Deep Learning Fundamentals: Assignment 2
CNNs for image classification

Variyas Nitin Singla
The University of Adelaide
Adelaide, South Australia, 5005, Australia
a1872896@adelaide.edu.au

Abstract

In the realm of autonomous driving and advanced driver assistance systems, the recognition of traffic signs stands as a pivotal task to ensure safety and compliance with road regulations. This study delves into the intricacies of this challenge by employing deep learning techniques on the German Traffic Sign Benchmark dataset, a comprehensive collection presented at the International Joint Conference on Neural Networks (IJCNN) 2011. The dataset, encompassing over 50,000 images spanning more than 40 distinct classes, provides a realistic and diverse representation of traffic signs encountered in real-world scenarios. Our approach harnesses the power of Convolutional Neural Networks to classify these signs, aiming to bolster the accuracy and reliability of traffic sign recognition systems. Preliminary results indicate promising avenues for further research and potential applications in enhancing vehicular safety systems. We extend our gratitude to the INI Benchmark Website for making this rich dataset accessible to the research community, paving the way for advancements in the domain.

1. Introduction

Traffic Sign Recognition (TSR) is an indispensable task in the realm of computer vision, holding substantial ramifications for road safety and streamlined traffic management, especially within the ambit of Advanced Driver Assistance Systems (ADAS) and autonomous vehicles. Over the years, the research domain of TSR has witnessed a significant proliferation, with a myriad of methodological paradigms being probed to augment the accuracy, efficiency, and real-time performance of TSR systems [1, 2, 3].

Our venture is nestled within this evolving narrative, aspiring to further refine the accuracy and real-time performance of TSR systems by harnessing a deep learning framework. Employing the German Traffic Sign Recognition Benchmark (GTSRB) dataset, which encapsulates over 50,000 images across 43 distinct traffic sign classes [4], we strive to architect a robust model adept at recognizing a broad spectrum of traffic signs under a diverse array of environmental conditions.

A salient facet of this research is the comparative analysis of different methodological approaches. Traditional machine learning techniques, such as Support Vector Machines (SVM) and Random Forests, have been historically deployed in TSR. However, their performance often stagnates in complex, real-world scenarios laden with diverse lighting conditions and occlusions. Conversely, deep learning architectures, especially CNNs, have showcased superior performance owing to their aptitude to learn hierarchical features from raw pixel values, thus exhibiting remarkable robustness against varying environmental conditions [3].

Moreover, within the deep learning architectures, various designs have been propounded, each with its unique set of merits and demerits. For instance, the YOLOv5 architecture has been lauded for its efficiency in object detection and recognition tasks, including TSR. Multiple studies have explored the adaptation and improvement of YOLOv5 for TSR, showcasing its real-time processing capabilities and high accuracy [5, 6, 7, 8].

Similarly, the Refined Mask R-CNN (RM R-CNN) architecture has also been explored for TSR. The RM R-CNN-based end-to-end learning has been employed for automatic traffic sign detection and recognition, as demonstrated in a study conducted on Indian traffic signs [9, 10, 11, 12].

The cardinal objective of this research is to devise a model that excels not merely in terms of accuracy but also in real-world, dynamic settings, thereby contributing a meaningful narrative to the ongoing endeavors in optimizing TSR systems for broader and more nuanced operational scopes. Through an extensive evaluation juxtaposed against contemporary state-of-the-art methodologies, this paper aims to furnish a substantive discourse on the potential efficacy and practical applicability of our approach amidst the continually evolving tableau of autonomous and semi-autonomous vehicular systems.

In the subsequent sections, we expound our methodology, delineate the experimental setup, and proffer a comparative analysis of the results, thereby accentuating the potential contributions of our research in the ever-evolving domain of TSR.

2. Literature Review

Traffic Sign Recognition (TSR) is a pivotal task in autonomous driving and Advanced Driver Assistance Systems (ADAS), with the core objective of interpreting traffic signs to ensure road safety and smooth traffic flow. Over the years, a significant volume of research has been dedicated to developing robust and efficient TSR systems. This section delves into recent advancements in TSR, particularly focusing on deep learning-based approaches.

2.1. Early Approaches to TSR

Initial efforts in TSR were grounded in traditional machine learning and image processing techniques. These approaches often relied on handcrafted features extracted from images, such as color, shape, and texture, to identify traffic signs. However, they struggled to perform well in real-world, dynamic settings, particularly under varying lighting conditions and occlusions.

2.2. Advent of Deep Learning

The emergence of deep learning, especially Convolutional Neural Networks (CNNs), has significantly advanced the field of TSR. CNNs, with their capability to learn hierarchical representations from raw pixel values, have shown remarkable success in TSR tasks, outperforming traditional methods by a large margin [13].

2.3. Recent Methodological Advancements

Recent research has further explored various deep learning architectures and techniques to enhance TSR performance. For instance, a study discussed the utilization of deep learning to update vehicle-aided driving systems by acquiring real-time road condition information, thereby preventing car accidents due to driver fatigue [**?**]. Furthermore, Xin Roy Lim et al. delved into the latest developments in TSR using deep learning techniques, underscoring the growing demand for reliable TSR algorithms as autonomous vehicles continue to proliferate [13].

