Traffic Sign Recognition System

Abstract—**Recognition of traffic signs plays a crucial role in ensuring road safety. As the number of traffic signs increases, manual recognition becomes prone to errors. To overcome this challenge, deep learning-based methods have gained significant popularity in recent years. These methods have proven to be more efficient in detecting and recognising patterns in data than humans. This work focuses on developing a system to detect traffic signs in videos and prompt drivers when a sign is detected. It leverages different convolutional neural network architectures, as well as computer vision techniques, to interpret the traffic signs. This work consists of two parts: building the model using various techniques and implementing the most effective technique for real-life traffic sign detection using an event camera.**

Index Terms—**component, formatting, style, styling, insert**

Introduction

Transportation System Resilience (TSR), defined as the ability of a transportation network to withstand and recover from disruptive events, heavily relies on traffic sign awareness. This awareness is crucial for maintaining road safety and plays a pivotal role in the system's overall resilience. The increasing variety of traffic signals on the road has made manual identification far more error-prone. Growing interest in artificial intelligence (AI)-based approaches is a solution to this difficulty [1]. To construct a system designed to recognize traffic signs and promptly notify drivers upon detection, this study offers a focused examination of that process.

This study aims to overcome the limitations of manual recognition by leveraging advanced deep learning techniques, specifically through the integration of a Convolutional Neural Network (CNN) using ResNet architecture. The objective of this project is to enhance the legibility of traffic signs by employing cutting-edge computer vision methods.

Standing at the nexus of computer vision and artificial intelligence, this work aims to provide a concrete response to the problems posed by the increasing complexity of traffic sign recognition, while also making a valuable contribution to the ongoing discourse on intelligent transportation systems. This study intends to promote traffic sign-detecting systems and eventually contribute to the formation of a safer and more efficient vehicular environment by clarifying the intricacies of model development and practical deployment in real-world scenarios.

Related Works

In recent years, there has been a growing focus on advancements in self-driving cars, particularly in the area of traffic sign recognition. Researchers have been actively exploring methods to enhance the ability of autonomous vehicles to accurately detect and interpret traffic signs, paving the way for safer and more efficient transportation systems. One notable avenue of exploration is the utilization of deep learning methods, which have demonstrated promising outcomes by enabling systems to learn intricate features directly from raw data. A recent study presents a deep learning-based real-time traffic sign recognition system tailored for urban environments. Through extensive evaluation using a large-scale highway image dataset, the authors compare and assess deep learning-based object detection and tracking models, proposing a novel categorization method for urban road scenes while addressing potential obstacles in recognition tasks [2].

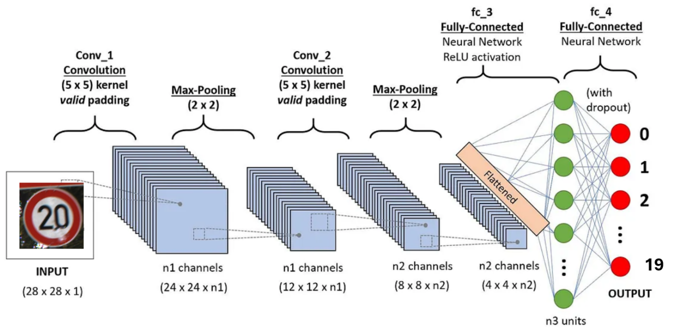
Additionally, the review by another study categorizes computer vision approaches for traffic sign detection into three types: color and shape-based, traditional machine learning-based, and deep learning-based, shedding light on their respective strengths and limitations [3]. Another investigation highlights the role of deep learning techniques in enhancing traffic sign recognition systems for driving assistance and road safety, emphasizing their significance in intelligent transportation [4]. Furthermore, another study provides a comprehensive overview of existing works in traffic sign interpretation, examining advancements and challenges from both methodological and dataset perspectives, thus contributing valuable insights to the field [5].

