MDPI

Article

Traffic Sign Detection Using Lightweight Deep Learning Model

Abrar Ahbabul Haque , Sakib Ul Haque, Nibir Neelim and Adib Reza

Abstract: In this time of autonomous driving systems, accurate traffic light detection is essential for ensuring safety. This paper will investigate the performance of some of the leading lightweight deep learning architectures namely MobileNet V3 and MobileNet V2 and compare them to more complex models, all of which have been customized for traffic light recognition contributing to the enhancement of autonomous driving systems. The aim of this research is to analyze the trade-offs between accuracy and computational complexity of these models to make it deployable on platforms that are resource constrained. These models were trained on a dataset containing a diverse range of traffic sign images under different lighting conditions. The models were fine tuned to optimize their performance in traffic sign detection while keeping the processing time to minimal. Our findings reveal that MobileNet V3 has outstanding performance despite its simple architecture. This analysis serves as a benchmark and highlights the potential for using these lightweight models in practical applications ensuring both performance and efficiency.

Keywords: autonomous driving systems, traffic light detection, lightweight deep learning architectures, mobileNet V3, mobileNet V2, complex models, customization, trade-offs, accuracy, computational complexity, resource-constrained platforms, fine-tuning, performance optimization, processing time, benchmark, efficiency

1. Introduction

The ever expanding field of autonomous navigation systems constantly demands increasing safety levels for smart systems that are employed to detect traffic signs as the safety of people's lives may depend on it. Here, this paper tries to bring attention to two leading lightweight deep learning models, MobileNet V3 and MobileNet V2, which have been adapted specifically to recognize traffic signs. There is a very big gap in literature when it comes to finding a balance between accuracy and processing demands in resource-constrained environments of self-driving vehicles. Our research addresses this by comparing the two models under varying real world conditions with models that have high computational expenses, namely EfficientNet-B7 and ResNet-152. Our aim is to learn exactly how different models perform in terms of accuracy and computational complexity and discover insights into improvements for autonomous navigation systems in the real world. This paper presents detailed experimental results and analyses of lightweight models in real life scenarios and use-cases, thus shedding light on their potential and their limitations.

2. Background

In today's societies, electric vehicles have taken the markets by storm, creating interest in all sorts of computerized autonomous driving systems. To accomplish this however, many sorts of techniques have been proposed and used over the years, all to identify traffic lights and their states in different lighting conditions and distances. Firstly, for the longest time, researchers have been trying out computer vision based approaches to solving this massive problem. Haltakov et al. used such a method by semantically separating day and night TLs but it had low robustness [1]. Du et al. used a system based on prior knowledge

Citation: Haque, A.A.; Haque, S.U.; Neelim, N.; Reza, A. Title. *Journal Not Specified* **2024**, *1*, 0.

Received:
Revised:
Accepted:
Published:

of the TLs, but it had the problem of feasibility, scalability and costs needed for improving its capabilities [2].

Then researchers looked into machine learning based methods to solve the problem. Vitas et al. brought forth a machine learning approach with dataset image augmentation [3] while Weber et al. presented his hierarchical traffic light detection algorithm [4]. Then Saini et al. presented a computer-vision based TL structure detection algorithm that worked in tandem with the convolutional neural network (CNN)-based state information extraction system [5]. Then Vitas et al. brought forth a method that used adaptive thresholding and deep learning for the tasks of region proposal and traffic light localization respectively [6]. Moreover, Kulkarni et al. used a deep learning approach using transfer learning to determine the traffic light recognition and detection in Indian traffic scenarios [7]. Furthermore, Wang et al. gives a novel LL4PH - Net framework and enhances images using dark channel prior knowledge to propose an accurate method for traffic objects in low light. It outperforms existing methods and has potential applications in traffic monitoring and autonomous driving [8]. Thus, researchers were leaning into using a combination of different techniques instead of relying on any one of them for better accuracy in real world scenarios.

