COMP30027 Assignment 2

Traffic Sign Classification

1. Introduction

The goal of this project was to build and evaluate machine learning models that can accurately classify traffic signs into one of the 43 classes provided by the given dataset.

This report outlines a full pipeline for the classification task, incorporating both traditional machine learning techniques like **Random Forest (RF)** and **Support Vector Machine (SVM)**, and a deep learning model like **Convolutional Neural Network (CNN)**. This included extensive feature engineering and pre-processing for the RF and SVM, while the CNN was trained directly on raw image inputs.

The performance of these models is compared through validation accuracy and error analysis, with a particular focus towards classification challeners, model behaviour, and model generalisation.

2. Methodology

2.1 Feature engineering

Feature engineering was only applied to traditional models (RF and SVM). Whereas the CNN used raw image inputs since there is less of a need for manual feature engineering (Bhattiprolu, 2025). A total of 12 features (shown in **Table 1**) were self-extracted using *OpenCV* and *scikit-image*, including the ones given as part of the specification.

Feature(s)	Notes
Mean B/G/R	Represents overall colour intensity
Mean H/S/V	Human-intuitive colour representation, more robust than BGR (Mohd Ali, Alang Md Rashid, & Mustafah, 2013)
Edge density	Canny edge detection to capture object boundaries (El-Sayed & Abd El-Hafeez, 2012)
Shannon entropy	Also used edge detection (El-Sayed & Abd El-Hafeez, 2012)
BGR colour histogram	Binned colour information, good for image classification tasks (Swain & Ballard, 1991)
HSV histogram	Similar to BGR colour histogram
Local Binary Pattern (LBP) histogram	Captures texture patterns(Ihalapathirana, 2023)
Histogram of Oriented Gradients (HOG)	Captures local shape and gradient structure effectively (Guo & Song, 2021)

Table 1 – Features extracted from raw images to use for RF and SVM models.

By engineering both colour and texture-based features, this gives more meaningful information for the models to learn from.

2.2 Feature processing

A stratified 90/10 train-validation split was used across all models to preserve class distributions during training and evaluation.

For the RF and SVM, to find the most informative predictors, **Mutual Information (MI)** was computed between each feature and the class labels. The histogram features were grouped (e.g., all bins for BGR histograms were treated as one feature group), and their MI scores were averaged. The top 10 out of 12 total features were selected for training. Furthermore, all features were standardised as distance-based models like SVMs are sensitive to feature scales.

For CNNs, no explicit feature engineering and processing was done, since as mentioned before, CNNs have embedded 'feature extraction' within itself. Instead, the CNN was trained directly on raw image inputs. Each image was normalised to [0, 1].

2.3 Models/Learners used

Three models were implemented and evaluated in this project: **Random Forest (RF)**, **Support Vector Machine (SVM),** and a **Convolutional Neural Network (CNN)**.

For both the RF and SVM, hyperparameters were tuned using grid search with 3-fold cross-validation (see **Table 2**). Due to time constraints, hyperparameter tuning was not done on the CNN, but rather based on commonly used configurations.

Model	Hyperparameter	Hyper-parameter values
RF	*n_estimators*	100, 200, 300, 400
	min_samples_leaf	1, 2, 3, 4
SVM	*C*	0.001, 0.01, 0.1, 1, 10, 100, 1000
	kernel	linear, polynomial, RBF

Table 2 – RF and SVM hyperparameters for grid search tuning.

The CNN architecture consisted of 3 convolutional blocks, each containing 2 convolutional layers. The first block used 32 filters, the second 64, and the third 128. Each block was followed by batch normalisation and max-pooling. After the 3 blocks, the model is followed by a flattening layer, a fully connected dense layer with 512 units, a dropout layer with dropout rate 0.5. And finally, a dense output layer with 43 units (one per class). The network was trained for 30 epochs minimising categorical cross-entropy loss.

To improve generalisation and prevent overfitting, data augmentation was applied to the training images using random rotation, zooming, shifts in width and height, etc. This strategy has been widely shown to enhance performance in many classification tasks (Thanapol, Lavangnananda, Bouvry, Pinel, & Leprévost, 2020).