In addition to scientific considerations, several recent efforts emphasize effective communication with drivers by including user-centric design concepts and human factors [6]. These studies add to the overall improvement of traffic safety. Our understanding of traffic sign recognition is informed by the synthesis of this wide range of literature, and the identification of potential and gaps leads to the development of our novel Traffic Sign Recognition System (TSRS). Our investigation encompasses historical methods, the shift to machine learning, deep learning breakthroughs, a variety of datasets and sensors, real-time execution, and human aspects concerns, all of which contribute to enhancing the field of our research [7].

Design

The system illustrates the CNN architecture used for traffic sign recognition as shown in Fig (1). The process begins with an input image of a traffic sign, which undergoes two convolutional layers with kernels and valid padding, each followed by max-pooling layers to reduce dimensionality. The output from these layers is flattened and passed through fully connected neural network layers with ReLU activation functions and dropout for regularization. The final layer produces the classification output, identifying the traffic sign from a predefined set of classes.

The project illustrates the use of the DeepLab model for traffic sign recognition, leveraging image segmentation tech-



Conv_1 Convolution (5 x 5) kernel valid padding
Max-Pooling (2 x 2)
Conv_2 Convolution (5 x 5) kernel valid padding
Max-Pooling (2 x 2)
fc_3 Fully-Connected Neural Network ReLU activation
fc_4 Fully-Connected Neural Network
(with dropout)
Flattened
INPUT (28 x 28 x 1)
n1 channels (24 x 24 x n1)
n1 channels (12 x 12 x n1)
n2 channels (8 x 8 x n2)
n2 channels (4 x 4 x n2)
n3 units
0
1
2
19
OUTPUT

Fig. 1: CNN-based Traffic Sign Recognition System

niques. The process starts with an input image of a traffic sign, which is passed through the encoder part of the model as shown in Fig (2). The encoder employs atrous convolution and various convolutional layers with different dilation rates to capture multi-scale contextual information. The extracted features are then processed by the decoder, which utilizes upsampling and concatenation with low-level features to produce a high-resolution segmentation mask [8]. This output effectively highlights and segments the traffic sign from the background, facilitating accurate recognition.

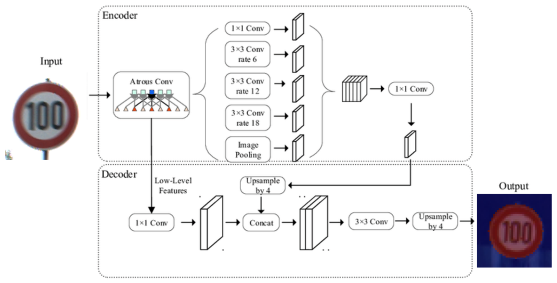


Fig. 2: DeepLab Model for Traffic Sign Recognition

Methodology

Convolutional Neural Network (CNN) architecture is employed, specifically adopting the ResNet framework. CNNs are a class of deep neural networks primarily designed for processing structured grid-like data, such as images [9]. The convolutional layers extract various features from input images through kernels, capturing hierarchical patterns. Pooling layers then downsample the extracted features to reduce dimensionality, enhancing computational efficiency. Finally, fully connected layers utilize these extracted features for classification or regression tasks. Within this framework, the ResNet architecture stands out for its unique approach to handling deeper networks by employing residual learning, enabling the training of significantly deeper architectures without encountering issues related to vanishing gradients [10].

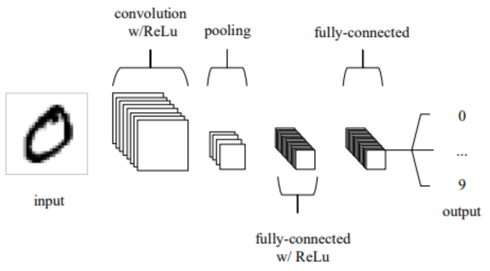


Fig. 3: Simple CNN architecture

A. ResNet architecture

Residual network (ResNet) introduces a novel approach to address the complexities of training deeper neural networks. To facilitate the training of networks with considerably greater depth than previous models, a residual learning methodology is employed. This unique approach reframes the layers within the network, focusing on learning residual functions concerning the inputs to the layers instead of learning unreferenced functions [11].ResNet architecture starts with initial convolutional layers for image preprocessing and further organizes the network into four primary layers (layer1 to layer4). Each of these layersconsists of multiple instances of the residual block, collectively enabling the model to capture and learn complex hierarchical features. These blocks employ residual connections, aiding in the efficient propagation of gradients and facilitating the training of deep networks. The architecture culminates with average pooling and a linear layer for classification, culminating in a model capable of effectively extracting features and making accurate predictions from input data(Fig. 4).