3. Method

3.1. Dataset

The dataset used in this paper consists of preprocessed traffic sign data in pickle format. The training dataset was balanced in all of its 43 classes of traffic signs through augmentation that transformed the original images with the adjustment of brightness and rotation. This would enable the model to be generalized across diverse traffic signs with training samples of up to 86,989. German Traffic Sign Recognition Benchmark (GTSRB) was the source of the initial data of this dataset. Each image in the dataset is a 32x32 pixel with 3 color channels (RGB). The preprocessing enabled the images in the dataset to be available for robust training and testing.

3.2. Preprocessing

The dataset was already preprocessed by the author. The original images were augmented to equalize the number of samples across 43 classes (2023 images per class), using rotations and changes in brightness. For each of the red, green and blue channels, the pixel values of all the images ranging from 0 to 255 were rescaled in the range between 0 and 1 to speed up the learning process. The order of the images are shuffled so that the model does not learn any order-dependent patterns from the training data. The following show some sample images from the training set:

Figure 1. Sample Images From Dataset

It must be noted that the number of training iterations are not kept the same for all models because some of them do not show improvements after a certain number of epochs. They also take a long time to train even with a dedicated gpu, so reducing training iterations also saves time.

3.3. MobileNet V2

The MobileNetV2 model, which is known for its effective performance on mobile devices because of its lightweight architecture, is used for traffic sign classification. To capture a wide range of features, the model starts with pre-trained ImageNet weights. To customize MobileNetV2 for the task of identifying 43 traffic sign classes, custom layers are added on top of the base network architecture (Dense, Activation[RELU], Batch Normalization, Dropout [with a dropout rate of 0.3], and lastly Dense and Activation [Softmax] as prediction layer). The dropout layer and a dense layer with 256 neurons are combined to reduce overfitting and capture the intricate relationships found in the data.

To stabilize the learning process, custom layers use batch normalization layers. The training of this model is split into two phases: the first phase keeps the base model layers frozen and trains only the custom layers and runs for 10 epochs. The second phase trains all of the layers except batch normalization layers to help maintain stability and prevent overfitting for fine-tuning.

The model is compiled using the Adam optimizer, which has a much lower learning rate and carefully fine-tunes the model weights. Callbacks like EarlyStopping, which stops training if the validation loss does not improve after a predetermined number of epochs, and ReduceLROnPlateau, which modifies the learning rate if the validation loss plateaus are used in the training process to prevent overfitting. Very similar approaches are used for MobileNet V3 and ResNet-152, as described in the following subsections.

3.4. MobileNet V3

MobileNet v3 Large model architecture is used that is highly optimized for its performance in mobile devices for its reduced size and complexity. The model is initialized with pretrained weights from the Imagenet dataset. Several custom layers were added to the base layers- GlobalAveragePooling2d, Dense, Batch Normalization and another Dense layer for predictions. The model is initially trained for 10 epochs to improve training accuracies and then for the next 10 epochs, the batch normalization layers are frozen as a fine-tuning technique.

Adam optimizer is used in this model that maintains distinct learning rates based on the weights updated. Categorical cross entropy is used as the loss function as the model has to deal with categorical classification tasks of 43 classes where the labels are one-hot encoded. This loss function measures the difference between the predictions and true labels which will guide the network to make more accurate predictions. The learning rate is set at 0.001 for epochs less than 5 and it is decreased to 0.0005 for epochs between 5 and 10. This adjustment of learning rates enables the model to converge faster in the initial phases with higher learning rate and fine-tune better in the later stages with lower learning rate.

3.5. ResNet-152

Resnet-152 is a deep convoluted neural network used in the classification of images. In this model, we used the weights pre-trained on the ImageNet-1K dataset version 2 in which the NVIDIA GeForce GTX 1070 is used as the GPU for faster deep learning tasks. Each channel of the image is normalized by using the same mean and standard deviation values as pre-trained on ImageNet images to adjust the input data having zero mean and unit variance. Data loaders are used in training, validation and test data for efficient and faster loading of data for feeding into the neural network. The use of multiple workers enables the data loader to parallelize data loading which results in much faster processing time during training. However, it also increases memory so the number of workers is set to 4 for optimal solution.