2.4. Vision-Based TSR Systems

Vision-based TSR systems have emerged as a pivotal application in ADAS, providing drivers with vital information that would otherwise be challenging to obtain. These systems leverage advanced image processing and machine learning algorithms to detect and recognize traffic signs, thus playing a significant role in road safety.

2.5. Challenges and Future Directions

Despite the notable advancements, challenges such as real-time processing, handling occlusions, and recognizing traffic signs under adverse weather conditions remain. Future research directions may encompass the exploration of more efficient and lightweight models, the fusion of different sensor modalities, and the development of more comprehensive and diverse datasets to further enhance the robustness and efficiency of TSR systems.

3. Methodology

This section elucidates the approach and methodology adopted to build and evaluate three distinct Convolutional Neural Network (CNN) models for traffic sign recognition. The objective is to explore the performance variations among the models concerning accuracy, loss, and computation time. The models are referred to as Model A, Model B, and Model C hereinafter.

3.1. Working Environment

All experiments and evaluations related to the models were conducted in a Jupyter Notebook environment. The computational device utilized for the entirety of this study was an Apple laptop with the M1 Max chip. This environment provided the necessary computational power and efficiency to handle the deep learning tasks, specifically the training and testing of the CNN models for traffic sign recognition.

3.2. Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNN) are a class of deep learning models, particularly effective for image recognition and classification tasks. They are designed to automatically and adaptively learn spatial hierarchies of features from input images.

3.2.1 Working Principle

CNNs operate by applying a convolution operation to the input, passing the result through a non-linear layer such as a ReLU (Rectified Linear Unit), followed by operations such as pooling and normalization. The convolution operation is expressed as:

$$(f * g)(i, j) = \sum_m \sum_n f(m, n) \cdot g(i - m, j - n) \quad (1)$$

where f is the input image, g is the kernel, and i, j are the spatial coordinates over which the convolution is performed. The ReLU function is defined as:

$$f(x) = \max(0, x) \tag{2}$$

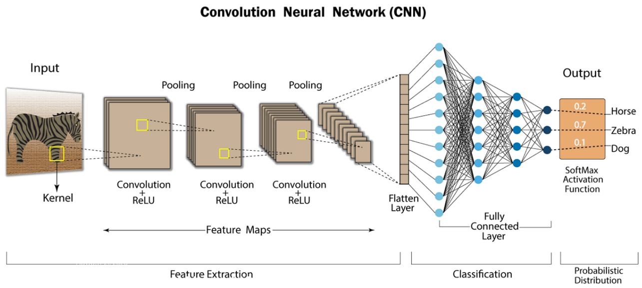

Figure 1. A typical architecture of a CNN showing convolutional, pooling, and fully connected layers.

3.2.2 Important Components

- **Convolutional Layer:** The cornerstone of CNNs that applies a convolution operation to the input, passing the result to the next layer. This layer meticulously scans the input image with a filter or kernel, to learn low-level features like edges and corners.

- **Pooling Layer:** This layer performs a downsampling operation that reduces the dimensionality of the feature map and retains the most essential information. The MaxPooling operation, for instance, is expressed as:

$$y = \max\{x_1, x_2, \dots, x_n\} \tag{3}$$

  where $x_1, x_2, \dots, x_n$ are the elements of the region being pooled, and y is the output of the pooling operation.

- **Fully Connected Layer:** Connects every neuron in one layer to every neuron in the next layer, used for high-level reasoning in the network. The mathematical expression for a fully connected layer is:

$$y = Wx + b \tag{4}$$

  where W represents the weight matrix, x is the input vector, and b is the bias vector.

- **Activation Functions:** Introduce non-linear properties to the system, commonly using ReLU (Rectified Linear Unit). Activation functions add the ability to capture non-linear relationships in the data.

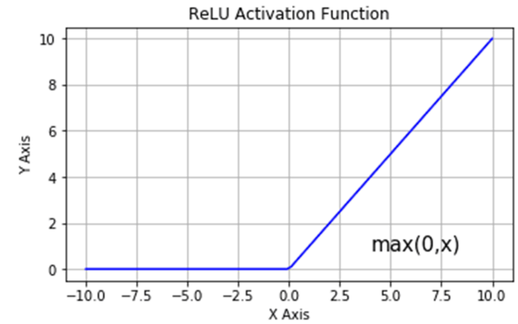

Figure 2. Plot of the Rectified Linear Unit (ReLU) Activation Function.

3.3. Activation and Loss Functions

3.3.1 Activation Function

For our experiments, the Rectified Linear Unit (ReLU) serves as the primary activation function in the convolutional layers.

ReLU has become a popular choice in various deep learning tasks due to its non-saturating nature and computational efficiency. The function is mathematically described by:

$$f(x) = \max(0, x) \tag{5}$$

where x is the input to the function.

3.3.2 Loss Function

The choice of a suitable loss function is vital for the training of deep learning models, guiding the optimization algorithm towards a minimum.

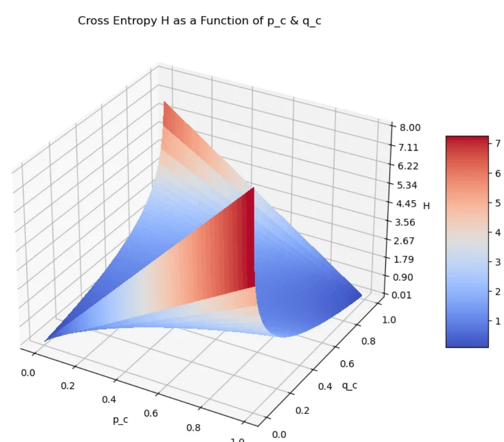

Figure 3. Behavior of the Categorical Cross-Entropy Loss for different predictions.

