COMP9444 Deep Learning 22T3 Group Report

**z5141973 Rachel Xie, z5285206 Zhao Huang, z5271698 Zhaoyan Li, z5272579 Peng Zhang, z5311209 Celine Lin**

# Introduction

In the modern world, driving is one of the most essential modes of transportation. In order to prevent drivers from disobeying traffic signs, avoid dangers, and promote safer driving, this project focuses on developing models that employ forward-sensing cameras to recognise traffic signals. Data exploration was our first step, followed by analysis of the image dataset and data pre-processing. We started with 5 models for deep learning, all of which were trained using the same dataset.

# Literature Review

CNNs are a class of deep neural networks that can recognize and classify features from images, which allows us to extract a higher representation of the content of an image. It is widely used in the field of computer vision. Unlike classical image recognition, a CNN defines its own image features, it takes the raw pixel data of the image, trains the model and then automatically extracts the features for better classification[1],. CNN is also very effective at reducing the number of parameters without losing model quality. Images often have high dimensionality, which is consistent with the capabilities described by CNNs.

# Models and Experimental Setup

For this project, we will use this GTSRB - German Traffic Sign Recognition Benchmark dataset. This is because it is large, lifelike database and contains 43 classes of images and more than 50,000 images. In addition, the images are in RGB with 30x30 pixels, and the test folder contains 12.6k images. What's more, the size of images is between 30 x 25 and 266 x 232.

To process the data, we split the training dataset 8:2 into a training set and a validation set. For the training set, we extracted the bounding box coordinates and categories from all the train files. And the test set is the given folder containing the images for further testing.

The following section introduces the trained five models with their changes in layers and parameters. Their results are then combined into Ensembled model for higher accuracy.

## A. AlexNet

AlexNet was originally designed by Alex Krizhevsky. It was published by Ilya Sutskever and Geoffrey Hinton, PhD Advisor to Krizhevsky, and is a convolutional neural Network or CNN[2],. AlexNet contains eight layers; the first five are convolutional layers, some of them follow by max-pooling layers, and the last three are fully connected layers. It uses the non-saturating (Rectified Linear Unit) activation function, which shows improved training performance over tanh and sigmoid.

For the pre-processing, the input images are resized to 30 x 30 and center cropped. In addition, the rotation method is used to input images. Image matrix values are normalized to ensure optimal comparison between data acquisition methods and texture instances, making the model less sensitive to the proportions of features and able to avoid gradient explosion/disappearance.

AlexNet applies Max-pooling after the first, second, and fifth convolutional layers. The kernels of the second, fourth, and fifth convolutional layers are connected only to those kernel maps in the previous layer, which reside on the same GPU. The kernels of the third convolutional layer are connected to all kernel maps in the second layer. The neurons in the fully connected layers are connected to all neurons in the previous layer. Dropout is applied in the first two fully connected layers. Regarding the key model parameters. For the model parameters, the input image size is 30x30x3, and the output is 43 classes. The first convolutional layer uses 11 x 11 kernel size and 4 x 4 strides. The second layer uses 5 x 5 kernel size and 1 x 1 strides. The third, fourth and fifth layers use the same size kernel with 3 x 3 and 1 x 1 strides. The last three layers are connected layers which use the dropout method to avoid overfitting. In addition, the evaluation metrics of AlexNet is “accuracy”.

## B. Generic CNN

In our model, the structure starts with 2 consecutive convolutional layers to extract and learn the features and pass them to the pooling layer, replacing the output with the maximum summary using the maximum pool to reduce the data size and processing time. The above structure is then repeated, followed by a convolutional layer, a pooling layer and a normalization layer for stable training, the corresponding results are flattened, dropout and normalization to effectively prevent overfitting of the training data, and the results are passed to the fully connected layer. And the final output layer will produce 43 classified results.

Regarding the key model parameters, for the generic CNN model we will test with different values of filter, kernel size and pool size, allowing zero padding and all dense layers will have 'Relu' activation. Higher accuracy is guaranteed by increasing or decreasing the dropout and normalization layer.