The ResNet-152 model is customized by having additional fully connected layers (equivalent to Dense layer from tensorflow) with 512 output features and a rectified linear unit (ReLU) activation function. The final output layer has 512 input features and 43 output features where log softmax activation is used to provide log probabilities for the classification. Batch normalization is applied to accelerate the convergence. After all these modifications, the model is moved to the appropriate device. The cross entropy loss function and Adam optimizer is used with a learning rate of 0.001 to update the parameters of the fully connected layers. Finally, the model was trained for 20 epochs. The training part was divided into two phases. In the first phase, only the custom final classification layers are trained from epochs 1 to 10 where the parameters of the base model are frozen to prevent them from being updated during backpropagation. In the second phase after epoch 10, the layers of the base model are unfreezed except the batch normalization layers. It should be noted that we tried training the model again with an even lower learning rate of 0.00005 after it showed poor performance on the test and validation sets.

3.6. EfficientNet-B7

The EfficientNet-B7 model is chosen because it scales well in depth, width, and resolution, improving accuracy and efficiency. It is well-known for its scalable performance across a variety of devices. By using pre-trained weights from the ImageNet dataset, it improves traffic sign recognition significantly. We did not add any custom layer to the model, but we did apply a very low learning rate of 0.00005. It should be noted that this model is the largest among all the ones we have chosen and is thus computationally most expensive.

By normalizing layer outputs, cutting down on training epochs, and preserving efficient learning dynamics, batch normalization is used to stabilize training. The model uses the categorical cross-entropy loss function, which is appropriate for multi-class tasks, and the Adam optimizer because of its effectiveness with large datasets.

4. Results

4.1. MobileNet V2

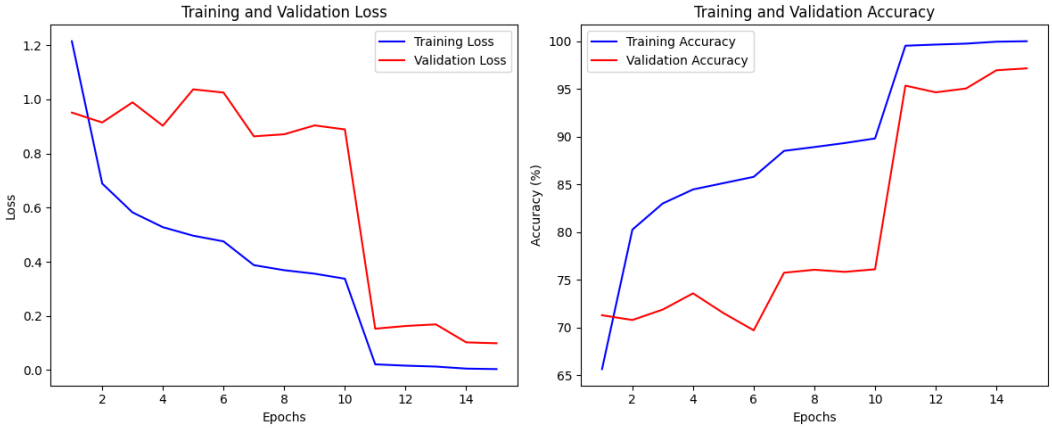

Figure 2. Training and Validation Loss (Left) and Training and Validation Accuracy(Right) graphs for MobileNet V2

After modifications, the MobileNetV2 model showed significant improvements. The training loss dropped dramatically from a starting 0.2536 to a very low 0.0306, indicating that the model's predictive accuracy had improved. Simultaneously, the training accuracy increased significantly to 99.22%. The validation results followed this upward trajectory, with the accuracy rising to 97.14% and the loss decreasing to 0.1687 by the end of the fifth epoch of the second phase. These numerical metrics highlight the model's ability to

correctly classify traffic signs and its efficient adaptation to the task's specifics, especially the high accuracy and low loss on both training and validation sets.

