Neural Network Performance: A Comparative Analysis of Pre-processing Techniques on Image Classification using Custom Convolutional Neural Networks

Adityaa Gupta
20324976
COMP47650 Deep Learning

Abstract

In the realm of image classification, particularly within traffic sign recognition, the choice of pre-processing techniques significantly impacts model performance. This study delves into the comparative analysis of Convolutional Neural Networks (CNN) under various pre-processing conditions, utilizing the traffic signs pre-processed dataset from Kaggle. We examine a range of pre-processing methods, including resizing, normalization, and augmentation, to ascertain their optimal application for CNN models. Through rigorous experimentation, we aim to uncover the most effective pre-processing strategies for CNNs, thereby enhancing their classification accuracy. The insights derived from this research are poised to contribute to the optimization of image classification models for traffic sign recognition, offering valuable guidance for deep learning practitioners and shaping future research directions in this domain.

Contents

1 Introduction

Image classification, a cornerstone of computer vision, has seen remarkable advancements with the advent of deep learning. Convolutional Neural Networks (CNNs) have emerged as powerful tools for this task, demonstrating high accuracy in various domains, including traffic sign recognition. However, the performance of these models is heavily dependent on the pre-processing techniques applied to the input data. This study aims to explore the impact of different pre-processing methods on the performance of CNNs in the context of traffic sign recognition. By leveraging the traffic signs pre-processed dataset from Kaggle, we investigate the effectiveness of resizing, normalization, and augmentation techniques. Our goal is to identify the most beneficial pre-processing strategies for CNNs, thereby enhancing their classification accuracy and contributing to the optimization of image classification models for traffic sign recognition.

2 Related Work

This section provides an overview of the research topic, focusing on the significance of CNNs in image classification tasks.

CNNs

A Convolutional Neural Network (CNN) stands as a pinnacle in image analysis within deep learning, drawing inspiration from the intricate workings of the human visual cortex. Its architecture comprises convolutional layers, pooling layers, and fully connected layers. Convolutional layers extract diverse features from the input data through the application of specialized filters, while pooling layers reduce spatial dimensions, enhancing computational efficiency and mitigating overfitting. These features culminate in fully connected layers, where high-level abstractions are utilized for classification tasks.

CNNs find extensive applications across various domains, showcasing remarkable prowess in image recognition and processing tasks. From image classification, where they discern intricate details to categorize images, to object detection, where they locate and identify objects within complex scenes, CNNs demonstrate versatility. Additionally, their capabilities extend to image segmentation, facilitating the partitioning of images into distinct segments for detailed analysis, as well as video analysis, enabling tasks like object tracking and event detection. Through these applications, CNNs emerge as indispensable tools in unraveling the complexities of visual data across numerous domains [1, 2].

Pre-processing Techniques for Image Classification

Pre-processing techniques play a crucial role in enhancing the performance of image classification models. These techniques, such as resizing, normalization, and data augmentation, are essential for preparing images for analysis by deep learning models. Resizing ensures that all images are of a consistent size, which is critical for convolutional neural networks (CNNs) that require fixed input dimensions. Normalization standardizes the pixel values of images, which helps in stabilizing the learning process by ensuring that all input features have a similar scale [4]. Data augmentation involves creating new training samples by applying transformations such as rotation, scaling, and flipping to the original images. This technique not only increases the diversity of the training set but also helps in making the model more robust to variations in the input data [8]. The choice between RGB and grayscale images is also significant, with grayscale images requiring adaptations in model architecture or input handling [7].

Comparison of Self-made Models to Pre-trained Models

The comparison between a self-made model and a pre-trained model provides valuable insights into the effectiveness of different architectures and training strategies. While a self-made model offers flexibility in design and customization, allowing for the development of a model tailored to specific requirements, pre-trained models have the advantage of extensive training on large datasets [3]. This extensive training can potentially reduce the risk of overfitting and accelerate the learning process, making pre-trained models a popular choice for many applications. The comparison between these two approaches is particularly insightful for tasks such as traffic sign recognition, where the adaptability

of the model to pre-processing techniques and the characteristics of the dataset play crucial roles in achieving high classification accuracy.

3 Experimental Setup

The dataset was preprocessed in several stages to optimize its performance with the CNN models. The preprocessing steps varied across different data files, each representing a unique combination of transformations applied to the images. These pre-processing techniques include shuffling, normalization, mean normalization, standard deviation normalization, grayscale conversion, and local histogram equalization. The aim of this study is to analyze the impact of these pre-processing techniques on model performance, questioning what impact they could play on the models.

Model Architecture and Training

A Convolutional Neural Network (CNN) model was trained on the preprocessed dataset. The architecture of the CNN model is shown in Table 1.