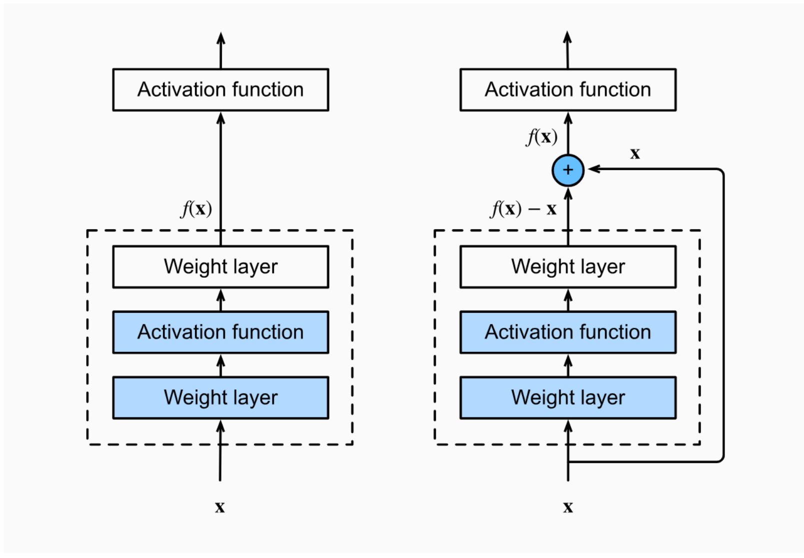


Fig. 4: ResNet architecture

1) Residual block: The residual block forms the fundamental unit within the ResNet architecture (Fig. 5). It encapsulates a basic building block for processing information within the neural network. This block structure comprises multiple convolutional layers integrated with batch normalization and rectified linear unit (ReLU) activation functions. Each block divides three consecutive convolutional layers, applying batch normalization and ReLU activation after each convolution. The block maintains a crucial residual connection, where the input is added to the output after the final convolutional layer. This mechanism facilitates the flow of gradients during training, allowing for the effective training of deeper networks without encountering vanishing gradient issues [11].

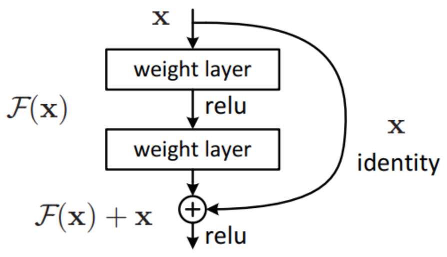


Fig. 5: Residual Block

B. MobileNet Architecture

MobileNet is a lightweight convolutional neural network (CNN) architecture designed for efficient inference on resource-constrained devices [12]. MobileNet employs depthwise separable convolutions, which split the standard convolution into two separate layers: depthwise convolution (which applies a single filter per input channel) and pointwise convolution (which combines the outputs of depthwise convolutions using 1x1 filters).This separation significantly reduces the number of parameters and computations while maintaining reasonable accuracy [13].

