COMP30027 Assignment 2 Report

Semester 1 2025

Word count: 2,733

1. Introduction

This report outlines the process of training and evaluating machine learning models tasked with classifying 43 distinct German road signs from the images provided in the GTSRB dataset[1] (see Figure 1.1 for some examples).

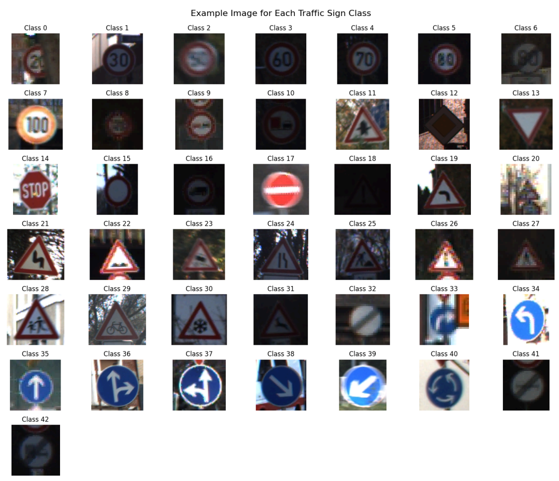

Figure 1.1 - Examples of an image from each class in the GTSRB dataset.

The GTSRB dataset consists of 5,488 labelled training images and 2,353 unlabelled test images.

We pre-processed, extracted features from, normalised and split this data into validation and train sets to assess the overall accuracy as well as the per-class F1 scores of each of the following learners:

- Logistic Regression
- Support Vector Machine (SVM)
- K-Nearest Neighbour (KNN) classifier
- Stacking Classifier
- Convolutional Neural Network (CNN)

We found that although simple learners like SVM, KNN and Logistic Regression are efficient, reliable and relatively accurate, they are limited in their accuracy potential. This is due to the inherent complexity of image classification, especially when classes are differentiated by small visual artefacts.

In recognition of this complexity, the use of a Convolutional Neural Network (CNN) is justified. The CNN manages to faithfully learn the distribution of the images and generalise successfully to unseen images with near perfect accuracy. All without the arbitrary process of feature - extraction and - engineering. Given these results, the CNN is the preferred learner for this task due to its robustness and consistent accuracy.

2. Methodology

Our approach to the classification task consists of three main stages: feature engineering, model selection and evaluation.

First, we performed exploratory data analysis to investigate our dataset by checking for missing values, inspecting image quality, and analysing the class distribution to check for imbalance.

Next, we processed the images to extract our own features, instead of using existing features from the dataset. This helped us gain a deeper understanding of how different processing steps affect the data, which allowed us to maximise the amount of relevant information captured in the features.

Using the provided features as a reference, we computed Histogram of Oriented Gradients (HOG) with PCA, binned HSV values, edge densities (ED), colour variance and skew, and Local Binary Patterns (LBP) and shape features (contours, Hu Moments and Convex Hull). These aim to capture visual properties such as colour, shape, and patterns, which are all important for distinguishing between different types of traffic signs.

To ensure reasonable training time and reduce noise, we limited the number of features using techniques such as mutual information to filter out those least correlated with the class labels.

During model training, cross-validation was used to reduce the risk of overfitting. We started with Logistic Regression. We then explored distance-based models - KNN and SVM, which are better suited to handling non-linear decision boundaries.

[1] https://www.kaggle.com/datasets/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign

Since the models above differ in their internal representation and how they learn patterns, we tried creating a stacking model to average out model errors. Both random forest and Logistic Regression were tested as the meta learner, and the results were compared to select the highest performing model.

After analysing the results of the four models, we refined our features and employed methods such as data augmentation to overcome common issues such as low resolution, noise, and variations in natural lighting. In addition, we used hyperparameter tuning with randomised parameter search to further optimise them.

Lastly, we trained a CNN model and compared the results to the simpler models. CNN was chosen as it is designed to capture both patterns and spatial information from raw pixel data, using deep learning.

For evaluation, we calculated F1 and accuracy scores on both the training and validation sets, as well as compared the scores for each class to assess model performance. This ensured that any class imbalance was taken into account.