4.2. MobileNet V3

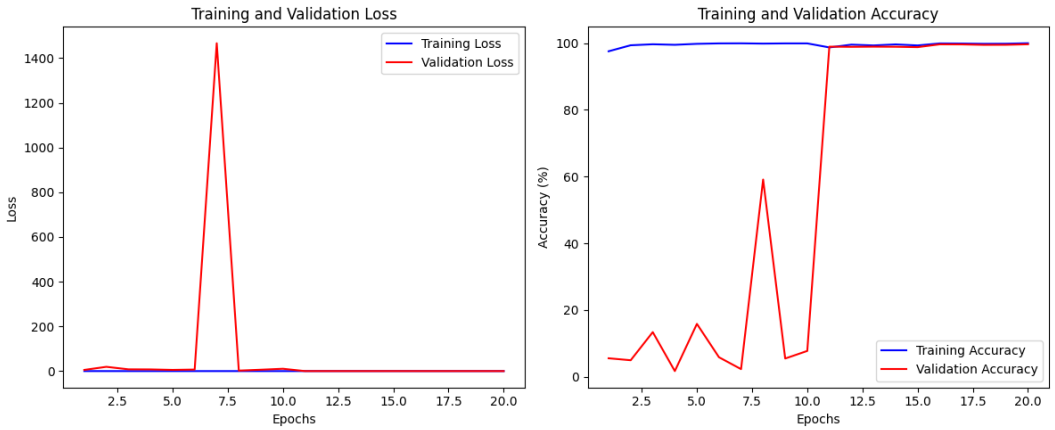

Figure 3. Loss and Accuracy graphs for MobileNet V3

For the first 10 iterations, we can see that the model reaches very high training accuracies, but very low validation accuracies. The validation curves are quite erratic for this phase. It may not be visible in the loss graph, but validation loss for the first phase was very high and reached over 1400 in the 7th iteration (which is why high losses are not understandable from the graph). But as soon as the second phase starts, where the batch normalization layers are frozen, the validation accuracy skyrockets and reaches very close to the training accuracy in the 11th epoch and plateaus.

4.3. EfficientNet-B7

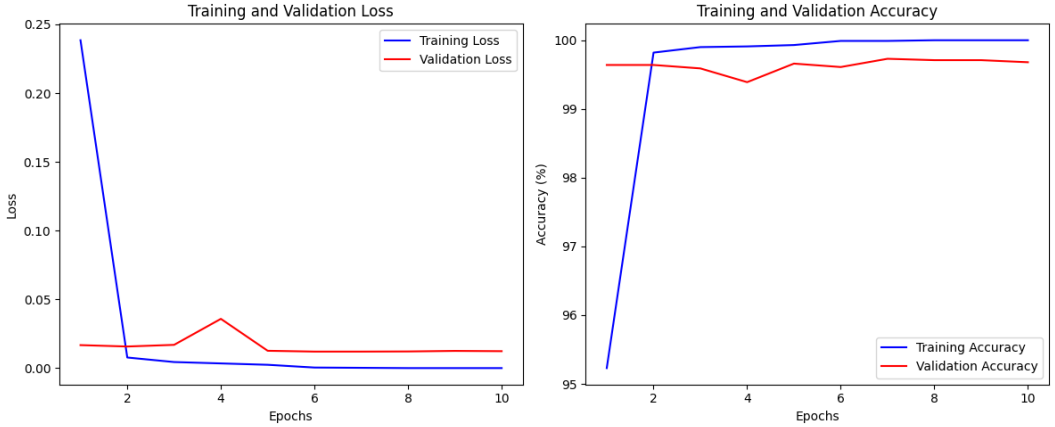

Figure 4. Loss and Accuracy graphs for EfficientNet B7

In the traffic sign classification task, the EfficientNet-B7 model performed perfectly on the training set, meaning it achieved an accuracy of 100%. This may indicate overfitting, but the model's validation metrics were very close, with an accuracy of about 99.7% and attaining a test accuracy of 99.06%. A thorough analysis of the model's performance using the classification report and confusion matrix showed that nearly all traffic sign classes had high metric scores. For most classes, the model's precision, recall, and F1 scores were all very close to 1, showing how good it was at identifying different traffic signs. Since this is a large model and achieved the highest performance, we can use this as a benchmark for the other lightweight models.