Table 1: CNN Model Architecture

Layer (type)	**Output Shape**	**Param #**
conv2d_24 (Conv2D)	(None, 64, 64, 16)	80
batch_normalization_30 (BatchNormalization)	(None, 64, 64, 16)	64
dropout_30 (Dropout)	(None, 64, 64, 16)	0
max_pooling2d_24 (MaxPooling2D)	(None, 32, 32, 16)	0
conv2d_25 (Conv2D)	(None, 32, 32, 32)	2,080
batch_normalization_31 (BatchNormalization)	(None, 32, 32, 32)	128
dropout_31 (Dropout)	(None, 32, 32, 32)	0
max_pooling2d_25 (MaxPooling2D)	(None, 16, 16, 32)	0
conv2d_26 (Conv2D)	(None, 16, 16, 64)	8,256
batch_normalization_32 (BatchNormalization)	(None, 16, 16, 64)	256
dropout_32 (Dropout)	(None, 16, 16, 64)	0
max_pooling2d_26 (MaxPooling2D)	(None, 8, 8, 64)	0
conv2d_27 (Conv2D)	(None, 8, 8, 128)	32,896
batch_normalization_33 (BatchNormalization)	(None, 8, 8, 128)	512
dropout_33 (Dropout)	(None, 8, 8, 128)	0
max_pooling2d_27 (MaxPooling2D)	(None, 4, 4, 128)	0
flatten_6 (Flatten)	(None, 2048)	0
dense_12 (Dense)	(None, 512)	1,049,088
batch_normalization_34 (BatchNormalization)	(None, 512)	2,048
dropout_34 (Dropout)	(None, 512)	0
dense_13 (Dense)	(None, 43)	22,059

The CNN model was trained on the RGB images (data0 - data3) and grayscale images (data4 - data8), with the architecture and hyperparameters optimized for this specific task. The training process involved adjusting the model's weights through backpropagation, using the Adam optimizer and categorical cross-entropy loss function to minimize the difference between the model's predictions and the actual labels. Early stopping and model checkpointing were employed to avoid overfitting and save the best performing model during training.

4 Results

Training and Validation Performance

The performance of the CNN models was evaluated on various preprocessed datasets. The models were trained on different datasets (Data0 - Data8), each with distinct pre-processing techniques applied.

The results of the training and validation phases are illustrated in the accuracy and loss graphs for each model:

Model Accuracy and Loss Graphs

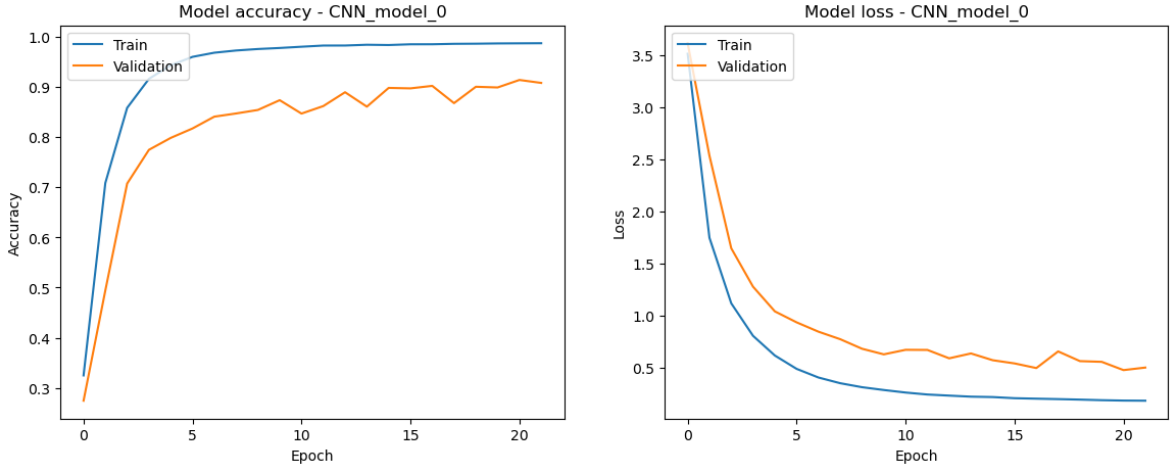

Figure 1: CNN Model 0 trained on Data0

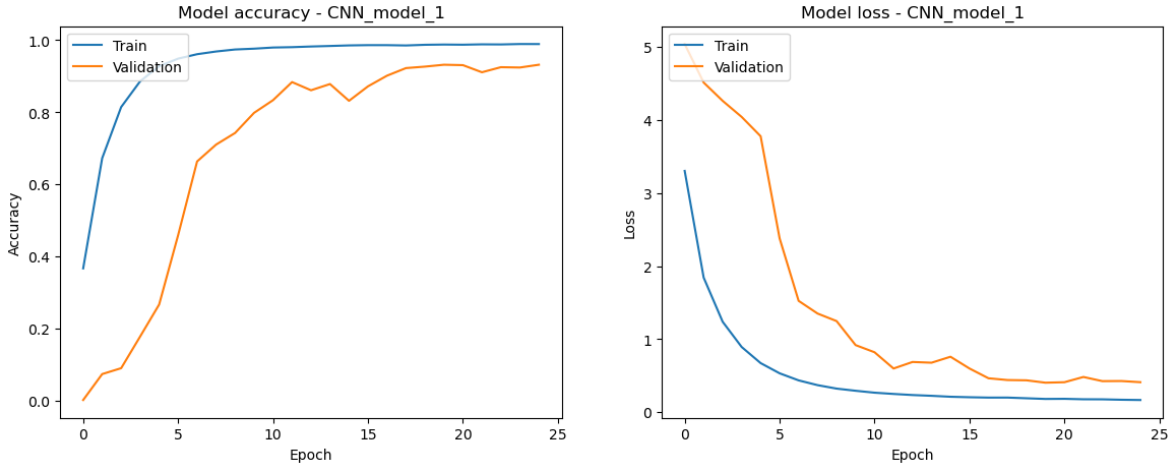

Figure 2: CNN Model 1 trained on Data1

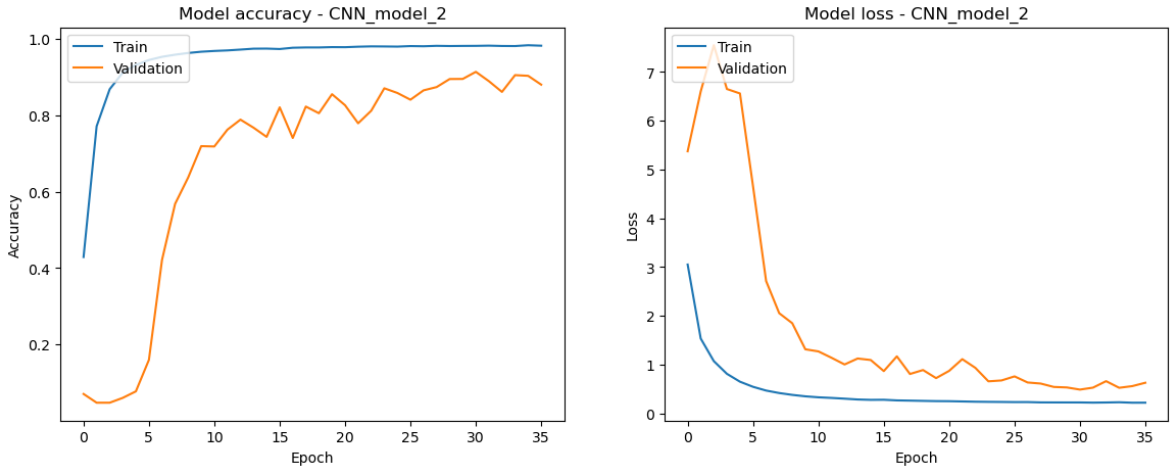

Figure 3: CNN Model 2 trained on Data2

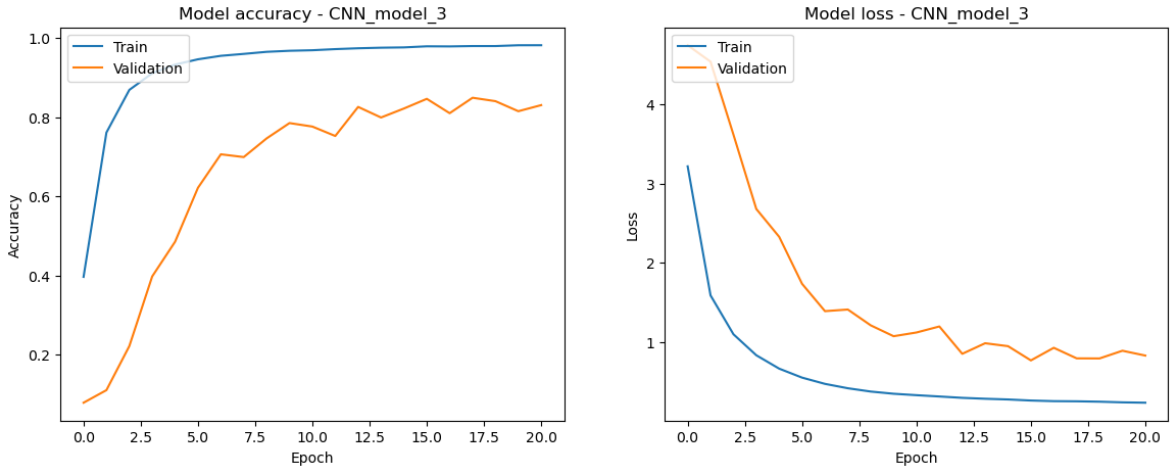

Figure 4: CNN Model 3 trained on Data3

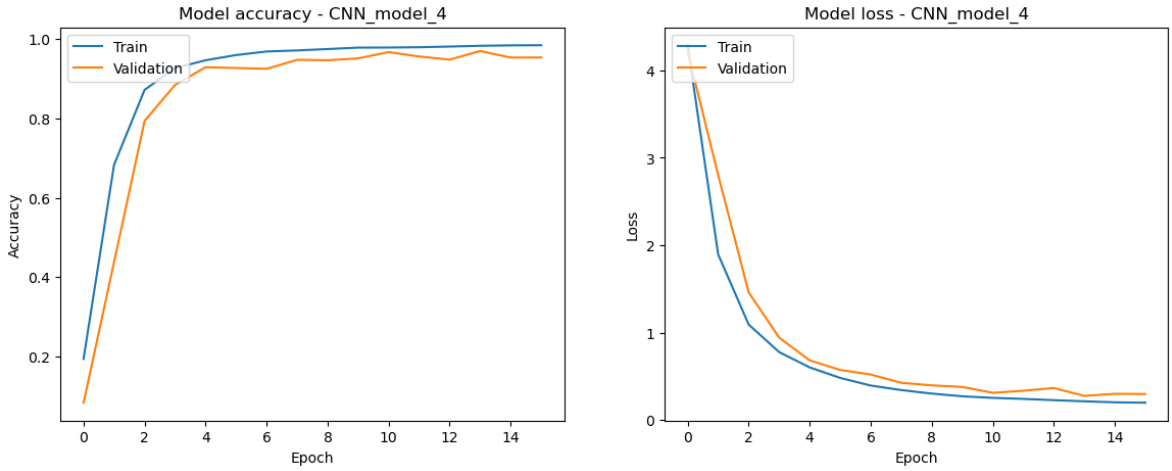

Figure 5: CNN Model 4 trained on Data4

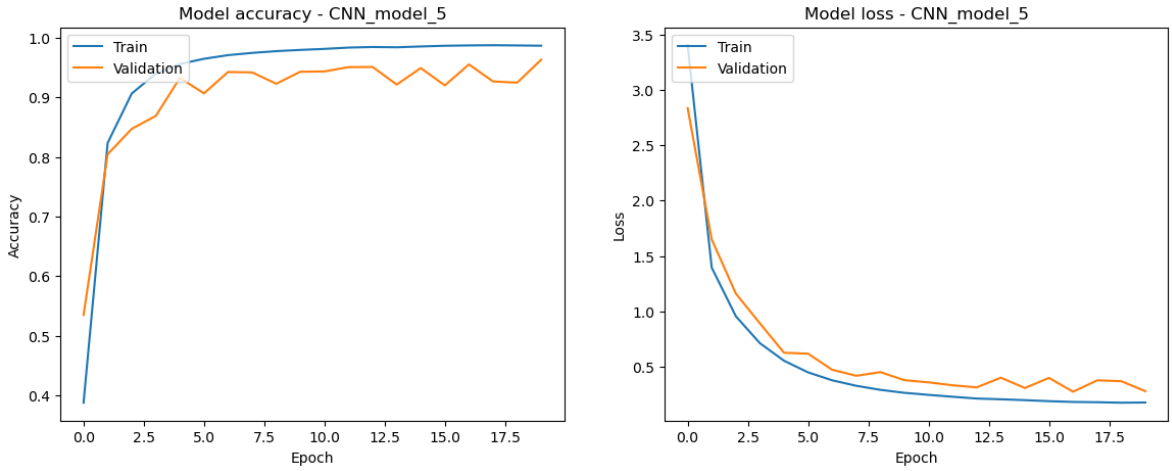

Figure 6: CNN Model 5 trained on Data5