## C. Deep CNN

## This model is built based on the basic CNN model presented by another group member in file "cnn.ipynb". It is chosen to further improve the training and test accuracy by increasing the convolutional layer size and kernel size.

Increasing these two parameters may result in overfitting and long training time problems, which were solved by adding BatchNormalization layer and Dropout layer after every two Convolutional layers in the model. As for hyperparameters, "l1\_l2" applies both l1 and l2 regularization penalties to reduce the weights. Activation of the first Dense layer is chosen as "Relu" to speed up the training process and to achieve better performance for its quick convergence and more strict saturation requirement compared to "sigmoid" activation. Then "softmax" activation is used in the second Dense layer as an output layer to compute the probability for this multiclass problem.

Moreover, input images are resized and centre-cropped to smaller images to reduce the training time, and they are randomly rotated by 10 degrees to increase the semantic coverage of the dataset. The image matrix values are then normalized to ensure optimal comparisons across data acquisition methods and texture instances, making the model less sensitive to scales of features, and able to avoid exploding/vanishing gradients. Adam is chosen as the optimizer, combining the strengths of both Momentum and Root Mean Square Propagation, to provide optimized gradient descent for its faster computation time and fewer parameters for tuning.

As for the training process, callbacks are used to reduce the learning rate and execute early stopping to apply dynamic adjustment during training. The learning rate is set to "1e-3", so that it is not too small to extend training time and not too large, resulting in undesirable divergent behaviour in the loss function.

With these changes and optimizations, the model is expected to achieve higher test accuracy than the generic CNN model, while resolving the overfitting problem.

## D. Efficient CNN

This model follows the idea of EfficientNet, which is a type of Convolutional Neural Network, where all dimensions of depth/width/resolution increase to achieve higher accuracy.

To simulate this net, six convolutional layers are added to increase the depth of the model. In each convolutional layer, the width increases by changing the filter, and in order to increase the nonlinear relationship between the layers in the model, we set the activation to ReLU. However, the lack of memory storage cannot increase image resolution. Hence, by comparing to the general EfficientNet, this model only follows the idea of increasing depth and width.

GlobalAveragePooling2D is also added to decrease the dimensions of the feature maps at the end to achieve higher accuracy. As the model is deep, 2d Dropout is used after every convolutional layer, by testing the best parameter for dropout is 0.5, and accelerates training using Batchnormalization to prevent overfitting. Finally, after softmax processing, we will output 43 classified results.

## E. LeNet

This model was built based on LeNet which was one of the first released CNNs with wide applications and influence.

The traditional LeNet consists of two sections: the first is a convolutional section that consists of two convolutional layers and the second is a dense layer section that consists of three fully connected layers [4].

In order to improve the accuracy of model training, the number of convolutional layers is increased from 2 to 3, and only one max pooling is kept, and also, in order to increase the nonlinear relationship between the layers in the model, the activation is set to ReLU.

After being Flattened, the result will be passed to fully connected layers consisting of three Dense Layers, and between every two Dense Layers we added a dropout to prevent overfitting of the training data and the best parameter for dropout is 0.2 and 0.5. Finally, after softmax processing, we will output 43 classified results

## F. Ensembled Model

With the testing accuracies of the five models above, the top three models are chosen to be ensembled to achieve a higher test accuracy by combining their strengths.

If the results from the top three models are all different, the ensembled model finds the model that has the highest test accuracy and chooses its result as the output. Otherwise, if there is a result with maximum votes, the ensembled model uses that result as the output.

By setting the default result and the combined result, this ensembled model is expected to achieve the highest test accuracy among all the models.

# Results

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1. Model | Generic CNN | Deep CNN | AlexNet | Efficient CNN | LeNet | Ensembled Model |
| Accuracy | 89% | 92.2% | 79% | 84% | 87% | 93.5% |

Table1. Overall results of all models

The findings were attained after attempting the five approaches outlined above. We compile the data findings from each model into a table, and it is clear that the ensembled model outperformed the others. We discovered that all the models performed better in terms of accuracy, with Ensembled model producing a greater accuracy—93.5%. And we discovered that the fastest to train was generic CNN. These factors influenced our decision to settle on the ensembled model as the ideal one.