3. Results (450 words)

3.1 Model Results

We will first present the results of all the learners, then a per-model result subsection.

Model	In-sample Accuracy (%)	Out-of-sample (validation) Accuracy (%)	Macro F1 Score	Weighted F1 Score
Logistic Regression	99.51	89.80	0.8817	0.8898
SVM	100	94.17	0.9231	0.9237
KNN	100	89.07	0.8943	0.8885
Stacking classifier	99.94	94.5	0.9325	0.9352
CNN	100	99.989	0.9975	0.9964

Table 3.1 - Table showing the in-sample and out-of-sample accuracies for all trained learners.

3.1.2 Logistic Regression

Logistic Regression has an 89% out-of-sample accuracy (Table 3.1).

Figure 3.2 gives an overview of the accuracy and the per-class F1-score for the Logistic Regression learner. This shows notable variation in per-class F1-scores.

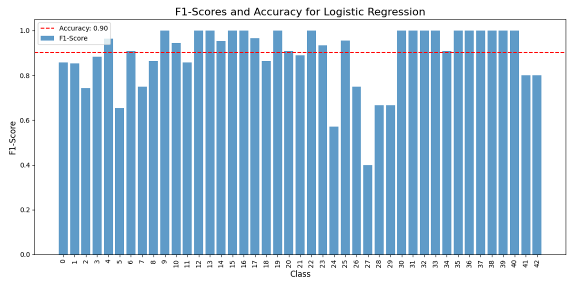

Figure 3.2 - The per-class F1-score and accuracy for Logistic Regression.

3.1.3 SVM

SVM achieves an out-of-sample accuracy of 94.17%, outperforming Logistic Regression by a margin of ≈ 4%.

Figure 3.3 shows that although there is notably less variation in the per-class F1 scores for SVM when compared to Logistic Regression. Both models have low F1 scores for many of the same classes.

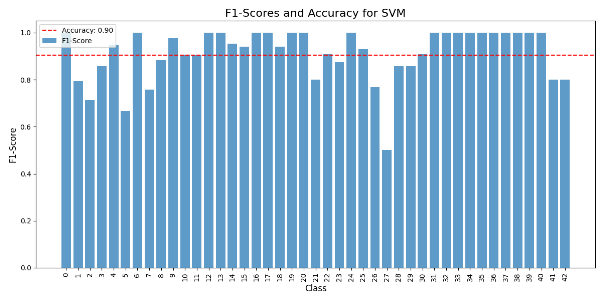

Figure 3.3 - The per-class F1-score and accuracy for SVM.

3.1.4 KNN

KNN has the lowest out-of-sample accuracy out of all learners. Figure 3.4 shows a similar F1-score variation to the Logistic Regression model.

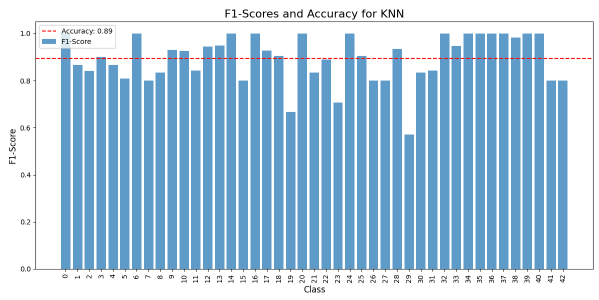

Figure 3.4 - The per-class F1-score and accuracy for KNN.

Figure 3.5 shows the optimal number of neighbours (K) for the KNN model. This turns out to be at $K = 1$.

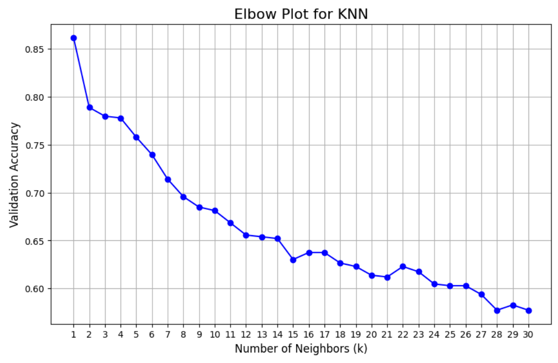

Figure 3.5 - Plot showing validation accuracy vs k hyperparameter in KNN

3.1.5 Stacking Classifier

The stacking classifier consists of 3 simple learners: Random Forest, SVM and Logistic Regression.