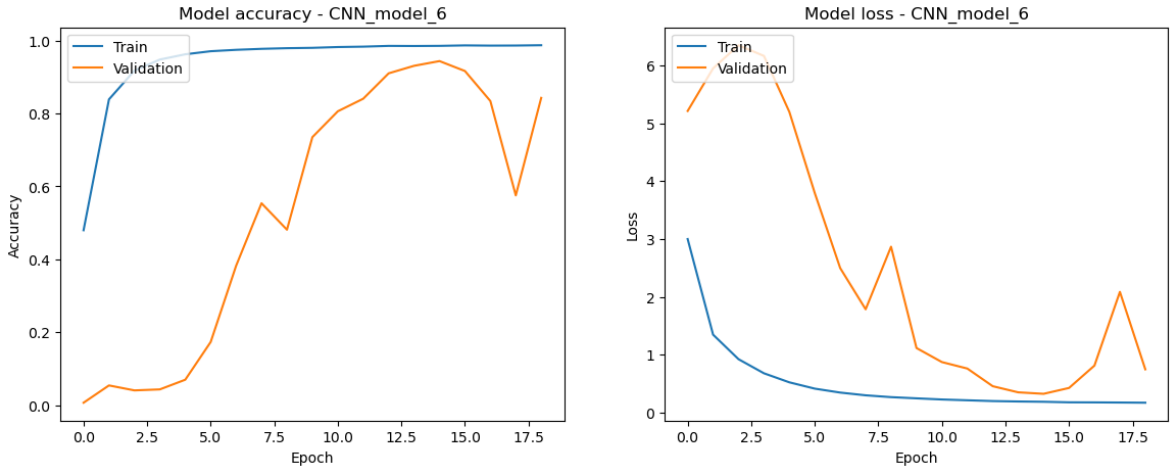

Figure 7: CNN Model 6 trained on Data6

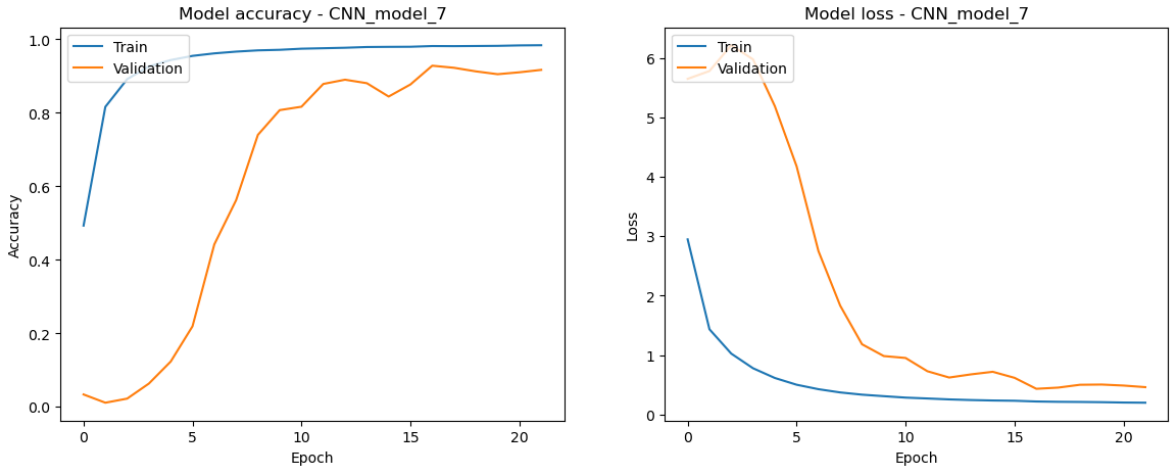

Figure 8: CNN Model 7 trained on Data7

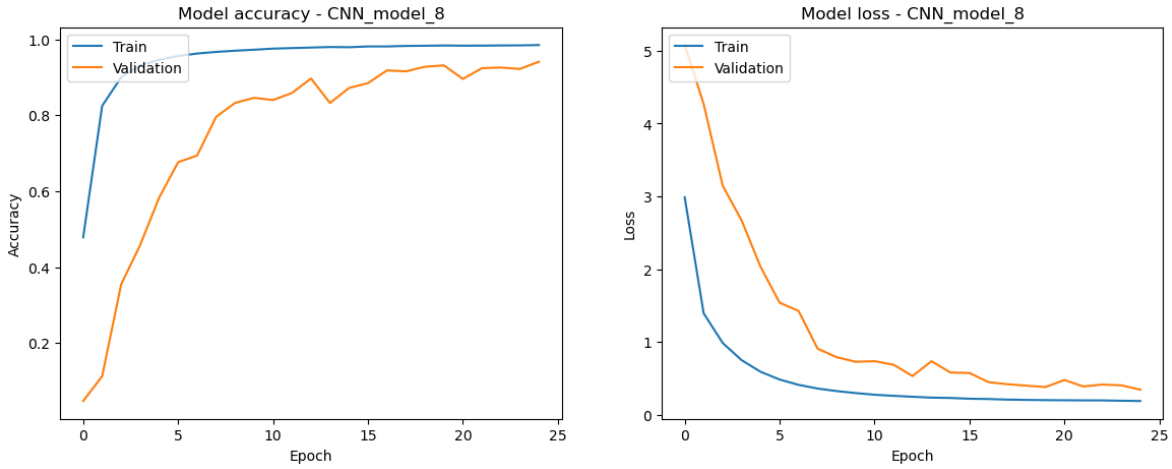

Figure 9: CNN Model 8 trained on Data8

The above graphs display the training and validation accuracy as well as the loss over epochs for each CNN model. Model 4, trained on Data4, exhibited the best performance with the highest validation accuracy and the lowest validation loss.

Evaluation Metrics

The evaluation metrics for each model, including Precision, Recall, and F1-Score, are presented below:

Model	**Precision**	**Recall**	**F1-Score**
Model 0	0.9266	0.9145	0.9121
Model 1	0.9404	0.9320	0.9307
Model 2	0.9251	0.9141	0.9135
Model 3	0.8779	0.8460	0.8408
Model 4	0.9730	0.9703	0.9698
Model 5	0.9652	0.9553	0.9566
Model 6	0.9490	0.9435	0.9398
Model 7	0.9456	0.9283	0.9303
Model 8	0.9571	0.9413	0.9405

Table 2: Performance Metrics for CNN Models

The best model was Model 4, with a Precision Score of 0.9730, Recall Score of 0.9703 and F1-Score of 0.9698.