## A. AlexNet

After running 70 epochs, the loss and accuracy rate gradually become stable. And the final training accuracy can reach 93.8 % and the test accuracy can reach 79.2%.

## B. Generic CNN

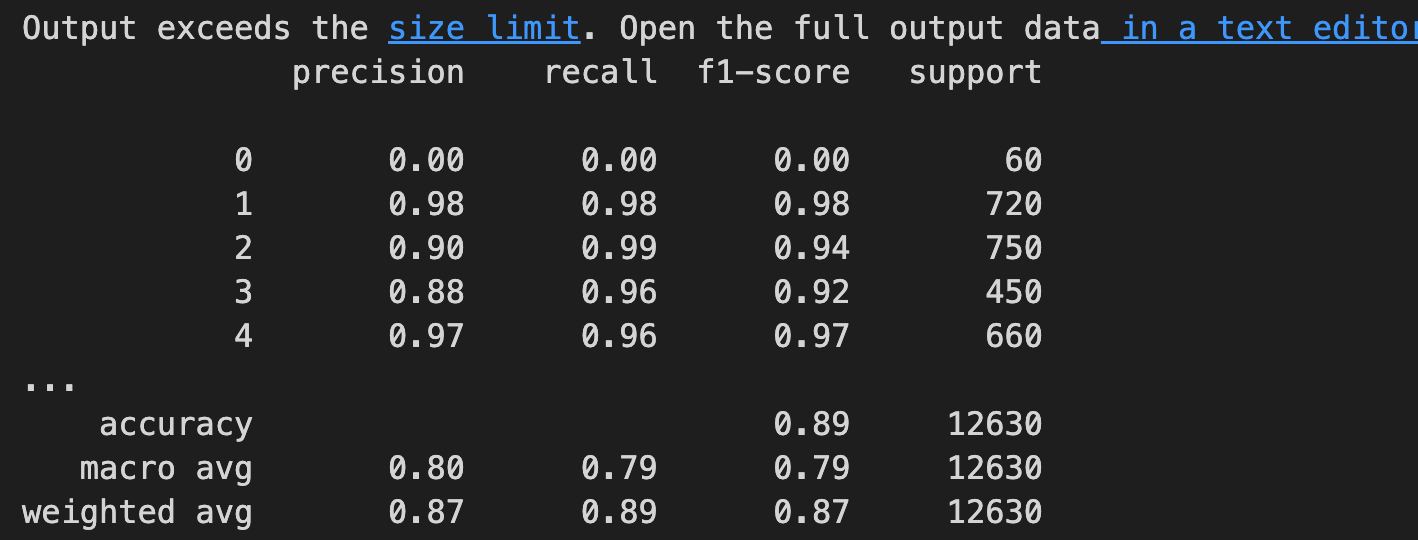


Figure 1: test accurancy of CNN

The test accuracy of the CNN model is 89% which is a good result, but not nearly good enough. However, the model has the advantage compare with others that the time required for training is exceptionally short, requiring only 10 epoches to converge.

## C. Deep CNN

Deep CNN model obtains the highest test accuracy as "0.922", while basic CNN obtains "0.88" and EfficientCNN obtains "0.84". The comparison of training epochs and accuracy is shown in the figure below.

Text

Description automatically generated with low confidence

Figure 2: training epochs and accuracy comparison

Although Deep CNN takes the longest training time, it reaches the highest test accuracy score among these three. This is because it has a deeper convolutional layer, which has more filters and a larger kernel size, helping extract and analyse more features. Moreover, the difference between test and train accuracy is the smallest, which indicates the added Dropout and BatchNormalization layer, randomly rotated input images and the optimizer have successfully reduced the overfitting effect.

However, a DNN solution achieves a higher test accuracy with less training time from site "https://www.kaggle.com/code/whitelord/german-sign-detection-dnn-solution". This model has less training time for it has a slightly smaller filter size as 32 instead of 64. The other main difference is that it uses one hot encoding instead of label encoding, which prevents the ranking between values and increases the train and test accuracy.