2.4 Final train and test

After tuning, each model was retrained using the optimised hyperparameters on the full train set (including validation), which is then used to predict the unlabelled test data.

3. Results

All models (except for the CNN) were tuned to yield the optimal hyperparameters (shown in **Table 3**). The results of the grid search for the RF and SVM, showing validation accuracies across different hyperparameter combinations, are visualised in **Figures 1 & 4**, respectively.

The best-performing configuration for each model is also evaluated in terms of validation accuracy (**Table 3**), and further examined using confusion matrices (**Figures 2, 5, & 7**) and per-class accuracy charts (**Figures 3, 6, & 8**).

Model	Optimal hyperparameters	Validation accuracy
RF	*n_estimators* = 400, *min_samples_leaf* = 1	0.8415
SVM	*C* = 10, *kernel* = RBF	0.8907
CNN	N/A	0.9799

Table 3 – Validation accuracies of each model tuned to their optimal hyperparameters.

Model	Test accuracy
RF	0.7407
SVM	0.7367
CNN	0.9860

Table 4 – Final test accuracies of each model tuned to their optimal hyperparameters and retrained on the full train + validation set.

For the CNN, a learning curve (**Figure 9**) was plotted to compare training and validation performance across epochs.

Final test set predictions for each model are summarised in **Table 4**.

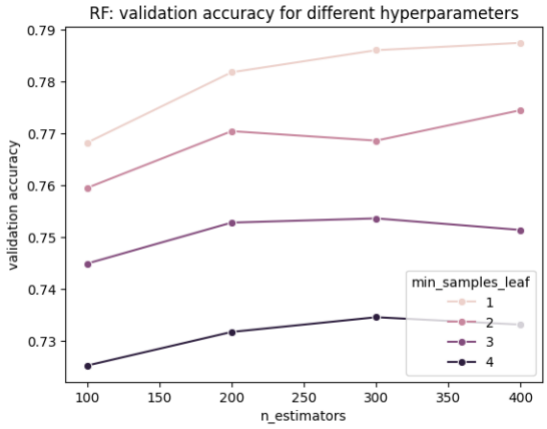

Figure 1 – RF validation accuracies for different hyperparameter combinations.

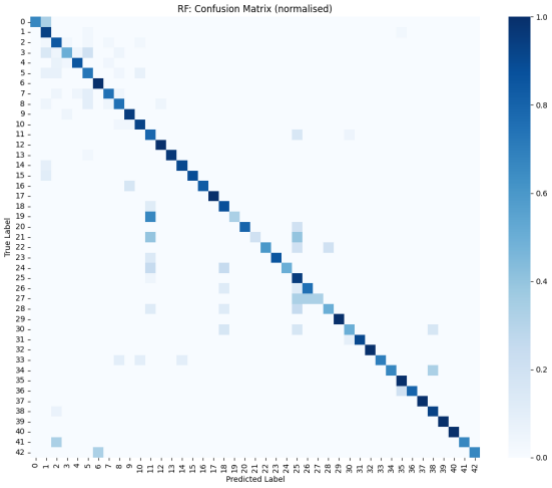

Figure 2 – RF confusion matrix (normalised) evaluated on validation set.

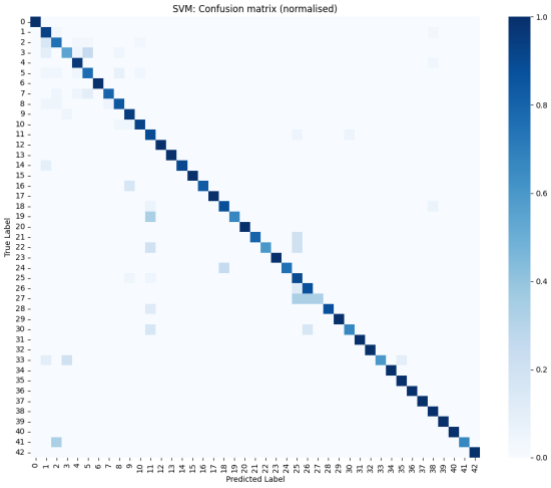

Figure 5 – SVM confusion matrix (normalised) evaluated on validation set.