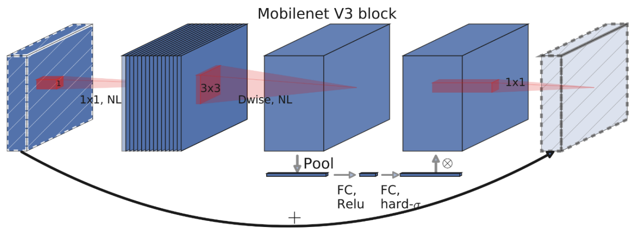


Fig. 6: MobileNet’s architecture

1) *Depthwise convolution:*

Depthwise convolution is a crucial element in many neural network architectures, especially for edge devices and segmentation applications [14]. For edge devices, Depthwise Separable CNN (DSCNN) is favored due to its efficient use of processing elements and excellent power efficiency .Additionally, the Multi-Scale Depthwise Separable Convolution module has been introduced for real-time semantic segmentation, focusing on extracting multi-scale spatial features to enhance the non-linear relationship between input and output. This improvement leads to better performance metrics, such as Mean Intersection over Union (MIoU), and increased parameter efficiency [15].

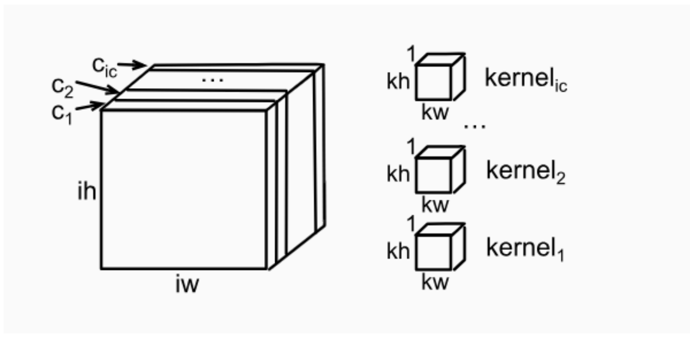


Fig. 7: Illustration of a depthwise convolution

2) *Pointwise convolution:*

Pointwise convolution is essential for optimizing lightweight convolutional neural network (CNN) models by effectively managing parameters and computational resources. Various studies have proposed innovative methods to enhance pointwise convolutions. For instance, the Ghost-PE and Ghost-PC blocks focus on optimizing channel-expanded and compressed convolutions [16]. Additionally, techniques like interleaved grouped filters and grouped pointwise convolutions have been developed to reduce complexity in deep CNNs while preserving their learning capacity [17]. Furthermore, the application of traditional transforms, such as the Discrete Walsh-Hadamard Transform (DWHT), to pointwise convolutions has led to substantial reductions in computational complexity without compromising accuracy, thus making them effective feature extractors in neural networks [18]. Moreover, a novel approach combining depthwise separable convolutions with frequency-domain convolutions using the Discrete Cosine Transform (DCT) has been proposed, which selectively prunes frequencies to cut down computation time without significantly affecting accuracy [19].

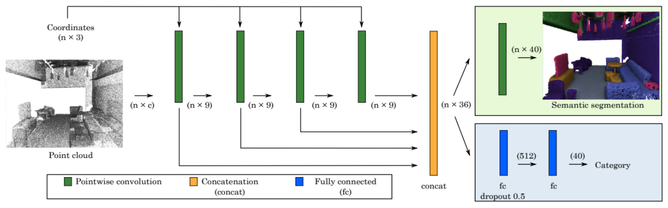


Fig. 8: Pointwise Convolution Neural Network

C. DeepLab Model and Image Segmentation

The DeepLab model is a state-of-the-art deep learning framework for semantic image segmentation, where the goal is to assign semantic labels to every pixel in an image. It stands out due to its use of atrous convolution, which allows it to capture multi-scale information and effectively segment objects at multiple scales [20]. This capability is particularly significant in the context of traffic sign recognition, where accurate and rapid identification of signs is crucial for the safety and efficiency of autonomous driving systems. Image segmentation serves as the backbone for this task, enabling the system to isolate and recognize traffic signs from complex backgrounds (as shown in Figure 2) , under various environmental conditions [21]. Moreover, by segmenting images into semantically meaningful regions, DeepLab facilitates robust feature extraction and improves the model's ability to discern fine-grained details, leading to enhanced accuracy and generalization performance in traffic sign recognition tasks. By precisely segmenting traffic signs, DeepLab aids in reducing computational effort and improving recognition speed, which is essential for real-time navigation and prompt decision-making in intelligent transportation systems [22].