The ensemble outperformed all of its base estimators (see Figure 3.6) as well as all other learners, excluding the CNN model.

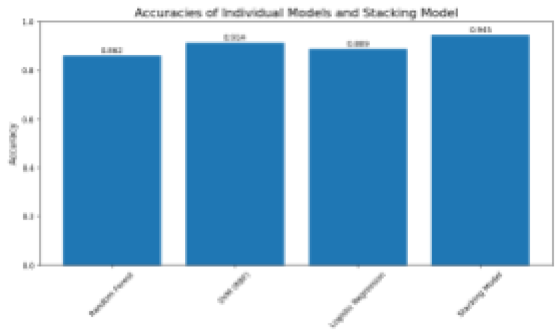

Figure 3.6 - Accuracy scores for individual models and stacking models.

Figure 3.7 shows that although the stacking model has less variable per-class F1-scores, it still misclassifies many of the same classes as the other models.

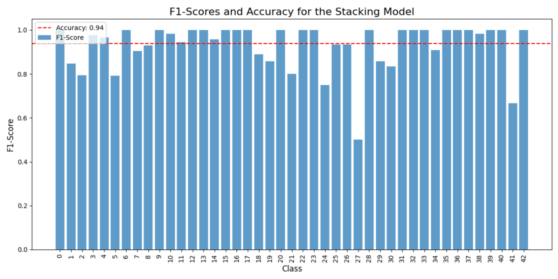

Figure 3.7 - The per-class F1-score and accuracy for the stack model.

3.1.6 CNN

The CNN model outperforms all other learners. It achieves 100% training accuracy and about 99. 8% validation accuracy.

Although we capped training at 50 epochs, the model converged fully after approximately 30 epochs, as seen in Figure 3.8 and Figure 3.9.

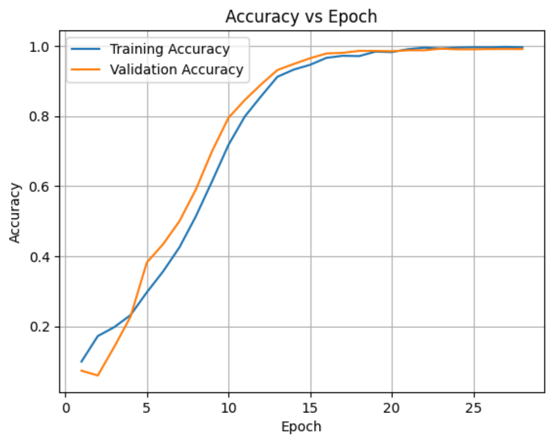

Figure 3.8 - Accuracy vs epoch during CNN model training.

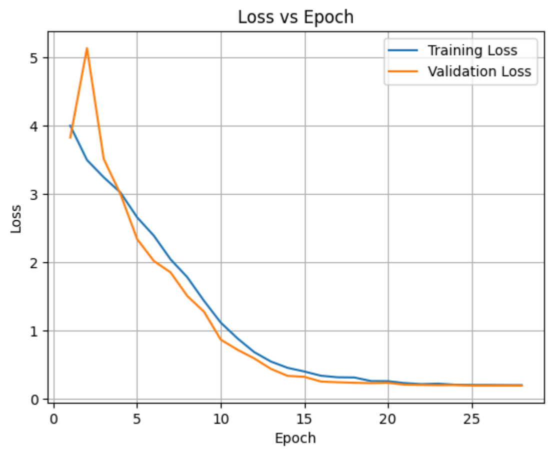

Figure 3.9 - Loss vs epoch during CNN model training.

Figure 3.10 demonstrates the consistency and robustness of the CNN model with; high F1 scores for all classes and a (rounded up) accuracy of 100%.