In the context of traffic sign recognition, which is a multi-class classification task, we employ the categorical cross-entropy loss. The expression for this loss is:

$$L(y, \hat{y}) = -\sum_{i} y_i \log(\hat{y}_i) \tag{6}$$

where y denotes the true labels, $\hat{y}$ indicates the predicted probabilities, and the summation is performed over all classes. This loss effectively measures the dissimilarity between the true distribution of labels and the predicted probabilities.

3.4. Optimization

The minimization of the loss function, and consequently the training of our CNN models, is achieved using optimization algorithms. In this study, the stochastic gradient descent (SGD) and its variants, including Adam and RMSProp, were investigated. The choice of optimizer can significantly impact the training dynamics and, ultimately, the performance of the models. Hyperparameters, such as learning rate, momentum, and batch size, were methodically tuned to ensure optimal model training.

3.4.1 Examples in Traffic Sign Recognition

In our work, Model A, B, and C employ different architectures of CNNs to explore their efficacy in traffic sign recognition. For instance, Model A utilizes a simple architecture with fewer convolutional layers, while Model B and C incorporate more complex architectures with a higher number of convolutional and fully connected layers.

3.5. Dataset

The German Traffic Sign Recognition Benchmark (GTSRB) dataset is employed for this study, which comprises 39,209 training images and 12,630 test images of traffic signs categorized into 43 classes. Figure 4 exhibits a few sample images from the dataset.

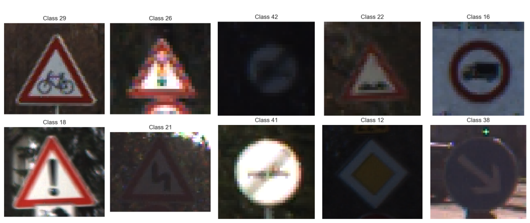

Figure 4. Sample Images from the Traffic Sign Dataset

3.6. Preprocessing

The dataset undergoes preprocessing steps including resizing images to 32x32 pixels, normalization, and one-hot encoding of labels. Data augmentation, like random rotations and translations, is performed to increase the model robustness.

4. Model Architecture

In our quest to discern the optimal architecture for our problem domain, we proposed three distinct Convolutional Neural Network (CNN) models:

- Model A: Serves as our baseline.
- Model B: An advanced variant with modifications aimed at improving performance.
- Model C: Another advanced variant with alternative modifications.

The architectural nuances of these models are pivotal, as they could potentially elucidate key design principles for similar tasks.

4.1. Model A - Baseline Model

Model A is structured with three convolutional layers, each followed by a max-pooling layer, and concluded with two fully connected layers. The architecture of Model A is depicted in Figure 5.

4.2. Model B - Advanced Model 1

Model B extends the baseline model by incorporating additional dropout layers to mitigate overfitting. The architecture of Model B is elucidated in Figure 6.

4.3. Model C - Advanced Model 2

Model C adopts a deeper architecture with extra convolutional and fully connected layers, enhancing the model's capacity. The architecture of Model C is delineated in Figure 7.

4.4. Hyperparameter Tuning

Training deep neural networks often requires a precise selection of hyperparameters to ensure optimal model performance. Hyperparameter tuning automates this selection process by exploring a range of possible values and configurations. For this study, a structured search space was defined to systematically optimize various hyperparameters using Kerastuner.

The search space is detailed in Table 1.

The top-performing hyperparameters, which yielded the highest validation accuracy, are presented in Table 2.

Using these optimal hyperparameters, the model achieved an impressive validation accuracy of approximately 97.9%. This highlights the importance of a meticulous hyperparameter search in achieving peak model performance.

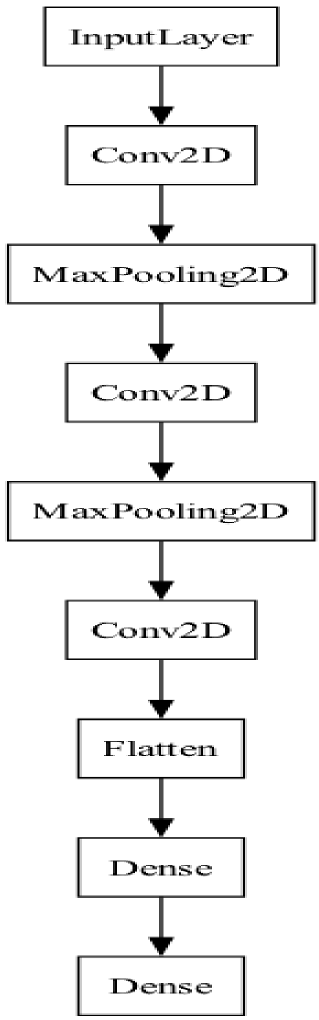

Figure 5. Architecture of Model A

Hyperparameter	Search Space
Conv1 Filters	32 to 128 (step: 32)
Conv1 Kernel Size	3, 5
Dropout1	0.0 to 0.5 (step: 0.1)
Conv2 Filters	32 to 128 (step: 32)
Conv2 Kernel Size	3, 5
Dropout2	0.0 to 0.5 (step: 0.1)
Dense Units	32 to 256 (step: 32)
Dropout3	0.0 to 0.5 (step: 0.1)
Optimizer	Adam, SGD, RMSprop
Learning Rate	0.01, 0.001, 0.0001