1) DeepLab architecture: DeepLabV3 is a convolutional neural network architecture designed for semantic segmentation tasks. It builds upon the success of its predecessors by introducing several key improvements to enhance segmentation accuracy and efficiency. One of the distinctive features of DeepLabV3 is the use of atrous convolutions, also known as dilated convolutions, which enable the network to capture multi-scale contextual information efficiently as shown in (Fig. 9). By varying the dilation rates of these convolutions, DeepLabV3 can effectively expand the receptive field of each convolutional layer, allowing it to capture both local details and global context [23]. Additionally, DeepLabV3 incorporates a powerful form of spatial pyramid pooling (ASPP), which further enhances its ability to capture contextual information at multiple scales. These architectural innovations enable DeepLabV3 to achieve state-of-the-art performance in semantic segmentation tasks across a wide range of domains.

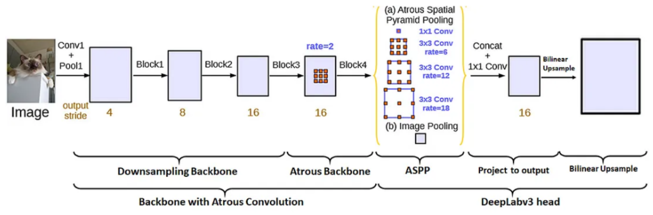


Fig. 9: DeepLabv3 Architecture

2) Preprocessing: Before being fed into the DeepLab model, the input images underwent several preprocessing steps to ensure compatibility and optimal performance. Initially, segmentation masks were created based on the x and y coordinates of each shape within the images. This function automates the process by drawing the specified shapes on a blank mask. This process facilitated the generation of ground truth labels for training the DeepLab model in a supervised manner. The PyTorch framework was employed to implement this preprocessing pipeline, leveraging its flexibility and efficiency for deep learning tasks. Additionally, to standardize the input data and enhance model convergence, the images were resized to a uniform dimension compatible with the input requirements of the DeepLab architecture. The dataset was split into training and testing sets using a predefined ratio of 0.8 (train) to 0.2 (test), ensuring an adequate amount of data for both training and evaluation. By integrating these preprocessing techniques into the pipeline, the input data were effectively prepared to be fed into the DeepLab model, thereby facilitating accurate and reliable image segmentation for traffic sign recognition.

3) Transfer Learning: Transfer learning was a fundamental component of our approach, where pretrained DeepLabV3 models were employed such as ResNet-50 and ResNet-101. These ResNet backbones had been previously trained on large-scale datasets for image classification tasks, enabling them to learn rich hierarchical features that are transferable to the traffic sign recognition task. By initializing the DeepLabV3 models with weights from the pretrained ResNet backbones, the generalization capabilities of these backbone networks were capitalized, which had already learned to extract meaningful visual features from diverse images. This facilitated faster convergence during training and enabled the models to achieve competitive performance [24]. Fine-tuning of the pretrained DeepLabV3 models on specific tasks further refined their segmentation capabilities, resulting in improved accuracy and robustness.

Experimental Setup

In this project, PyTorch framework was used to implement traffic sign recognition models. Several model architectures were experimented, including ResNet-50, ResNet-101, and MobileNet. For loss functions Mean Squared Error (MSE), and Cross-Entropy Loss, were utilized. In the ResNet from scratch experiments, Adam optimizer with a learning rate of 0.001 and a weight decay of 0.0001, as well as Stochastic Gradient Descent (SGD) with a learning rate of 0.01 and momentum of 0.9 were applied. For the DeepLab model, Adam optimizer with learning rate of 1e-3 was used. The batch sizes were set to 64 for both ResNet from scratch and DeepLab models. Training was conducted for 20 epochs for ResNet from scratch and 15 epochs for the DeepLab model, with an 80-20 train-validation split. Evaluation metrics included training, validation, and test accuracies, F1 score, and balanced accuracy.

