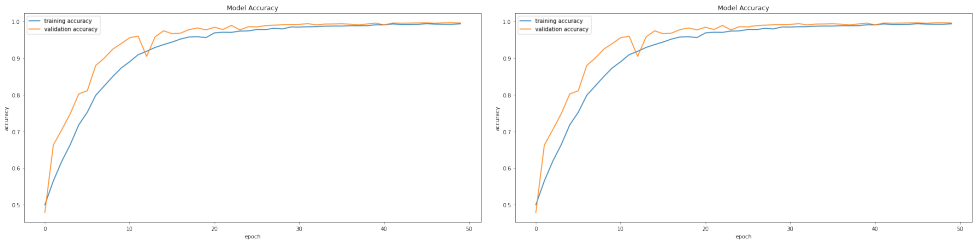


Figure 5: Model Accuracy(left) and Model Loss(right) of the GoogLeNet on Training data

On the test dataset, the accuracy of the AlexNet model came out to be **95.94%** with an F-1 score of **0.96** on the classification of the traffic signs.

Figure 6: Sample of nine predictions made by AlexNet(left) and GoogLeNet(right)

Figure 6 shows the predictions made by both the models for a small sample of test images. It can be seen that both the models predict fairly accurate outputs even with bad lighting and distorted pixels. Due to low resolution images for some signs, the classifiers were not able to distinguish between few speed limits. Even for a human eye, those images are a bit ambiguous.

## 4 Conclusion

Both the models were trained using the same Adam Optimizer but with different learning rates. The AlexNet architecture performed slightly better than the GoogLeNet. The difference of the F1-Score between those two models is 0.1, and both of them almost reach the best score. Considering of the errors during the computations, those two models almost have same performance on classifying traffic sign images. Even with a smaller learning rate the GoogLeNet architecture has comparatively lower accuracy than the AlexNet model.

## 5 Contributions

**Ayush Dhar**: Worked on implementing the Neural Networks along with training and helping evaluate the networks.
**Darshit Patel**: Worked on dataset acquisition and pre-processing the dataset along with training the networks.
**Zihao Cai**: Worked on evaluation and comparison of the proposed architectures.

## 6 References

[1] J. Stallkamp, M. Schlipsing, J. Salmen, C. Igel, Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition, Neural Networks, Available online 20 February 2012, ISSN 0893-6080, 10.1016/j.neunet.2012.02.016

[2] Saadna, Y., Behloul, A. An overview of traffic sign detection and classification methods. Int J Multimed Info Retr 6, 193–210 (2017). https://doi.org/10.1007/s13735-017-0129-8

[3] Krizhevsky, Alex, et al. "ImageNet Classification with Deep Convolutional Neural Networks." Communications of the ACM, vol. 60, no. 6, 2017, pp. 84–90. Crossref, https://doi.org/10.1145/3065386.

[4] C. Szegedy et al., "Going deeper with convolutions," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1-9, doi: 10.1109/CVPR.2015.7298594.