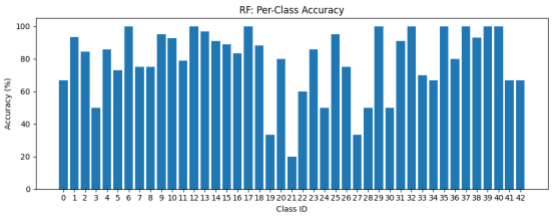

Figure 3 – RF per-class accuracy evaluated on validation set.

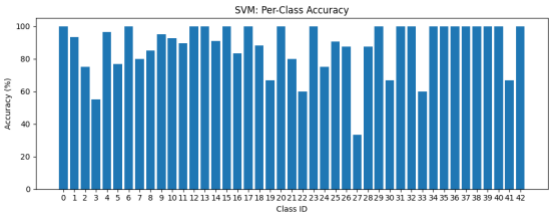

Figure 6 – SVM per-class accuracy evaluated on validation set.

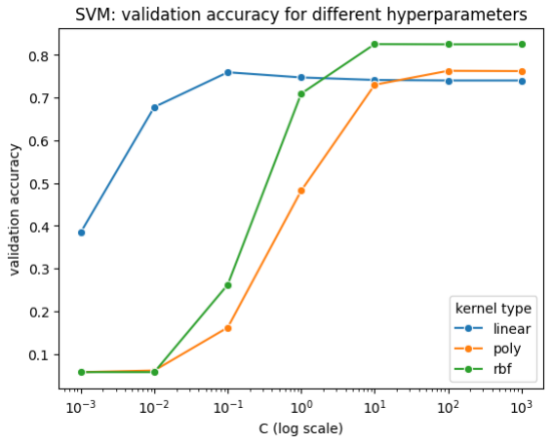

Figure 4 – SVM validation accuracies for different hyperparameter combinations.

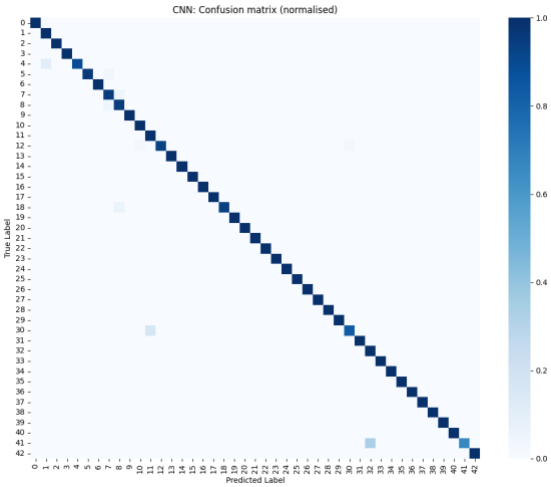

Figure 7 – CNN confusion matrix (normalised) evaluated on validation set.

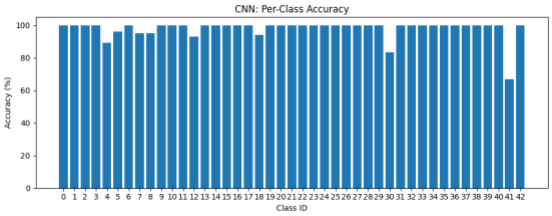

Figure 8 – CNN per-class accuracy evaluated on validation set.

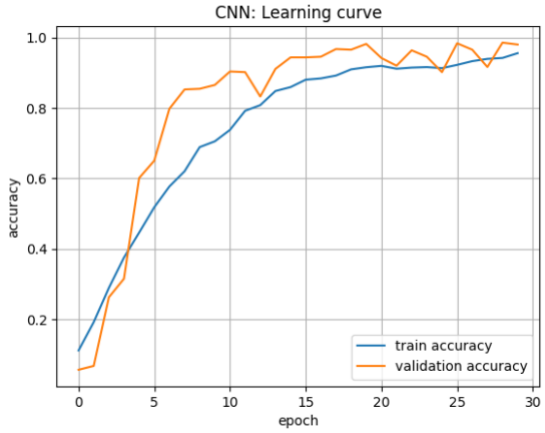

Figure 9 – CNN learning curve comparing training and validation accuracy over 30 epochs.

