**GTSRB Classification**

**COMP30027 Report**

1. **Introduction**

In 2025, convolutional neural networks (CNNs) are said to “lead the charge” (Alvi, 2024) when it comes to image classification tasks. For the GTSRB traffic sign classification task, I hence sought to utilize a CNN as well as other ML models such as a Random Forest (RF) and Support Vector Machine (SVM) and compare their effectiveness for labelling images. Accuracy is of utmost importance for real world applications of our classification task (e.g. self-driving cars), so I stacked each model I built and chose optimal parameters in order to make my predictor as reliable as possible.

To produce good results, I needed relevant and informative features of which I extracted deliberately so each sign image could be distinguished from those of other classes.

Lastly, to achieve the best accuracy I set up a stacking classifier to combine the predictive power of my models and best classify test instances.

**2.0 Methodology**

Tackling this task effectively involved creating a robust workflow where data processing errors do not occur and hard coding is discouraged. To do this I broke the task up into these subtasks:

**2.1 Feature Extraction**

To distinguish between classes, I engineered additional features from the raw image data to capture more discriminative information for future classifiers to utilise.

Firstly, I extracted features to represent shape-based characteristics of the images. To do this I pre-processed the images to ensure my feature extraction methods could distinguish these shape-based features as best as possible. This included applying filters, blurs, masks as well as enhancing contrast. Preprocessing was slightly different for each feature engineered to ensure the best results were achieved, this was done iteratively by visually evaluating effects on feature distinctiveness.

For shape features I utilised OpenCV to extract:

1. Contours (curves or boundaries for the subject of the image) and passed their associated properties such as aspect ratio and solidity into my model.
2. Hu Moments (seven shape descriptors which are transformation invariant), these are calculated from statistical properties of pixel intensities. I chose to use these as they help identify shapes even if the subject is rotated, scaled or translated.
3. Edge Orientation Histograms: This feature captures the distribution of edges directions of the subject in our image. It utilises edge detection algorithms which calculate gradient direction at edge pixels in the image and group them into bins. Given that traffic signs are characterised by their well-defined boundaries that do not change based on class, this feature provided beneficial information to help models distinguish signs.

Lastly, I extracted features from the HSV (Hue, Saturation, Value) colour space to provide my models with more robust colour information. HSV separates colour from brightness to give my models information about the colour of signs in low light photos. These features seemed useful to differentiate instances where colour was a key discriminative factor in whether the sign was one class or another.

**2.3 Feature Selection**

Before training, I selected the most informative features using mutual information as I had an abundance of continuous features, some of which were not strongly informative. I also removed highly correlated features as information extracted from them can be extracted from other features instead.

|  |  |
| --- | --- |
| Feature: | MI |
| hog\_pca\_0 | 0.882652 |
| hog\_pca\_3 | 0.796677 |
| Edge\_Hist\_Bin\_6 | 0.596259 |
| Edge\_Hist\_Bin\_2 | 0.566049 |
| hog\_pca\_1 | 0.554871 |
| Edge\_Hist\_Bin\_7 | 0.530975 |
| Edge\_Hist\_Bin\_3 | 0.519416 |
| hog\_pca\_2 | 0.493234 |
| Edge\_Hist\_Bin\_5 | 0.429996 |
| H\_hist\_bin\_16 | 0.40201 |

**Figure 2.3.1-** Most informative features, sorted by mutual information.

**2.4 Model Selection / Validation**

I then chose to create 3 base models to train and stack using a meta classifier. I trained each model using the same (K=5) fold test validation splits to ensure no data leakage.

1. SVM – Good for separating classes in high dimensional feature spaces like mine with the features I created.
2. RF – Robust to noise, overfitting, no extra preprocessing/scaling needed. Easy and fast to train and perform well on a large range of feature types which our dataset has.
3. CNN – Models complex spatial patterns in the image data, layers extract features that are more informative than those I created for the other models.

A graph with a line going up

AI-generated content may be incorrect.

**Figure 2.4.1-** SVM Accuracy VS Hyperparameter: C

As seen in figure 2.4.1, I then used grid search to tune each model (like SVM) to predict accurately and not overfit. IN this case I chose C = 5 to balance accuracy and efficiency and not risk overfitting.

After training, I extracted the prediction probabilities from each validation fold for each of the models, these validation sets hadn’t been seen by each model and hence data leakage could be avoided. These were then fed into a Logistic Regression metamodel, which learnt how to optimally combine my base model predictions by identifying patterns in the way each model classified instances. I chose this approach to ensure a reduction in bias and variance and **better generalisation** from my original base models, as stacking enables the meta-learner to correct systematic errors by learning how to weight model outputs based on their strengths and weaknesses.