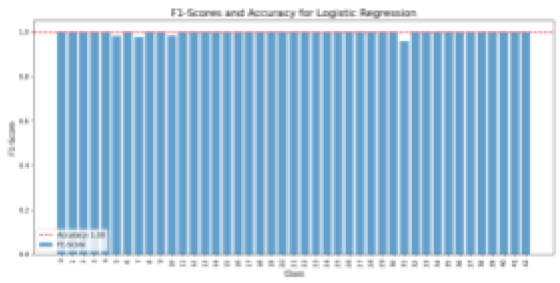

Figure 3.10 - The per-class F1-score and accuracy for the CNN.

3.2 Hyperparameter Tuning

Using GridSearch we obtained the following results for each model:

Model	Hyperparameters	Values
Logistic Regression	Max Iterations	500
	Penalty	L2
SVM	C	0.15
	Kernel	Linear
KNN	K	1
	Metric	Manhattan
	Weights	Uniform

Table 3.2 - Tuned hyperparameter values for Logistic Regression, SVM, and KNN models.

3.3 Feature Engineering and Selection

After feature engineering, we had the following feature-sets:

Engineered Feature	Description
Binned HSV	Offers a more interpretable brightness and colour representation.
HOG PCA	Offers a rich description of shape patterns without spatial specificity.
Local Binary Pattern	Gives a localised description of shape patterns
Shape Features	Uses contours, Hue moments and Convex Hull to explain global shape patterns
R, G, B moments	This includes the Skew and Variance for each of these colour channels. Gives more insight into the colour composition of images.

Table 3.3.1 - Comparison of model accuracies before and after feature selection.

Doing feature selection with a mutual information criterion, we decreased the number of features from 204 to 107. This reduction can be seen in Figure 3.11 and Figure 3.12.

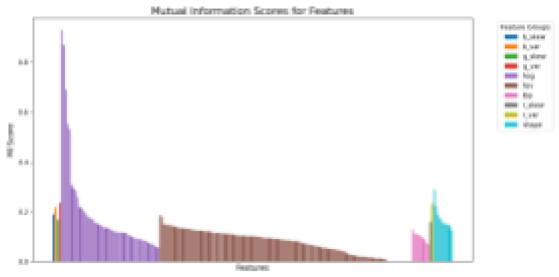

Figure 3.11 - The mutual information scores for different feature sets before feature selection.

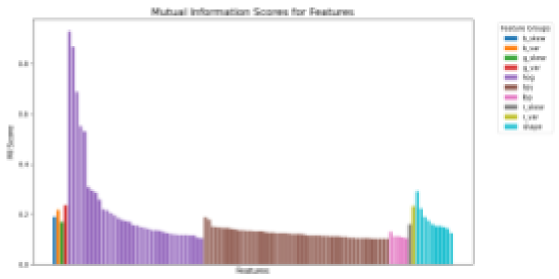

Figure 3.12 - The mutual information scores for different feature sets after feature selection.

Using this feature selection, we obtained the following accuracies:

Model	Pre Feature extraction out-of-sample (validation) Accuracy (%)	Post Feature extraction out-of-sample (validation) Accuracy (%)
Logistic Regression	89.80	88.89
SVM	94.17	91.9
KNN	89.07	89.2
Stacking classifier	94.5	94.5

Table 3.3.2 - Comparison of model accuracies before and after feature selection.

4 Discussion and Critical Analysis

Although all the learners performed strongly ($> 89\%$ validation accuracy), there are notable patterns and useful observations evident in each of the learners' performances. These insights help in identifying strategies for improving these models and identifying better model options. This iterative process culminated in our CNN model achieving near-perfect accuracy.

Before analysing individual model behaviour, it is important to mention some of the inherent challenges in the data. A major challenge present in the dataset is the class imbalance of the training data, as seen in Figure 4.1. Some classes have as few as 30 training instances. Given this limitation, it becomes challenging for simple learners to generalise effectively.