Confusion Matrix for Model 4

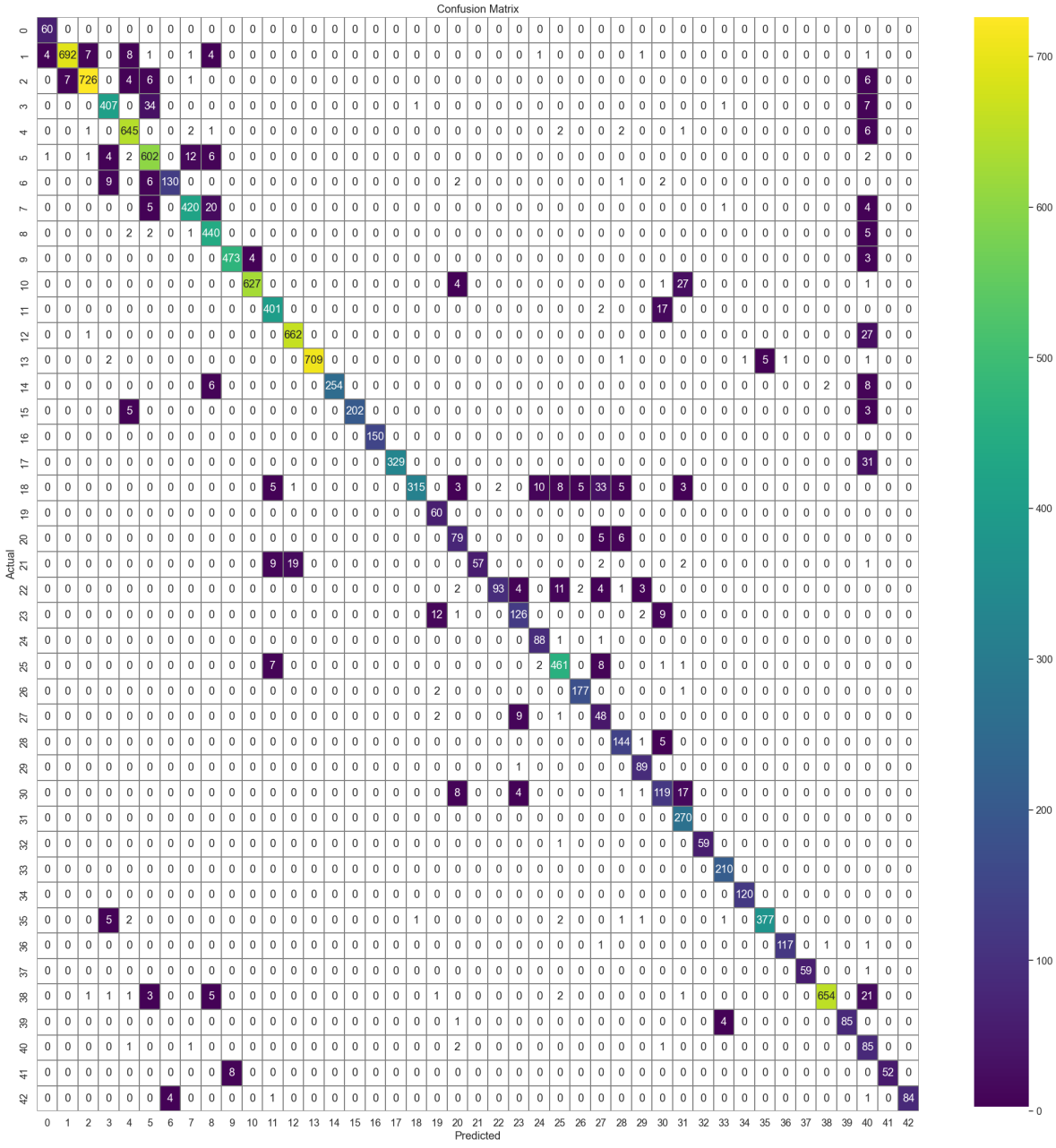

Figure 10: Confusion Matrix for Model 4

- The diagonal elements represent the correctly classified instances for each class, which should ideally be high for a well-performing model.
- Off-diagonal elements indicate misclassifications, which help identify specific classes where the model may be underperforming.
- Classes with high misclassifications may need further investigation to understand if certain features are confusing the model or if more data is needed for those classes.

Evaluation Metrics Across Test Sets:

The following figures illustrate the evaluation metrics (Precision, Recall, F1 Score, Test Loss, Test Accuracy) across various test sets:

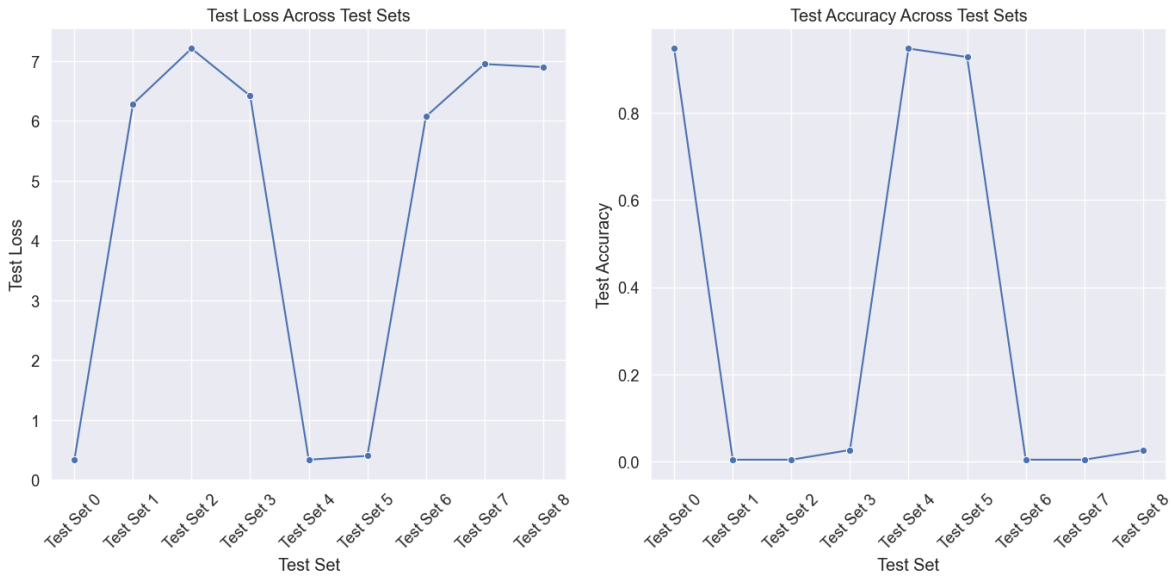

Figure 11: Test Loss/Accuracy Across Test Sets

- **Test Set 0, 4, and 5**: These sets show low test loss and high test accuracy, indicating that the model performs well and generalizes effectively on these datasets.
- **Test Set 1, 2, 6, 7, and 8**: These sets exhibit high test loss and correspondingly low test accuracy, suggesting that the model struggles to make accurate predictions on these datasets. This could be due to differences in pre-processing or data distribution that the model was not well-trained to handle.
- **Test Set 3**: This set shows moderate test loss and test accuracy, indicating some level of difficulty in making precise predictions, but not as poor as sets 1, 2, 6, 7, and 8.

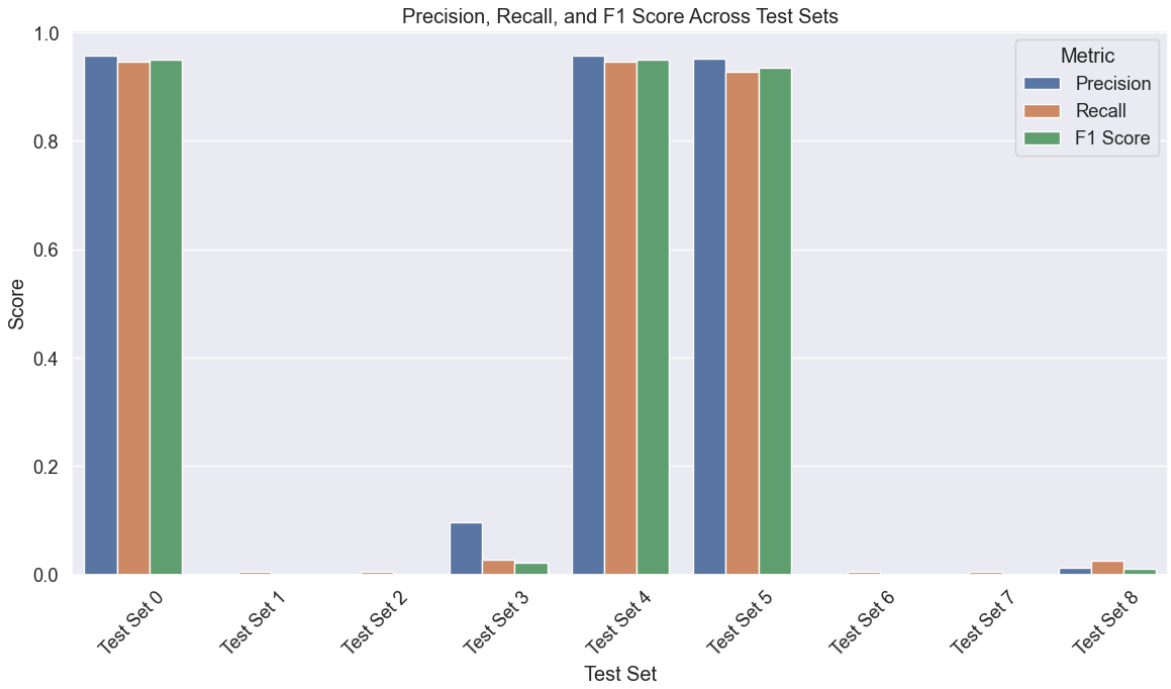

Figure 12: Precision, Recall, and F1 Score Across Test Sets

- **Test Set 0, 4, and 5**: These sets have high scores for precision, recall, and F1, indicating that the model is making accurate predictions with a good balance between precision (low false positives) and recall (low false negatives).
- **Test Set 1, 2, 6, 7, and 8**: These sets have significantly lower scores, showing poor performance with likely higher rates of false positives and false negatives.
- **Test Set 3**: This set shows lower but somewhat moderate scores compared to the best-performing sets, indicating moderate performance with room for improvement.