## D. Efficient CNN

The result demonstrates that the model has a relatively high accuracy, with 84% of training accuracy. Since this model is deep and wide, the disadvantage of this is that the training time would be longer than simple models. However, this model can train more complex resources for future applications.

## E. LeNet

After customization and optimization, the result of our LeNet is improved comparing to original LeNet. From the generated statistical chart and the program output, after 146 Epoch training, the Training and Validation Accuracy finally converges to 96%, and the Test Accuracy is 87%. Compared with the traditional LeNet (2 Convolutional Layer version), the customized model has a higher accuracy rate, and the final convergence of the validation accuracy and the training accuracy is closer.

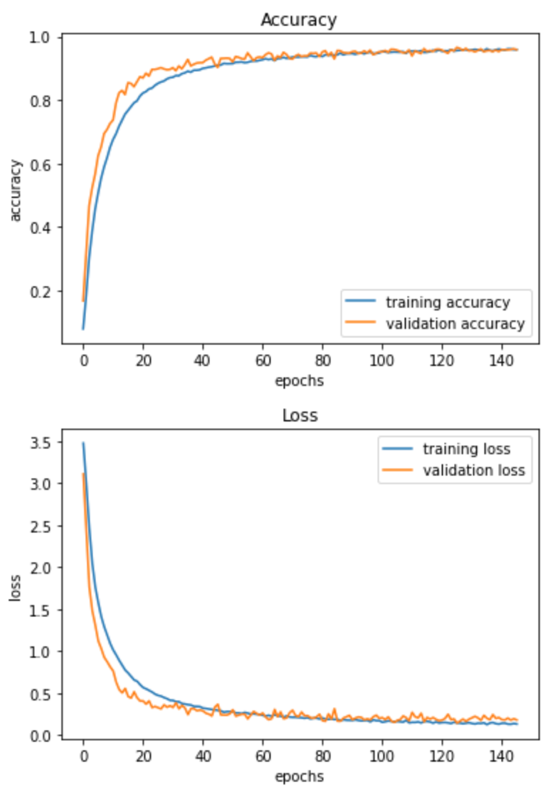


Figure 3: LeNet accuracy and loss figures

## F. Ensembled Model

The test accuracy of the ensembled model is the highest "0.935". This is because it uses the result of the model that has the highest test accuracy as default and improves the result by considering other models' outputs.

# Conclusion

The proposed solution is the ensembled model. This model ensembles the results of provided model group and can achieve the highest test accuracy among these models. The code of this model is well structured so that it will not be limited by the number of input models, and the input model has flexible functions and self-parameters to change the layers or do the customization.

However, the disadvantage of this model is that it needs to train all the models and store their results together to get the final result. This can draw back the training efficiency and become memory costing if the dataset is large.

The future improvement can be constructing a linear layer to simultaneously train the weights of each model's output so that the results of each model do not need to be stored and the training time can be reduced.

As for the preprocessing of all the models, there is one common disadvantage of resizing the image when there are images with larger sizes, like 90, which can lose lots of important information after resizing and cropping to size 30. This can be further improved by resizing the images to a larger size and ignoring the black paddings for smaller images.

# References

[1] Tatan, V. (2019, December 24). Understanding CNN (Convolutional Neural Network)Vincent Tatan. Retrieved November 19, 2022, from <https://towardsdatascience.com/understanding-cnn-convolutional-neural-network-69fd626ee7d4>

[2] Team, G. (2022, October 31). AlexNet: The first CNN to win Image net. Retrieved November 19, 2022, from <https://www.mygreatlearning.com/blog/alexnet-the-first-cnn-to-win-image-net/>

[3] EfficientNet: Rethinking Model Scaling forConvolutional Neural Networks<https://arxiv.org/abs/1905.11946v5>

[4] Dive into Deep Learning. (2022, July). Dive into Deep Learning. Retrieved November 19, 2022, from <http://d2l.ai/chapter_convolutional-neural-networks/lenet.html>