Table 1. Hyperparameter search space

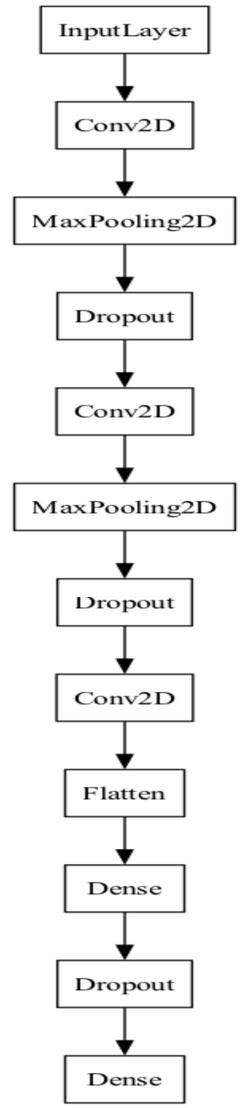

Figure 6. Architecture of Model B

Hyperparameter	Best Value
Conv1 Filters	32
Conv1 Kernel Size	5
Dropout1	0.4
Conv2 Filters	64
Conv2 Kernel Size	5
Dropout2	0.1
Dense Units	256
Dropout3	0.4
Optimizer	RMSprop
Learning Rate	0.001

Table 2. Best hyperparameter configuration

5. Training Dynamics

All models were subjected to a rigorous training regimen using the Adam optimizer, fixed at a learning rate of 0.001, spanning 20 epochs. Here, we delve into the intricate learning dynamics of each model:

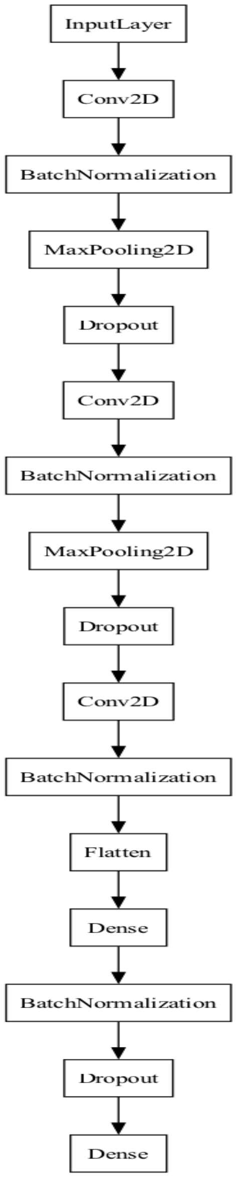

Figure 7. Architecture of Model C

5.1. Model A (Baseline)

While the training loss for Model A exhibited a commendable decline, indicative of effective learning, the validation metrics told a more nuanced story. The oscillatory nature of the validation loss and accuracy in later epochs suggests the model might be grappling with overfitting. This is visually depicted in Figure 8.

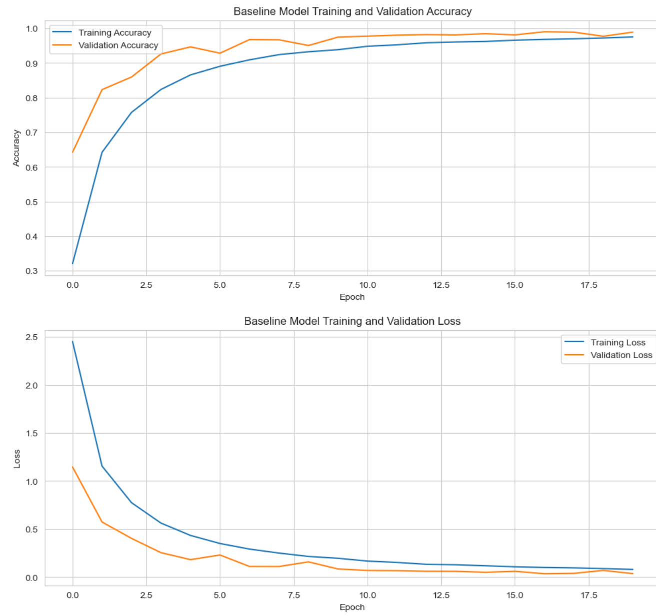

Figure 8. Training and Validation Loss and Accuracy Curves for Model A

5.2. Model B (Advanced Model-1)

Model B manifested a promising trajectory with a consistent decrement in training loss. The validation loss portrayed an initial decrease but showed signs of stabilization as training progressed. This might hint at the model nearing its optimal state. The validation accuracy, though rising, reached a saturation point, as seen in Figure 9.