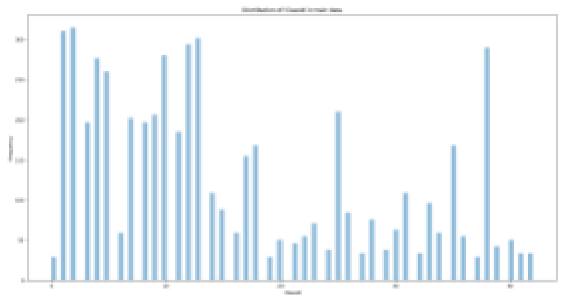

Figure 4.1 - Distribution of class ID in training data.

4.1 Logistic Regression

Logistic Regression serves as a useful baseline due to its simplicity, interpretability and minimal tuning. Its performance is, however, constrained by the assumption of linear decision boundaries and sensitivity to outliers (Techietory, 2022).

As shown in Figure 3.2, Logistic Regression exhibits lower per-class F1 scores for several specific classes, but still performs strongly with an accuracy of 89.8%. This strong performance indicates that the data is likely linearly separable.

Many of the mis-classified classes correspond to the classes with the lowest number of training instances (Figure 4.1). This suggests that Logistic Regression struggles to generalise from the limited dataset. This is an expected result. Logistic Regression does not handle class imbalance well and lacks the capacity to model complex patterns (Techietory, 2022).

Moreover, many of the often-misclassified classes share similar visual features to one another (see Figure 4.2). That is, they have similar shape and colour profiles and differ only in small, fine-grained details that Logistic Regression is not sophisticated enough to differentiate between.

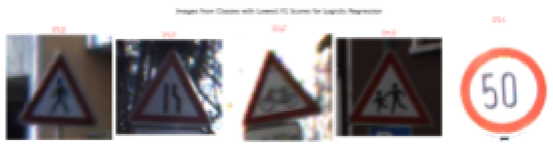

Figure 4.2 - Examples of classes with the lowest F1 scores for Logistic Regression.

This limitation motivated us to experiment with models that have non-linear decision boundary functionality and are less sensitive to outliers.

4.2 SVM

SVM generalises well to higher dimension data, supports non-linear decision boundaries through kernel functions, and is robust to outliers. This addresses many of the issues we faced with the Logistic Regression classifier.

Although our GridSearch included non-linear kernels, the best-performing model used a linear kernel, indicating that the classes are approximately linearly separable in the current feature space (see Table 3.2).

These improvements on linear regression allowed for an improvement of approximately 5% to a total accuracy of 94. 17%.

The error profile in Figure 3.3 shows that although SVM misclassified many of the same classes as Logistic Regression, each of these often-misclassified classes have a higher F1-score than in Logistic Regression. Clearly, then, SVM has improved on the issues of Logistic Regression.

The SVM, however, still has many of the same issues that Logistic Regression had. Specifically, not being able to differentiate between small differences, this can be seen in Figure 4.3.

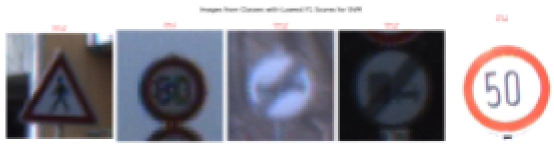

Figure 4.3 - Examples of classes with the lowest F1 scores for SVM.

Figure 4.4 shows that the least confidently classified instances (closest to the decision boundary) from the validation set have the same red-black colour palette with similar shape features. It is worth noting, however, that SVM still classifies all of these instances correctly. Figure 4.5 shows that the misclassified instances belong to the same family of classes as the least confidently classified instances.

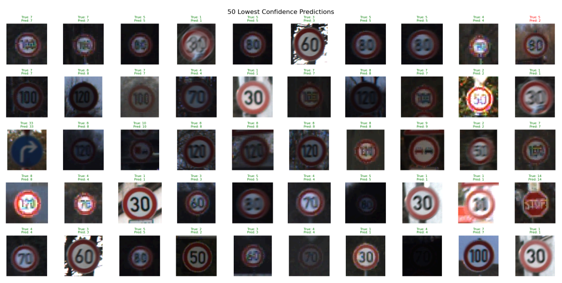

Figure 4.4 - Least confidently classified instances by SVM.