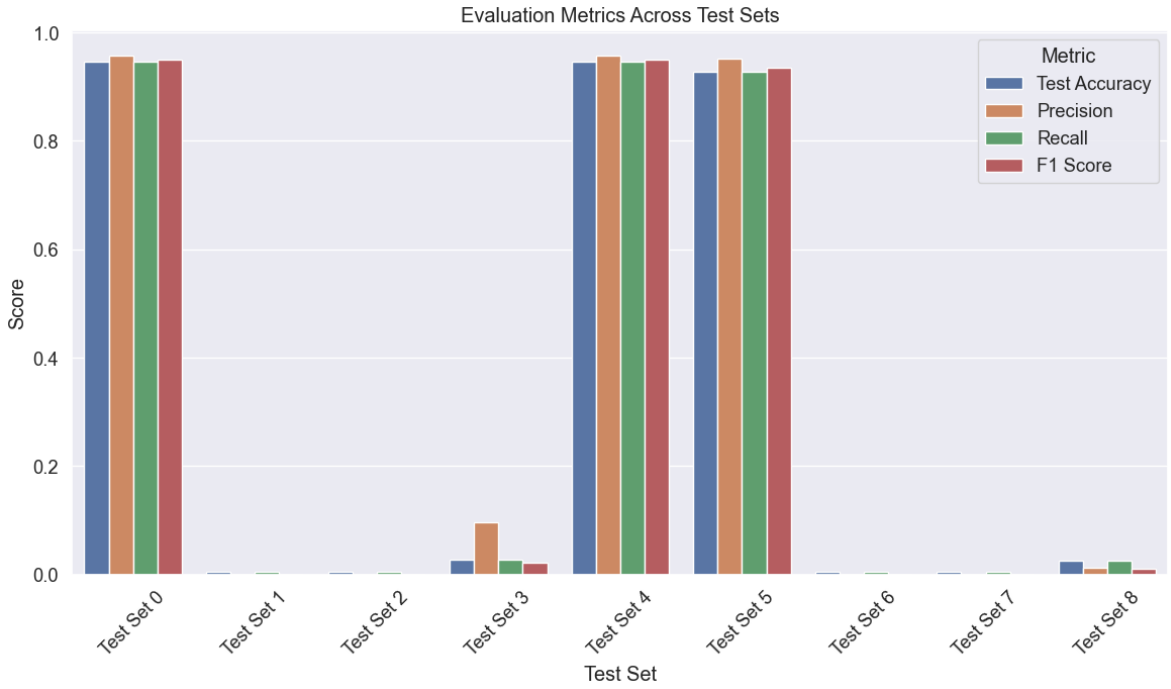

Figure 13: Evaluation Metrics Across Test Sets

Key Observations:

- **Test Set 0, 4, and 5**: Consistently high across all metrics, showing that the model is robust and performs well on these datasets.
- **Test Set 1, 2, 6, 7, and 8**: Consistently low across all metrics, indicating that the model has difficulty generalizing to these datasets.
- **Test Set 3**: Moderate performance across metrics, with some capability to generalize but still room for improvement.

5 Conclusion & Future Work

This study highlights the significant impact of pre-processing techniques on the performance of CNN models for traffic sign recognition. Among the various techniques explored, Model 4, trained on data preprocessed with the least amount of re-processing, Shuffling & greyscale conversion, proved to be the most effective, achieving the highest precision, recall, and F1-Score.

Future Work

Future work could focus on exploring additional pre-processing techniques and model architectures, possibly comparing a pre-trained model such as ResNet50, to further enhance the performance of image classification models. Investigating the impact of advanced data augmentation methods and transfer learning approaches could provide valuable insights. Additionally, experimenting with larger and more diverse datasets would help in validating the generalizability of the findings and improving the robustness of the models.

References

[1] He, K., Zhang, X., Ren, S., and Sun, J. (2016). *Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).*

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-based learning applied to document recognition*, Proceedings of the IEEE, 86(11):2278–2324, November 1998.

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, *ImageNet: A Large-Scale Hierarchical Image Database*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *ImageNet Classification with Deep Convolutional Neural Networks*, Advances in Neural Information Processing Systems 25 (NIPS 2012), 2012.

[5] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, *Face recognition: A convolutional neural-network approach*, IEEE Transactions on Neural Networks, 8(1):98–113, 1997.

[6] Journal of Big Data. (2021). *Pre-processing techniques in deep learning: A survey. Springer Open.* https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00444-8

[7] Masood, D. (2023). *Pre-train CNN architectures: Designs, performance analysis, and comparison. Medium.* https://medium.com/@daniyalmasoodai/pre-train-cnn-architectures-designs-performance-analysis-and-comparison-802228a5ce92

[8] Karen Simonyan and Andrew Zisserman, *Very Deep Convolutional Networks for Large-Scale Image Recognition*, in arXiv preprint arXiv:1409.1556, 2014.