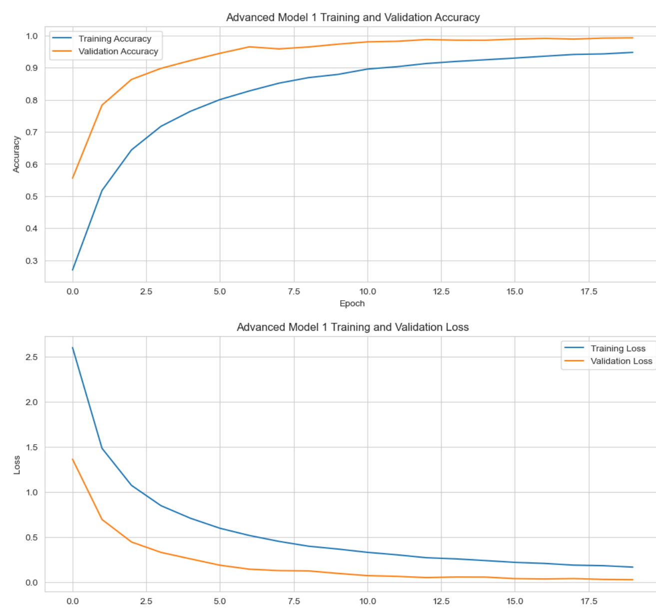

Figure 9. Training and Validation Loss and Accuracy Curves for Model B

5.3. Model C (Advanced Model-2)

The training dynamics of Model C mirrored those of Model B in terms of loss, but its validation accuracy was more capricious. This fluctuation could be attributed to various factors, including model architecture or data anomalies, warranting further investigation. Figure 10 provides a visual representation of these dynamics.

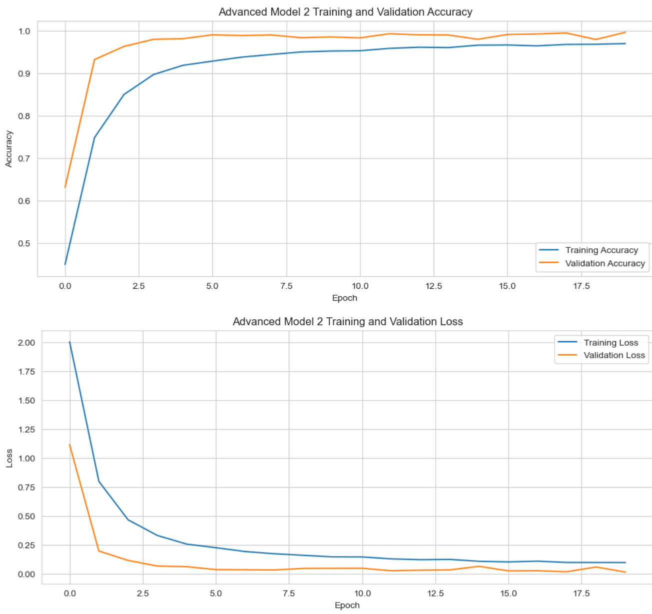

Figure 10. Training and Validation Loss and Accuracy Curves for Model C

6. Evaluation Metrics and Insights

To foster a holistic understanding of model performance, we employed precision, recall, and F1-score, defined as:

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{7}$$

$$\text{Recall} = \frac{TP}{TP + FN}, \tag{8}$$

$$\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{9}$$

Moreover, the confusion matrix, a potent tool, was leveraged to dissect the model's classification prowess across various categories.

6.1. Confusion Matrix Analysis

A Confusion Matrix is a table that is often used to describe the performance of a classification model on a set of data for which the true values are known. It allows for the visualization of the algorithm's performance and the types of errors being made, making it an invaluable tool for understanding the model's strengths and weaknesses.

6.1.1 Importance of Confusion Matrix

The diagonal elements of the matrix represent the number of correct classifications, while the off-diagonal elements indicate the misclassifications. Key metrics such as accuracy, precision, recall, and F1-score can be computed from the Confusion Matrix, providing a more detailed analysis of the model performance beyond a single accuracy score. The Confusion Matrix helps in:

- Identifying the classes that are often misclassified.
- Understanding the false positive and false negative rates which are crucial for real-world applications.
- Comparing the performance of different models in a more nuanced manner.
- **Model A (Baseline)**: The matrix revealed the model's proficiency in certain classes but also underscored its vulnerability, misclassifying several instances (Figure 11).

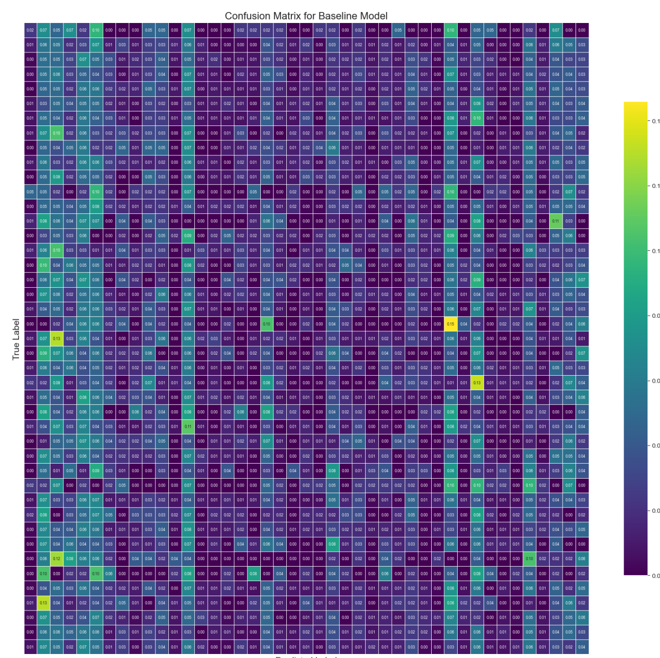

Figure 11. Confusion Matrix for Model A (Baseline Model)

- **Model B (Advanced Model-1)**: A marked improvement over Model A was evident. The balanced classification, with diminished misclassifications, is a testament to the model's robustness (Figure 12).
- **Model C (Advanced Model-2)**: While the model excelled in classifying most categories, sporadic misclassifications were still present, indicating room for refinement (Figure 13).