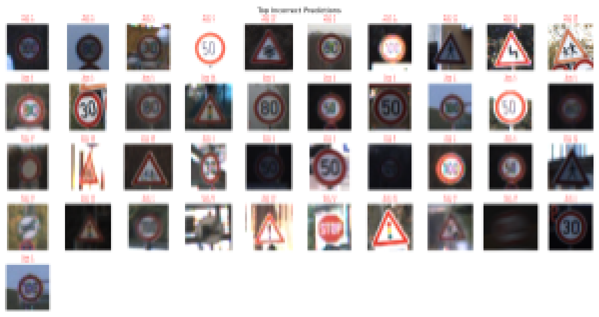

Figure 4.5 - Incorrectly classified instances by SVM.

Although the SVM model has high accuracy, it could be improved by addressing the issues that we saw in both Logistic Regression and SVM.

4.1 KNN

Like SVM, KNN supports non-linear decision boundaries. Unlike SVM, it makes decisions based on the local grouping of data, requiring no assumption about the global structure of the data. This can be advantageous when the class boundaries are complex, but is prone to overfitting.

We performed hyperparameter tuning using randomised search between the interval $[1,\ 10]$ on KNN and found $k\ =\ 1$ neighbour to be the value that corresponds to the highest accuracy score of 87.8%.

Figure 3.4 shows the per-class F1 and accuracy scores. Similar to the Logistic Regression and SVM models, the scores for certain classes fall below the average, most prominently classes 19 and 29 which are both triangular signs. This suggests KNN “extracts” the same type of information from the features as the previous models did, and lacks the ability to generalise fine detail patterns to classify similar signs.

Although KNN supports flexible decision boundaries, selecting k = 1 causes the model to rely on individual training points during classification, which makes it susceptible to noise. This means the model tends to generalise poorly - any small shift in colour and position, or the presence of outlier images, can lead to misclassifications. Since the dataset is imbalanced, noisy and the features are high dimensional, this sensitivity results in overly complex decision boundaries. Therefore, it is highly likely for the model to overfit to the training data.

Through experimentation, we found that data augmentation also degrades model performance. This is likely because it adds noise to the training data, which makes distances in the feature space less meaningful. Therefore, KNN is not an ideal model if we wish to achieve higher classification accuracy.

4.1 Stacking

The goal of stacking is to combine the strengths of multiple base models while mitigating their individual weaknesses.

Initially, we attempted to stack all the previously mentioned models with a Random forest meta-classifier.

This ensemble underperformed, achieving only 92% validation accuracy. This is lower than the best single model (SVM at 94.17%).

This result can be explained by the fact that these models share similar error patterns. Logistic Regression, SVM and KNN all misclassify many of the same classes as discussed previously. Figure 4.6 confirms that the learners all share similar error patterns.

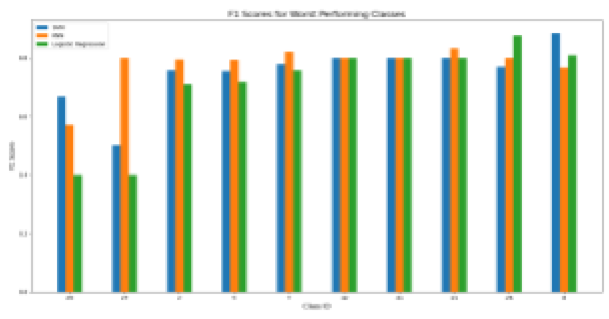
Figure 4.6 - Worst classified classes for each model.

This behaviour reduces the effectiveness of stacking, which benefits most from diverse base learners (Patnaik et al., 2020).

To address this, we constructed a new stacking ensemble using base models with distinct biases and error profiles: Random Forest, Radial Basis SVM, and Logistic Regression, again using Random Forest as the meta-learner.

As a base estimator, Random Forest captures non-linear feature interactions and is resilient to outliers and noise. Radial Basis SVM excels in high-dimensional, non-linearly separable spaces by maximising the decision margin and is able to capture relationships that Random Forest cannot. Logistic Regression offers a simple model for simple, linearly separable cases. By combining these models, we aimed to enhance generalisation through model diversity.