4.4. ResNet-152

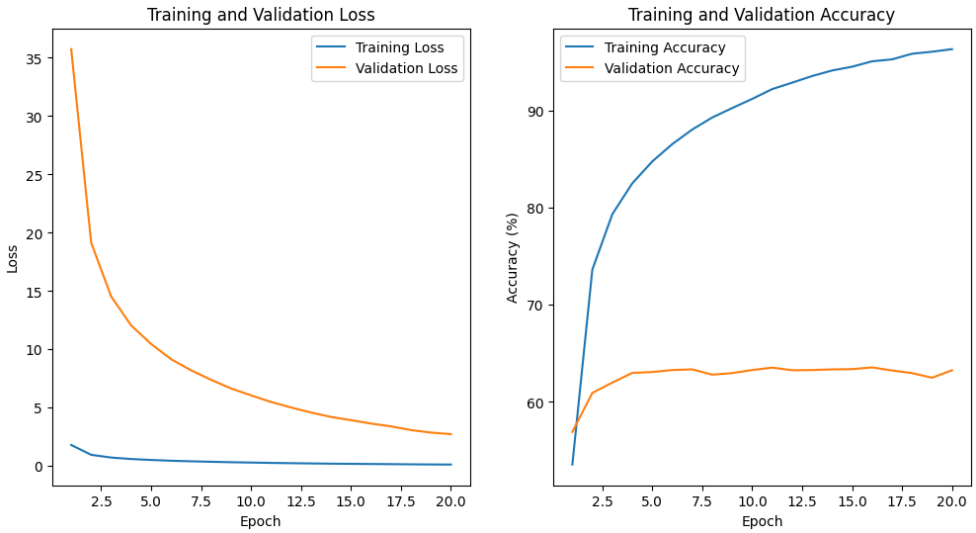

Figure 5. Loss and Accuracy graphs for ResNet-152

We initially trained this model with a relatively high learning rate of 0.001, but the test accuracy was around 61% while the training accuracy was over 95%, which meant overfitting. So, we trained the model again but this time with a learning rate of 0.00005. However, it still resulted in little to no change. The graphs above are for the updated learning rate. The model reached a training accuracy of 96.36% after 20 epochs and a validation accuracy of 63.2

4.5. Model Evaluation

After training, the models were evaluated with testing data to produce test accuracies, confusion matrices and performance metrics. We have produced confusion matrices for all the models, but they were not labeled with text. We had attempted to build an improved MobileNet V3 Large model and there we used all the proper text labels. It achieved a slightly lower test accuracy of 96.3%. Below is the confusion matrix of that model with proper text labels.