D. Dataset

The dataset utilized in this study stems from the German Traffic Sign Benchmark, commonly referred to as GTSRB. Introduced as a multi-class, single-image classification challenge during the International Joint Conference on Neural Networks (IJCNN) in 2011, the GTSRB dataset stands as a benchmark for traffic sign recognition tasks [25]. The GTSRB dataset is organized into multiple folders, each representing a specific class of traffic signs, such as maximum speed limits, stop sign and yield sign, providing a comprehensive collection for traffic sign recognition.

However, due to the substantial volume of images found within the dataset, this project focuses solely on the first 20 classes. This selective approach streamlines the dataset for proof-of-concept purposes, allowing for efficient processing and model development. By limiting the scope to these initial classes, we maintain a manageable dataset size while demonstrating the effectiveness of our classification approach. This approach ensures that, despite the reduction in classes, our pipeline can effectively showcase the feasibility and efficacy of the proposed methodologies within a limited yet representative subset of the GTSRB dataset, taking into consideration that the dataset is not balanced (Tab. I).

Class	Number of images	Shape
0	210	Circle
1	2220	
2	2250	
3	1410	
4	1980	
5	1860	
6	420	
7	1440	
8	1410	
9	1470	
10	2010	
11	1320	Triangle
12	2100	Diamond
13	2160	Inverse-triangle
14	780	Octagon
15	630	Circle
16	420	
17	1110	
18	1200	Triangle
19	210	

TABLE I: Class Distribution

E. Preprocessing

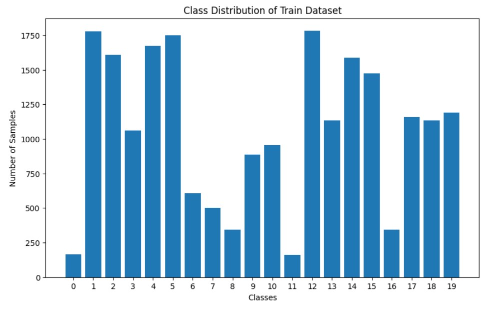
1) Transformations: The preprocessing pipeline begins with a sequence of transformations applied to the input images, such as resizing and normalization, to ensure compatibility with the architectures used in this work. Several other transformations, such as random rotations and flips, are also employed in order to ensure the robustness of these networks, thus promoting effective generalization to unseen data.

2) Dataset Split: The labelled images in the dataset undergo a partitioning process to create *disjoint* train and validation sets using a $80 - 20$ ratio. As for the test set, it is composed of the unlabeled images of the dataset.

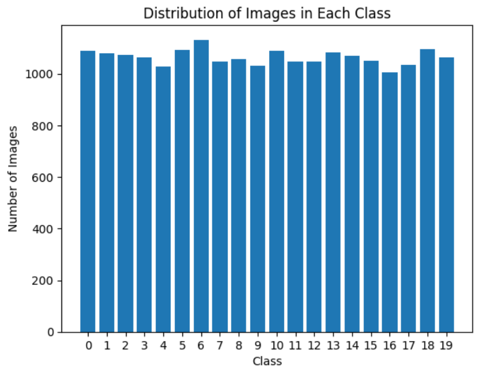
As previously discussed (Tab. I), the classes present an imbalance that should not be overlooked when training our networks, as they may develop biases towards the majority classes, thus generalizing badly to unseen data. The literature suggests various techniques to address this issue, ranging from simple approaches like undersampling or oversampling to more sophisticated methods such as SMOTE [26]. In this work, we use a weighting mechanism to oversample minority classes. The weight of a class $\mathcal{C} \in C$ is given by

$$w_{\mathcal{C}} = \frac{1}{\#\mathcal{C}} \tag{1}$$

where $\#\mathcal{C}$ represents the number of instances in class $\mathcal{C}$. The reasoning behind this is to assign higher weights to minority classes, making them more likely to be sampled during the training process. The figure below (Fig. 10) shows the imbalance and distribution of the classes after applying weighted oversampling.



(a) Before



(b) After

Fig. 10: Class distribution, before and after oversampling

Experiments & Results

In this section, we compare the performance of different models trained using distinct optimization algorithms and hyperparameters.