As shown in Figure 4.7, this approach was successful. The ensemble achieved the highest validation accuracy so far at 94.5%, while also producing more consistent per-class F1 scores with reduced variance. This indicates improved robustness and better generalisation across all classes.

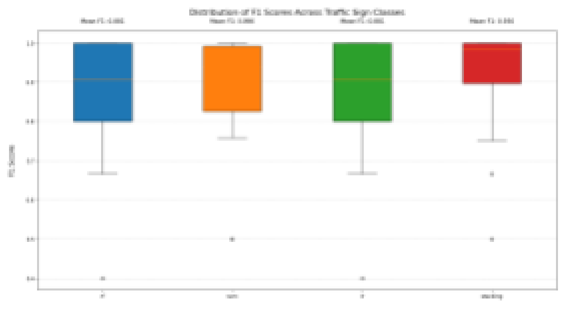
Figure 4.7 - Worst classified classes for each model

Although stacking provides a modest accuracy improvement, all four of the models explored so far, lack the ability to learn fine-detail patterns.

This is because the current feature set does not provide location information. This led us to experiment with feature engineering by first splitting images into 4 position-specific chunks, then binning HSV values in each chunk, which should theoretically provide rough spacial information. This did not result in measurable changes in the accuracy scores.

A possible explanation for this is that binning by chunk makes it harder for the models to generalise global patterns, which counteracts the gain in spatial information. Additionally, splitting each image into 16 chunks may not provide significant information to improve the classification performance.

Furthermore, despite attempts to improve the amount of information encoded in the tabular data through feature selection using mutual information, we found that reducing dimensionality did not yield consistent accuracy gains across any models (Table 3.3.2).

Similarly, after feature engineering with a focus on spatial understanding (Table 3.3.1), we reached a point where all models were stagnant in their performance.

This suggests that while feature selection removes noisy dimensions and feature engineering adds some depth to model understanding, features still lack complex spatial specificity needed for differentiating similar traffic signs. This limitation further justifies transitioning to CNNs, which rich spatial understanding directly from pixels.

4.2 CNN

Our CNN model is inspired by commonly used architectures for image classification (Jain, 2025; Ioffe et al., 2024), especially Visual Geometry Group (Alzubaidi et al., 2021, 30). After trial and error, we finalised on 3 convolution blocks, each with 2 convolution layers. We also used GlobalAveragePooling2D to reduce the number of parameters to minimise overfitting. We also added an intermediary dense layer to learn combinations of the pooled features to improve final classification.

Among all the models, CNN performed the best - with a round 100% training accuracy and 99.8% validation accuracy. This significant performance boost is likely due to CNN's ability to capture spatial features from raw pixel data.

Unlike previous models such as SVM which ignore location, CNN learns both the pattern and where it occurs, allowing it to recognise fine patterns such as arrows and numbers. These patterns are critical for distinguishing similar traffic signs like speed signs. Figure 4.8 shows the patterns learnt by the CNN at different layers, understood by looking at the activations at each convolutional layer.

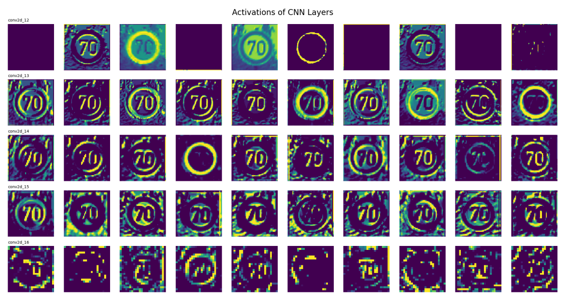

Figure 4.8 - Activation of convolution layers.

Furthermore, the CNN's robustness to small translation accounts for the natural shifts in the framing of signs in the dataset. However, the initial model still struggled with other types of noise, such as grain, extreme brightness and partial obstruction. Figure 4.9 shows an example of misclassification due to noise. To address this, we applied data augmentation to add noise to the training data. This helped the model correctly classify more noisy images with an increased validation accuracy of 0.62% from 99.27% to 99.89%.