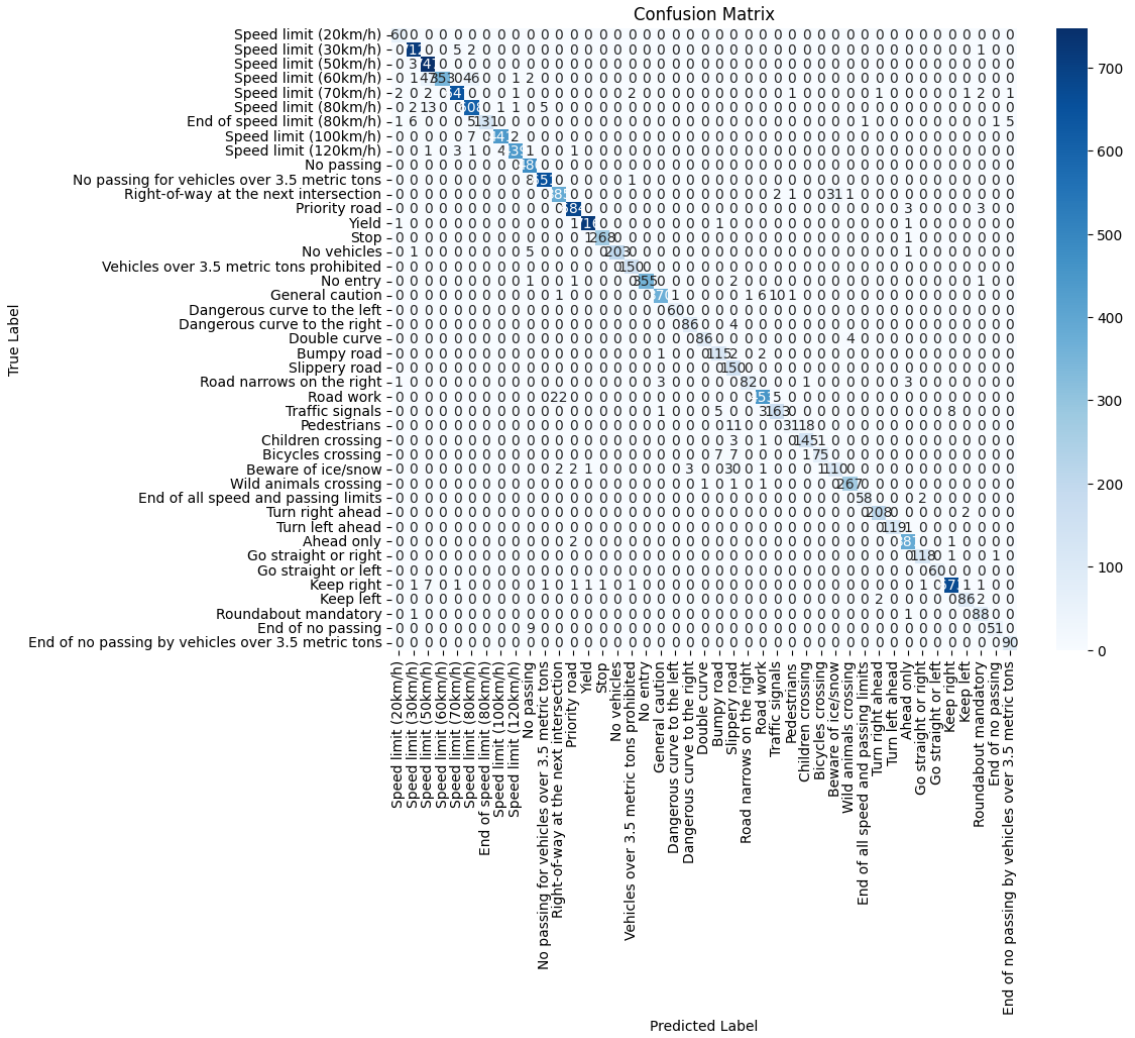

Figure 6. Confusion Matrix of MobileNet V3 (Demonstration Version)

Table 1. Performance metrics of the Models

Model	Training Accuracy (%)	Validation Accuracy (%)	Test Accuracy (%)	Macro-average Precision	Macro-average Recall	Macro-average F1 Score
EfficientNet B7	100	99.68	99.06	0.99	0.98	0.98
ResNet152	96.36	63.27	63.66	0.55	0.54	0.54
MobileNet V2	99.98	97.14	95.22	0.93	0.92	0.92
MobileNet V3 Large	99.99	99.68	98.53	0.97	0.97	0.97

One of our main goals was to compare the results between lightweight and heavyweight models. Below is a table showing the parameter sizes of the models in descending order. We could not compare training times because different devices were used for training these models. MobileNet V2 and V3 Large were trained using a cpu (AMD Ryzen 5 3600) and ResNet-152 and EfficientNet-B7 were trained on a gpu (NVIDIA GTX 1070).