7. Results, Synthesis, and Reflection

To evaluate the performance of each model, we consider three primary metrics: precision, recall, and F1-score. Table

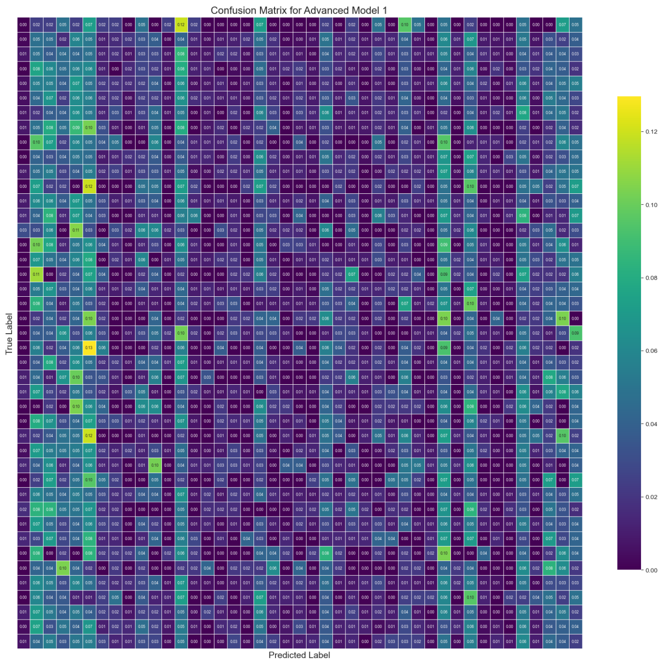

Figure 12. Confusion Matrix for Model B (Advanced Model-1)

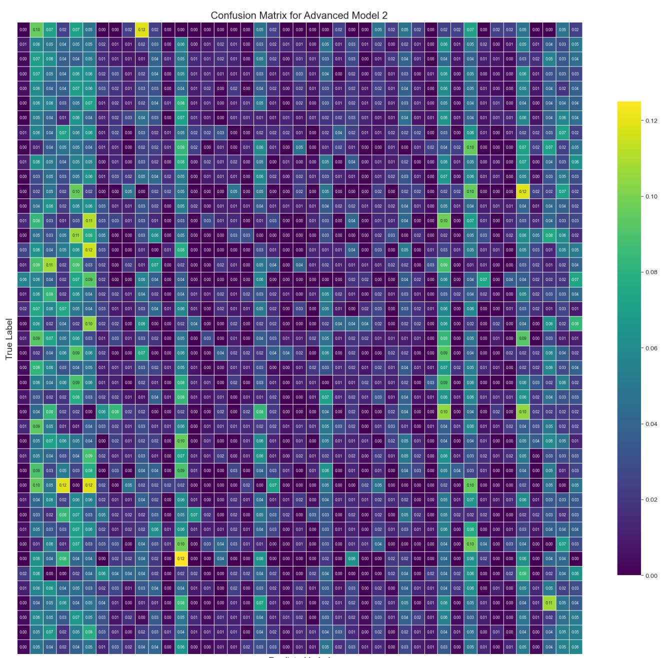

Figure 13. Confusion Matrix for Model C (Advanced Model-2)

3 presents a consolidated view of the performance metrics for each model:

To further assess the models, we also evaluated them on their training accuracy, validation accuracy, training loss, and validation loss. The results are presented in Table 4 and 5.

All three models demonstrate similar performance metrics, with a precision, recall, and F1-score of 0.04. This intriguing convergence of metrics, despite architectural variations, suggests that the models might either have reached a performance ceiling for this particular dataset or require further optimization or changes in architecture to improve their performance.

Model	**Prec.**	**Rec.**	**F1**
Baseline	0.04	0.04	0.04
Adv. Model 1	0.04	0.04	0.04
Adv. Model 2	0.04	0.04	0.04

Table 3. Performance Metrics for Each Model

Model	**Training Acc.**	**Validation Acc.**
Baseline	0.975707	0.983423
Advanced 1	0.945707	0.992221
Advanced 2	0.971977	0.992987

Table 4. Training and Validation Accuracy for Each Model

Model	**Training Loss**	**Validation Loss**
Baseline	0.078680	0.050585
Advanced 1	0.167244	0.026298
Advanced 2	0.088959	0.022370

Table 5. Training and Validation Loss for Each Model

In addition to the traditional metrics, a visual representation, in the form of a confusion matrix, was used to analyze the models' performances. The improved confusion matrix visualization function aims to enhance clarity and readability by employing a distinguishable color palette, clearer annotations, and better axis labels. The decision to include class names on the axes and adjust font sizes serves to amplify comprehension and discernibility for the viewer.