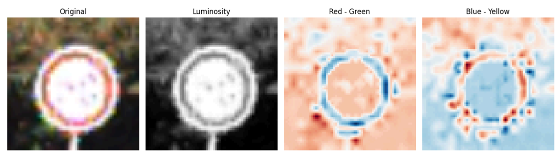

Figure 4.9 - Breakdown of luminosity, red - green, and blue - yellow features, compared to the original image. The image belongs to class 2, but was misclassified as 1 before data augmentation.

The per class F1 and accuracy scores shows that the CNN classifies with similar performance for all classes. All class scores lie near the average scores of almost 100% as shown in Figure 3.10. This is approximately a 5% - 10% improvement compared to previous models after tuning. This shows that the CNN model extracts more meaningful patterns across all sign types, despite the class imbalance and visual similarities among certain sign types.

Initially, we used HSV values as input for the model. However, the scale for hue (0-255) is circular, which means both ends of the scale are red. Hue is also unstable under low light conditions as the brightness is defined by the V component. The discontinuity in the hue could confuse the model. Therefore, we experimented using red - green and blue - yellow contrasts as well as luminosity as input. This highlights the colour contrasts between different types of traffic signs, which are typically red, yellow, or blue. This also emulates human vision, where colours are detected in red/green and blue/yellow pairs. Figure 4.9 visualises the image after processing. Results show a slight improvement in the accuracy scores by +2.0%, suggesting these channels help the model generalise features under real world lighting conditions.

To reduce overfitting, we applied L2 regularisation to each convolution layer, which penalises larger weights. Large weights can cause the model to over assign importance to local regions of images, increasing the risk of overfitting and making the model susceptible to noise. This would increase the variance and result in lower performance.

5 Conclusions

Although simple learners like Logistic Regression, Support Vector Machine, and K-Nearest Neighbours achieve strong performance on the GTRSB classification task, they are ultimately constrained by the representational limitations of tabular features.

These models rely on a compressed, fixed set of engineered features which leads to them producing similar error patterns, especially on training instances that differ only in fine-grained details.

While ensemble approaches, like stacking, can give a marginally improved result by combining these learners, ultimately, these classifiers suffer from the same problems that the previously mentioned learners do, as they operate on the same dataset. Without access to the raw image data, even an ensemble classifier struggles to distinguish between visually similar classes, as they share similar shape and color profiles.

To overcome these representational challenges, a model capable of learning the complexity of image data from the pure pixel data is necessary. A Convolutional Neural Network is well suited to this problem and performs as it learns latent, complicated features from the raw image data and recognises small nuances in similar class instances. In our experiments, the CNN dramatically outperformed all other approaches, achieving near-perfect validation.

This progression highlights the importance of both model complexity and feature representation in image classification tasks. While simpler machine learning techniques can offer strong performance with well-crafted features, deep learning models like CNNs are essential for achieving a reliable model with consistently high accuracy.

6 References

Alzubaidi, L., Zhang, J., & Humaidi, A. J. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Review of deep learning: concepts, CNN architectures, challenges, applications, future directions.*, *8*(53), 74. https://doi.org/10.1186/s40537-021-00444-8

Ioffe, S., Szegedy, C., & Jain, S. (2024, May 13). *What is Batch Normalization in CNN?* GeeksforGeeks. Retrieved May 23, 2025, from https://www.geeksforgeeks.org/what-is-batch-normalization-in-cnn/

Jain, S. (2025, April 26). *Convolutional Neural Network (CNN) Architectures*. GeeksforGeeks. Retrieved May 23, 2025, from https://www.geeksforgeeks.org/convolutional-neural-network-cnn-architectures/

Patnaik, S., Yang, X.-S., & Sethi, I. K. (Eds.). (2020). *Advances in Machine Learning and Computational Intelligence: Proceedings of ICMLCI 2019*. Springer Nature Singapore.

Techietory. (2025, January 11). *Introduction to Logistic Regression*. Techietory. https://techietory.com/ai/introduction-to-logistic-regression/