F. Resnet from scratch

The table below summarizes the accuracy metrics achieved by the 50-layer convolutional neural network (ResNet50).The weighted sampling method was applied solely on the train set, it is important to note that this approach was not utilized on the validation and test sets. For training, the model was trained over 20 epochs from scratch, using different optimization algorithms, with cross-entropy as the loss function given by

$$H(y, \hat{y}) = -\sum_{i=1}^{\#C} y_i \cdot \log(\hat{y}_i) \tag{2}$$

where y and $\hat{y}$ are the true and predicted probabilities of the i-th class.

The table below (Table II) illustrates the accuracy results obtained after training the ResNet50 models using different optimization algorithms and hyperparameters. These results demonstrate the influence of optimization techniques on model performance and convergence.

Model	Opt. Alg.	LR	Momentum	Batch Size	Accuracy	Balanced Acc
ResNet50	Adam	0.001	-	64	0.927	0.929
ResNet50	SGD	0.01	0.9	64	0.916	0.931

TABLE II: ResNet from Scratch Model Comparison and Accuracy

G. DeepLab Model

In this study, we evaluated the performance of the DeepLabv3 model using different architectures, specifically ResNet-50, ResNet-101, and MobileNet. The models were trained using the Adam optimizer with a learning rate of 1e-3 over 15 epochs, with a train-test split ratio of 0.8 to 0.2, using a batch size of 64. The loss functions used was MSE (mean squared error).

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \tag{3}$$

where N is the number of data points, y and $\hat{y}$ are observed and predicted values of the i-th class.

The results table below presents the performance metrics of these models, focusing on train loss, test loss, train AUROC (Area Under the Receiver Operating Characteristic curve), and test AUROC [27] as shows in (Table (III). The results provide a comprehensive comparison of the models' capabilities and generalization to unseen data.

	Train loss	Test loss	Train F1 Score	Test F1 Score	Train AUROC	Test AUROC
ResNet50	0.050	0.063	0.827	0.820	0.940	0.930
ResNet101	0.050	0.040	0.827	0.802	0.940	0.930
MobileNet	0.090	0.111	0.731	0.729	0.918	0.908

TABLE III: DeebLabv3 Models Comparison and Accuracy

Discussion

Notably, ResNet50 and ResNet101 architectures achieved the highest accuracy among the models tested. Both models utilized the Mean Squared Error (MSE) loss function, which proved to be effective in this context. The superior performance of ResNet50 and ResNet101 can be attributed to their deep architectures, which enable the capture of intricate features and patterns within the traffic sign images. This capability is crucial for distinguishing between a diverse set of traffic signs. The high accuracy achieved by these models demonstrates their robustness and effectiveness in real-world applications, highlighting their potential for deployment in autonomous driving systems and advanced driver-assistance systems.

Moreover, incorporating DeepLab for background removal further enhanced the performance and accuracy of the models. By isolating the traffic signs from their backgrounds, the models were able to focus on the relevant features without interference from extraneous elements. This preprocessing step significantly improved the model's ability to correctly classify the traffic signs, leading to higher accuracy and better overall performance. The integration of DeepLab for background removal underscores the importance of preprocessing in enhancing model effectiveness. This improvement illustrates the potential for advanced image processing techniques to enhance the performance of deep learning models in practical applications.

Conclusion

To sum up, this study on traffic sign recognition smoothly combines methodological, procedural, and architectural aspects. Utilizing a Convolutional Neural Network (CNN) in conjunction with the ResNet architecture is a purposeful and advanced method of tackling the difficulties involved in traffic sign identification.

The dataset, which was carefully selected from the German Traffic Sign Benchmark (GTSRB), is limited to the first twenty classes. This painstaking curation intentionally aims to strike a compromise between computing efficiency and a comprehensive examination, much like the controlled settings of experimental design.