4. Discussion and Critical Analysis

Refer to **Figure 10** for an image reference of the different classes.

Figure 10 – Image references for all 43 traffic sign classes.

4.1 RF Evaluation

The RF model achieved a validation accuracy of 0.8415, with the optimal hyperparameters being *n_estimators* = 400 and *min_samples_leaf* = 1 (**Table 3**). As seen in **Figure 1**, validation accuracy improved with an increasing number of estimators, consistent with findings noting that larger ensembles help reduce variance and overfitting, ultimately improving performance (Breiman, 2001). The highest accuracy was achieved when the minimum samples per leaf was smallest, allowing trees to become more complex and recognise finer patterns within the features.

However, setting *min_samples_leaf* = 1 may have introduced unwanted sensitivity to noise. While the ensemble structure helps offset some of this overfitting, more complex trees can still overreact to outliers or rare feature combinations.

As illustrated in the per-class accuracy plot (**Figure 3**), classes 19, 21, and 27 exhibited the lowest accuracies – all below 40%. The confusion matrix (**Figure 2**) further reveals that class 19 (*dangerous curve to the left*) and class 21 (*double curve*) were often confused with class 11 (*right of way at intersection*). These signs share similar triangular shapes and red borders, suggesting that the model likely struggled to differentiate them based solely on the engineered features.

This highlights a key limitation of a feature-based classifier like RFs. While extracted features can describe general shape and texture, these models often overlook subtle differences that play a huge factor in traffic sign recognition.

4.2 SVM Evaluation

The SVM model achieved a validation accuracy of 0.8907, using *kernel* = RBF and *C* = 10 as the optimal hyperparameters (**Table 3**). As shown in **Figure 4**, the linear kernel performed reasonably well at lower values of *C*, but then later plateaued around 0.7. In contrast, both the polynomial and RBF kernels showed significant improvements at higher *C* values. The RBF kernel ultimately yielded the best results, indicating its suitability for modelling non-linear decision boundaries in high-dimensional feature spaces (Hsu, Chang, & Lin,

2003).

The linear kernel's early advantage at low C values reflects its tendency to perform well under less strict decision boundaries, being less sensitive to outliers, and therefore generalising better to unseen data. However, as C increases and the margin becomes stricter, the linear kernel lacks the flexibility to capture complex patterns like how the polynomial and RBF kernels do.

Despite overall strong performance of the SVM, classification challenges remained. **Figure 6** revealed that class 27 (*pedestrian crossing*) had the lowest per-class accuracy (< 0.4). And **Figure 5** showed that it was frequently misclassified as class 25 (*road work*) and class 26 (*traffic signals*). The visual overlap of these signs may have made it difficult for the model to find a distinct hyperplane capturing the nuanced differences between them.

Overall, the SVM with an RBG kernel effectively reduced bias by allowing non-linear separation of the data. Moreover, setting $C = 10$ offered a good trade-off between regularisation and minimising training error, therefore helping the model maintain low variance and generalise well to testing data.

4.3 CNN Evaluation

The CNN model achieved a validation accuracy of 0.9799, despite the absence of hyperparameter tuning (**Table 3**). The model architecture was arbitrarily selected based on commonly used design patterns, which proved highly effective in practice.

The learning curve (**Figure 9**) shows a stable training process. In the initial epochs, the training accuracy started out higher than the validation accuracy, which is typical as the model begins to fit. However, by epoch 3-4, the validation accuracy surpassed training accuracy and remained higher throughout the remainder of the training process. This pattern suggests that data augmentation significantly enhanced generalisation by increasing the diversity of training examples. Both training and validation rose steadily, with no significant divergence, indicating that the model learned progressively and avoided overfitting. The small gap between the two curves further supports the conclusion that the model maintained a good balance between bias and variance, being able to capture complex patterns while still generalising well.