Table 2. Model Parameter Sizes

Model	Parameter Size (MB)
EfficientNet B7	243.33
ResNet152	223
MobileNet V3 Large	13.4
MobileNet V2	9.91

5. AI EXPLAINABILITY

Even though we use the testing set to see a model's performance on unseen data and look for closeness in accuracy with training performance, we still do not know what features the models are observing in order to make predictions. Essentially, these models are black boxes. To be able to trust our trained models, we need to be able to see if they are focusing on the salient features of the data. Explainable AI makes the models interpretable for humans. For our project we will use LIME or Local Interpretable Model-agnostic Explanations. It should be noted that LIME shows approximations, not the exact selected feature regions. The following figures show the explanations of the best 3 models for six images picked from different classes of the test set.

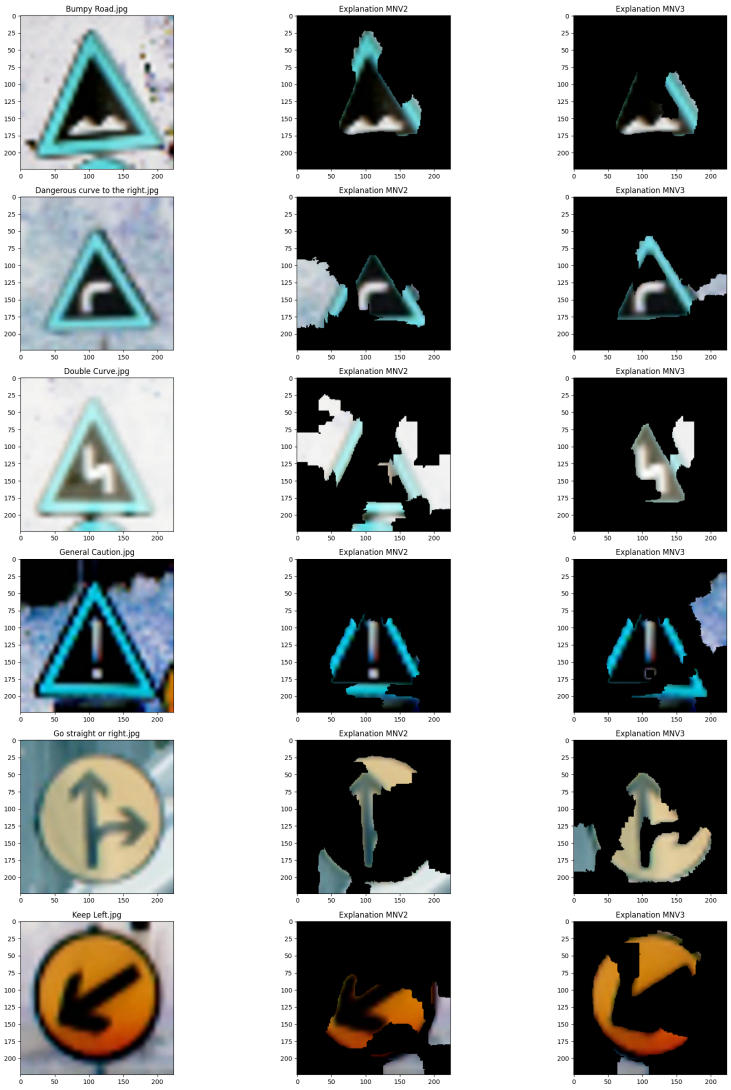

Figure 7. Explanations for MobileNet V2(Middle Column) and MobileNet V3 Large (Right Column)