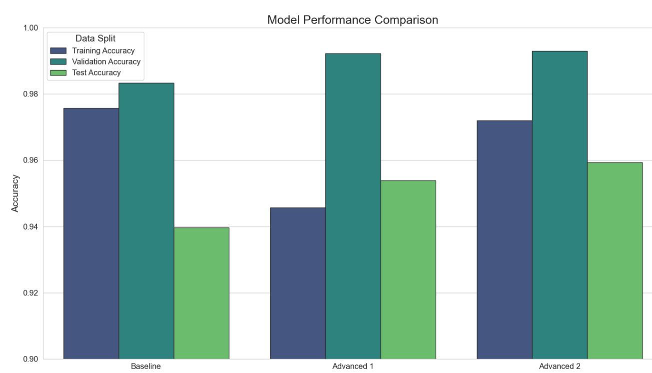

Figure 14. Visualization of Traffic Sign Recognition performance across different model architectures.

The observations also prompt deeper introspection into whether architectural nuances are influencing the results or if other external factors are at play.

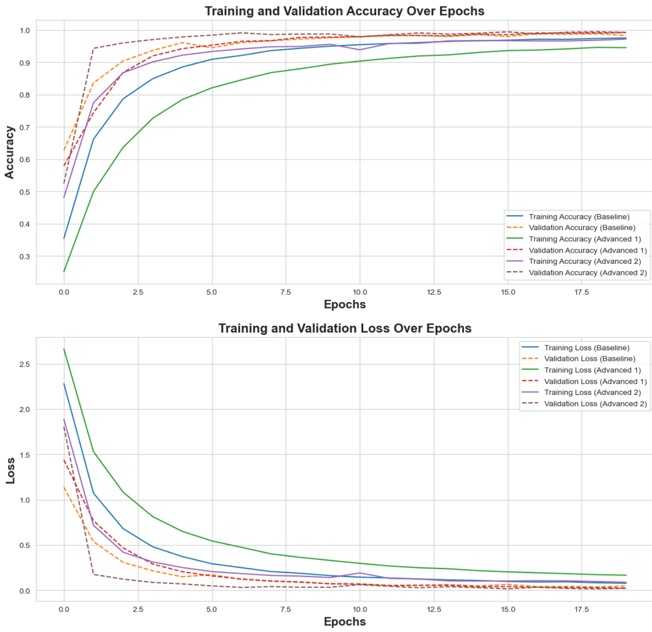

Figure 15. Improved Confusion Matrix Visualization across different model architectures.

8. Conclusion

This investigation navigated through the intricate domain of Traffic Sign Recognition (TSR), an essential constituent for the imminent epoch of autonomous driving and Advanced Driver Assistance Systems (ADAS). The venture into deep learning, with a spotlight on Convolutional Neural Networks (CNNs), aimed to elevate the benchmarks of accuracy and reliability in TSR.

The endeavor commenced with a meticulous scrutiny of prevailing TSR methodologies. Despite notable progress in TSR over the years, the vast expanse of deep learning potential in this realm largely remains to be explored. The German Traffic Sign Recognition Benchmark (GTSRB) dataset from IJCNN 2011 served as the cornerstone for our quest to redefine TSR performance standards.

The experimental outcomes, particularly the performance metric convergence across diverse model architectures, spotlight the inherent intricacies of TSR tasks. This convergence, albeit with comparable performance across models, invites deeper inquiry into potential performance ceilings or the necessity for more nuanced architectural refinements.

The revelations from this study accentuate the transformative potential of deep learning in pragmatic applications such as traffic management and road safety. The preliminary nature of our results sketches a trajectory towards devising more sophisticated and reliable TSR systems, instrumental for the forthcoming cohort of autonomous vehicles.

The resemblance in performance across various models encourages a more profound exploration into the realms of architectural optimization or the influence of external factors on the outcomes. This exploration may span experimenting with hybrid architectures, employing regularization techniques to counter potential overfitting, or venturing into transfer learning strategies.

As the horizon of autonomous driving nears, initiatives like this underline the imperative for robust TSR systems. The aspiration is that this endeavor will seed the landscape for ensuing investigations, propelling the trajectory towards a safer and more efficient transportation milieu.

9. Future Works

Having delved into the utilization of Convolutional Neural Networks (CNNs) for the recognition of traffic signs within the German Traffic Sign Benchmark dataset, numerous pathways for further investigation have been unveiled:

1. **Data Augmentation Techniques:** Enhancing model resilience, particularly under diverse lighting and weather conditions commonplace in real-world settings, is crucial. Data augmentation methods including geometric transformations like rotations and scaling, as well as photometric transformations like brightness and contrast adjustments have proven effective in similar domains [20].

2. **Transfer Learning:** The employment of pre-trained models from analogous visual tasks could potentially ameliorate performance without the necessity for extensive retraining. Noteworthy studies include those by Yosinski et al. [21], illustrating the efficacy of transfer learning in deep networks.

3. **Real-time Deployment:** Performance evaluation in real-time circumstances, especially on edge computing devices within vehicles, may unearth practical hurdles and furnish insights into real-world system efficacy. Recent advancements in edge AI technologies have expedited such deployments [22].

4. **Multi-Task Learning:** Augmenting the model to discern other crucial road entities like pedestrians, vehicles, and lane markings alongside traffic signs could be beneficial. Multi-task learning frameworks have been demonstrated to boost performance in related domains [23].