Analysis of the confusion matrix (**Figure 7**) and per-class accuracies (**Figure 8**) showed near-perfect classification across all 43 classes. However, there was a minor, but consistent, misclassification that occurred between class 41 (*end of no passing*) and class 32 (*end of speed and passing limits*) – two signs which are very similar visually and semantically. As such a subtle visual overlap was the only notable source of error, this further confirms the model's capacity to learn fine-grained features directly from the raw image data.

Overall, despite the absence of hyperparameter tuning, the model architecture combined with regularisation through data augmentation enabled strong model generalisation.

4.2 Model Comparison

In terms of overall performance, the CNN significantly outperformed both traditional methods.

The RF and SVM models relied heavily on engineered features like colour histograms, HOGs, etc. While this approach gave reasonable results, it lacked the depth needed to recognise subtle visual distinctions. Contrastingly, the CNN removed the need for manual feature extraction by learning its own representations, and therefore significantly improving validation accuracy across all classes.

The nature of classification errors also varied between models. The RF and SVM frequently confused signs with similar outlines, shapes, or patterns. Whereas the CNN made very insignificant misclassifications.

Analysing bias and variance, the CNN was the most well balanced – being able to capture complex patterns (low bias) while mitigating overfitting with a dropout layer and data augmentation (low variance). The SVM was also able to generalise well with the chosen hyperparameters, whereas the RF may have showed signs of overfitting with small leaf sizes.

5. Conclusion

This project evaluated three machine learning approaches for traffic sign classification, demonstrating the effectiveness of both feature-based and deep learning methods. Although the RF and SVM performed well using engineered features, the CNN consistently outperformed them in both accuracy and generalization. Its ability to learn directly from the raw image data allowed it to capture subtle distinctions between classes, and therefore showed strong robustness without requiring extensive hyperparameter tuning. These results underscore the suitability of deep learning for complex image classification tasks.

6. References

Bhattiprolu, S. (2025, March 11). *Feature Learning vs. Feature Engineering: Image Analysis Gets a Deep Learning Upgrade*. Retrieved May 2025, from Zeiss: https://www.zeiss.com/microscopy/en/resources/insights-hub/foundational-knowledge/deep-learning-vs-feature-engineering-in-image-analysis.html

Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern recognition and machine learning.* Springer.

Breiman, L. (2001, October). Random Forests. *Machine Learning, 45*, 5-32.

El-Sayed, M., & Abd El-Hafeez, T. (2012, November 11). New Edge Detection Technique based on the Shannon Entropy in Gray Level Images. *International Journal on Computer Science and Engineering, 3*.

Guo, S., & Song, Y. (2021, August 11). Traffic sign recognition based on HOG feature extraction. *Journal of Measurements in Engineering, 9*(3), 142-155.

Hsu, C.-w., Chang, C.-c., & Lin, C.-J. (2003, November 29). A Practical Guide to Support Vector Classification. *Department of Computer Science, National Taiwan University*.

Ihalapathirana, A. (2023, August 25). *Understanding the Local Binary Pattern (LBP): A Powerful Method for Texture Analysis in Computer Vision*. Retrieved May 2025, from Medium: https://aihalapathirana.medium.com/understanding-the-local-binary-pattern-lbp-a-powerful-method-for-texture-analysis-in-computer-4fb55b3ed8b8

Mohd Ali, N., Alang Md Rashid, N., & Mustafah, Y. (2013, September 3). Performance Comparison between RGB and HSV Color Segmentations for Road Signs Detection. *Applied Mechanics and Materials, 393*, 550-555.

Swain, M. J., & Ballard, D. H. (1991, November). Color indexing. *International Journal of Computer Vision, 7*, 11-32.

Thanapol, P., Lavangnananda, K., Bouvry, P., Pinel, F., & Leprévost, F. (2020). Reducing Overfitting and Improving Generalization in Training Convolutional Neural Network (CNN) under Limited Sample Sizes in Image Recognition. *5th International Conference on Information Technology (InCIT)*, 300-305.