Preprocessing shows a careful dedication to correcting intrinsic class imbalances and strengthening dataset resilience using a variety of transformations and an accurate class balancing technique. The ResNet architecture, which incorporates residual learning principles and is well-known for its ability to train deep neural networks and handle vanishing gradient problems, is the central component of this work.

Hierarchical feature extraction is made easier by the residual block, the core component of the ResNet architecture. Integrated smoothly into the ResNet class, this complex network design gracefully captures complex representations, resulting in a neural network ready for accurate prediction results.

This study is in line with more general efforts to further the conversation around intelligent transportation systems, which go beyond its technological roots. This work greatly advances the overall goal of improving vehicle safety in modern traffic situations by increasing the effectiveness of traffic sign recognition systems. When taken as a whole, the technological innovations showcased make a significant addition to the continuing story of transportation system optimization for the benefit of public safety and welfare.

Future Work

Several key areas can be explored to enhance the traffic sign recognition system. Optimizing the model for efficiency through pruning, quantization, and knowledge distillation will make it more suitable for deployment on edge devices. Advanced data augmentation techniques and synthetic data generation can improve the robustness of the model against real-world variability. Integrating the system with multi-sensor data and developing real-time processing capabilities will enhance its applicability in autonomous driving scenarios. Expanding the model's capabilities through multi-task learning and hierarchical classification can provide a more comprehensive understanding of traffic environments. Additionally, improving model uncertainty estimation will make the system's decision-making process more transparent and reliable. Moreover, incorporating techniques like adversarial training and domain adaptation can improve recognition accuracy in poor visibility conditions and ensure the system works efficiently even in adverse conditions such as blurriness caused by bad weather.

Acknowledgment

I thank Faisal Jayousi and Özgür Özdemir for assistance with the methodology and for comments that greatly improved the manuscript.

References

[1] P. Y. Alexander Shustanova, "Cnn design for real-time traffic sign recognition," 2017.

[2] "Deep learning-based real-time traffic sign recognition system for urban environments," 2023.

[3] "Two-stage traffic sign detection and recognition based on svm and convolutional neural networks," 2020.

[4] "Enhancing traffic sign recognition (tsr) by classifying deep learning models to promote road safety," 2023.

[5] "Traffic sign interpretation in real road scene," 2023.

[6] I. van Schagen & Luuk Vissers, "Human factors in traffic management," 2016.

[7] D. P. Radu Dobrescu, "Real time primary image processing," 2011.

[8] "Image semantic segmentation algorithm based on a multi-expert system," 2023.

[9] R. N. Keiron O'Shea, "An introduction to convolutional neural networks," in *Advances in neural information processing systems*, 2015.

[10] "Traffic sign detection and recognition based on convolutional neural network," 2023.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[12] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017.

[13] "Mobile-hr: An ophthalmologic-based classification system for diagnosis of hypertensive retinopathy using optimized mobilenet architecture," 2023.

[14] "A-dscnn: Depthwise separable convolutional neural network inference chip design using an approximate multiplier," 2023.

[15] "Depthwise convolution based pyramid resnet model for accurate detection of covid-19 from chest x-ray images," 2023.

[16] "Optimized pointwise convolution operation by ghost blocks," 2023.

[17] "An enhanced scheme for reducing the complexity of pointwise convolutions in cnns for image classification based on interleaved grouped filters without divisibility constraints," 2022.

[18] "Grouped pointwise convolutions reduce parameters in convolutional neural networks," 2022.

[19] "A new pointwise convolution in deep neural networks through extremely fast and non parametric transforms," 2022.

[20] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," 2015.

[21] A. Krishnamurthy, "Traffic symbol recognition by means of image segmentation," 2014.

[22] G. S. Juanhong Xie and W. Zhu, "Intelligent recognition technology for the segmentation of traffic indication images concerning different pavement materials," 2022.

[23] F. S. Liang-Chieh Chen, George Papandreou and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017.

[24] "Semantic segmentation with extended deeplabv3 architecture," 2019.

[25] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark," in *International Joint Conference on Neural Networks*, no. 1288, 2013.

[26] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, p. 321–357, June 2002.

[27] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," 1997.