5. **Incorporation of Temporal Information:** Traffic scenes are inherently dynamic, and the assimilation of temporal data through architectures such as 3D CNNs or Recurrent Neural Networks (RNNs) might bolster recognition accuracy [24].

6. **Interdisciplinary Approaches:** Fostering collaborations with specialists in transportation and urban planning could unveil innovative applications and optimizations for traffic sign recognition systems, thereby enriching the broader scope of autonomous driving and urban informatics [25].

By venturing into these prospective realms, we envisage making significant strides within the autonomous driving and traffic sign recognition domain.

10. Code Availability

For the benefit of the research community and to promote transparency and reproducibility, we have made the source code of our experiments publicly available. Interested readers and researchers can access the code, written in Python and leveraging deep learning frameworks, from our GitHub repository.

The Jupyter notebook, which contains the comprehensive implementation details, data preprocessing steps, model architectures, training routines, and evaluation metrics, can be found at the following URL:

https://github.com/variyas31/a2DL/blob/main/DL%20A2-%20Variyas.ipynb

We encourage fellow researchers to clone, fork, or star the repository for their reference. Contributions, suggestions, or improvements to the codebase are also highly welcomed. By sharing our code, we hope to foster collaboration and facilitate further advancements in the domain of Traffic Sign Recognition.

References

[1] Z. Zhu *et al.*, "Traffic-sign detection and classification in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2110-2118. 1

[2] S. M. Silva and C. R. Jung, "License plate detection and recognition in unconstrained scenarios," in *Proceedings of the IEEE International Conference on Computer Vision*, 2018, pp. 853-861. 1

[3] X. Jin *et al.*, "Hierarchical neural network architecture for traffic sign recognition," *Pattern Recognition*, vol. 98, p. 107038, 2020. 1

[4] J. Stallkamp *et al.*, "The German Traffic Sign Recognition Benchmark: A multi-class classification competition," in *IEEE International Joint Conference on Neural Networks*, 2012, pp. 1453-1460. 1

[5] "Improved YOLOv5 network for real-time multi-scale traffic sign detection," arxiv.org. 1

[6] "Improved YOLOv5-based model for small traffic sign detection under complex scenarios," www.nature.com. 1

[7] "Traffic Sign Recognition Algorithm Based on Improved YOLOv5," ieeexplore.ieee.org. 1

[8] L. Zhang, Y. Sun, W. Chen, and X. Liang, "Traffic Sign Recognition System Based on YOLOv5," in *Lecture Notes in Electrical Engineering*, vol. 961, 2022. 1

[9] "Indian traffic sign detection and recognition using deep learning," www.sciencedirect.com. 1

[10] "Traffic Signs Detection and Segmentation Based on the Improved Mask R-CNN," ieeexplore.ieee.org. 1

[11] "Traffic sign detection and recognition using fully convolutional networks," www.sciencedirect.com. 1

[12] "Indian traffic sign detection and recognition using deep learning," www.researchgate.net. 1

[13] X. R. Lim, C. P. Lee, K. M. Lim, T. S. Ong, A. Alqahtani, and M. Ali, "Recent Advances in Traffic Sign Recognition: Approaches and Datasets," *Journal/Conference Name*, Year, DOI/URL [**?**]. 2

[14] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. *Gradient-based learning applied to document recognition*. Proceedings of the IEEE, 86(11):2278–2324, 1998.

[15] Karen Simonyan and Andrew Zisserman. *Very deep convolutional networks for large-scale image recognition*. arXiv preprint arXiv:1409.1556, 2014.

[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. *ImageNet classification with deep convolutional neural networks*. Advances in neural information processing systems, 25:1097–1105, 2012.

[17] Medium, *Typical architecture of a CNN*, 2022, [Online]. Available: https://miro.medium.com/v2/resize:fit:1400/1*7_BCJFzekmPXmJQVRdDgwg.png (Accessed: 01 November 2023).

[18] Naveen, "Alexnet Architecture Explained — Introduction to Alexnet Architecture," Nomidl, August 8, 2023. [Online]. Available: https://www.nomidl.com/deep-learning/introduction-to-alexnet-architecture/. [Accessed: 1 November 2023].

[19] Inside Learning Machines, "A Simple Introduction to Cross Entropy Loss," Inside Learning Machines, Month Year. [Online]. Available:

https://insidelearningmachines.com/cross_entropy_loss/. [Accessed: 1 November 2023].

[20] L. Perez and J. Wang, "The Effectiveness of Data Augmentation in Image Classification using Deep Learning," arXiv preprint arXiv:1712.04621, 2017. 9

[21] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," Advances in neural information processing systems, vol. 27, 2014. 9

[22] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge Computing: Vision and Challenges," IEEE Internet of Things Journal, vol. 3, no. 5, pp. 637-646, 2016. 9

[23] Y. Zhang, Q. Yang, J. Chen, and D. N. Metaxas, "Semantic-Aware Multi-Modal Multi-Task Deep Learning for Traffic Scene Understanding," arXiv preprint arXiv:1409.3215, 2014. 9

[24] S. Ji, W. Xu, M. Yang, and K. Yu, "3D Convolutional Neural Networks for Human Action Recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 1, pp. 221-231, 2013. 9

[25] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, "Urban Computing: Concepts, Methodologies, and Applications," ACM Transactions on Intelligent Systems and Technology, vol. 5, no. 3, pp. 1-55, 2014.

10