Lastly, to assess model performance I calculated Bias, Variance, Categorical Cross-Entropy Loss and Accuracy and tweaked hyperparameters for best performance.

1. **Results**

**3.1 SVM Performance**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Fold** | **Accuracy** | **Bias** | **Variance** | **Cross-Entropy Loss** |
| 1 | 0.8634 | 0.071 | 0.0142 | 0.5618 |
| 2 | 0.8807 | 0.0694 | 0.01387 | 0.5519 |
| 3 | 0.8789 | 0.0681 | 0.01393 | 0.5185 |
| 4 | 0.8651 | 0.0702 | 0.01411 | 0.5385 |
| 5 | 0.8678 | 0.0701 | 0.01391 | 0.5491 |
| **Mean** | **0.8712** | **0.0698** | **0.014** | **0.544** |

|  |  |
| --- | --- |
| Feature: | MI |
| hog\_pca\_0 | 0.882652 |
| hog\_pca\_3 | 0.796677 |
| Edge\_Hist\_Bin\_6 | 0.596259 |
| Edge\_Hist\_Bin\_2 | 0.566049 |
| hog\_pca\_1 | 0.554871 |
| Edge\_Hist\_Bin\_7 | 0.530975 |
| Edge\_Hist\_Bin\_3 | 0.519416 |
| hog\_pca\_2 | 0.493234 |
| Edge\_Hist\_Bin\_5 | 0.429996 |
| H\_hist\_bin\_16 | 0.40201 |

**Figure 3.1.1-** SVM Model Accuracy, Loss, Bias & Variance across Validation folds.

**3.2 RF Performance**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Fold** | **Accuracy** | **Bias** | **Variance** | **Cross-Entropy Loss** |
| Fold 1 | 0.8306 | 0.1012 | 0.006 | 1.0685 |
| Fold 2 | 0.8388 | 0.1018 | 0.0058 | 1.0852 |
| Fold 3 | 0.8324 | 0.1017 | 0.0059 | 1.0725 |
| Fold 4 | 0.8295 | 0.1025 | 0.0057 | 1.0915 |
| Fold 5 | 0.8177 | 0.1037 | 0.0057 | 1.1137 |
| **Mean** | **0.8298** | **0.1022** | **0.0058** | **1.0863** |

**Figure 3.2.1-** RF Model Accuracy, Loss, Bias & Variance across validation folds.

**3.3 CNN Performance**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Fold** | **Accuracy** | **Bias** | **Variance** | **Cross-Entropy Loss** |
| Fold 1 | 0.9617 | 0.037 | 0.0217 | 0.188 |
| Fold 2 | 0.9545 | 0.039 | 0.0216 | 0.16 |
| Fold 3 | 0.9608 | 0.0371 | 0.0216 | 0.1303 |
| Fold 4 | 0.9572 | 0.0399 | 0.0215 | 0.1875 |
| Fold 5 | 0.9663 | 0.0345 | 0.0217 | 0.1341 |
| **Mean** | **0.9601** | **0.0375** | **0.0216** | **0.16** |

**Figure 3.3.1-** CNN Model Accuracy, Loss, Bias & Variance across Validation folds.

**3.4 Stacking Model Performance**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Fold** | **Accuracy** | **Bias** | **Variance** | **Cross-Entropy Loss** |
| Fold 1 | 0.9763 | 0.0293 | 0.0216 | 0.0973 |
| Fold 2 | 0.9709 | 0.0314 | 0.0217 | 0.1105 |
| Fold 3 | 0.9754 | 0.0298 | 0.0215 | 0.0959 |
| Fold 4 | 0.9708 | 0.0309 | 0.0215 | 0.1043 |
| Fold 5 | 0.9754 | 0.0306 | 0.0216 | 0.1054 |
| **Mean** | **0.9738** | **0.0304** | **0.0216** | **0.1027** |

**Figure 3.4.1-** Stacking Meta-model Accuracy, Loss, Bias & Variance across Validation folds.

1. **Discussion & Critical Analysis**
2. **Conclusions**

Concluding text.

1. **References**

Christopher M Bishop and Nasser M Nasrabadi. 2006. *Pattern recognition and machine learning*, volume 4. Springer.

Alvi, F. (2024, March 20). *Image classification in 2025: Insights and advances*. OpenCV. <https://opencv.org/blog/image-classification/>