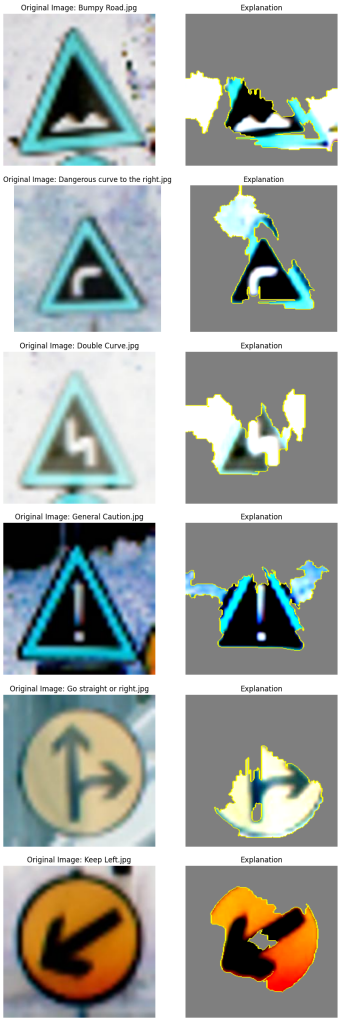

Figure 8. Explanations for EfficientNet-B7

We can see from the explanations that all three models are able to observe most of the salient features in order to make predictions. For most of the explanations we can see that the visible regions (i.e. the regions that the models are "looking at") are bounded within the circular and triangular plates of the traffic signs. As for the symbols themselves, the models are mostly able to capture them as the salient features.

6. Discussion

The MobileNetv3 model showed exceptional training behavior with good generalization on the validation set and high test accuracy which makes it ready for deployment on real-world data. Its performance should be monitored to ensure it maintains high accuracy outside of the test environment. The model should be tested on more challenging scenarios that are outside of the test set which would enable us to understand the limits of the model's capabilities. Besides, early stopping can be applied to prevent unnecessary training if the model consistently performs well on the validation data before the last epoch.

The results of fine-tuning the MobileNetV2 model show that it is a very successful adaptation for the traffic sign classification task. Significant accuracy gains were obtained by the fine-tuning strategy, indicating that the model was able to learn and generalize well from the available data. The model was not overfitting and was in fact becoming more predictive, as evidenced by the steady decline in validation loss and rise in validation accuracy. The model's strong performance was probably enhanced by the use of ReduceLROnPlateau and EarlyStopping callbacks, which guaranteed ideal learning rates during the training phase and prevented overtraining. These results highlight the value of carefully controlling

learning rates and adding task-specific custom layers to pre-trained models to improve their performance. Future studies could examine how well the model performs in real-world scenarios or on a more varied and difficult dataset to verify its suitability and dependability for autonomous driving systems.

As for the ResNet-152 model, the test accuracy is very much lower than the training accuracy which suggests that the model is overfitting. It occurs when the model learns the training data so well that it even learns the noise and cannot generalize to unseen data. Although we have tried a much lower learning rate, there may be possibilities for improvement. To address this issue of overfitting, some strategies can be followed to enhance the model's ability to not just fit on the training data but also perform well on the new and unseen data:

1. Regularization techniques can be used (other than freezing batch norm layers) such as dropouts and L2 regularization
2. Increasing the diversity of the training data through data augmentation
3. Simplifying the model in case it is too complex for the training data through the tuning of hyperparameters.

The study showed that by making use of its sophisticated architecture and ImageNet pre-training, the EfficientNet-B7 model is very successful at classifying traffic signs. By adaptively adjusting the learning rate, the Adam optimizer enabled optimal training dynamics. A different model configuration that kept the batch normalization weights intact revealed a calculated method to improve generalization. This study provides evidence for the efficacy of EfficientNet-B7 and provides guidance on how to best optimize intricate models for particular applications.

7. Conclusion

This research has delved into the capabilities of the lightweight models MobileNet V2 and MobileNet V3. As expected, the EfficientNet-B7 model showed the highest performance, but MobileNet came very close, showcasing its remarkable efficiency. The ResNet-152 model, however, despite its numerous layers and large parameter size and computation expense, struggled to generalize the data. These results demonstrate why choosing the correct model based on specific operational and resource availability conditions is crucial in the real world. In the future, more could be explored regarding the optimization of these models, potentially incorporating real-time adaptive learning algorithms which will then improve accuracy and robustness under varied operating conditions. Moreover, including more challenging and diversified scenarios in the dataset may help make models that are more resilient to environmental changes and more reliable in many different use-cases in real life. Thus, this research paves the way for the development of safer and more reliable autonomous vehicles based on targeted and efficient usage of deep learning in this field.

Author Contributions: All the authors have contributed equally.

Funding: Not applicable.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank German Traffic Sign Recognition Benchmark (GTSRB) for their diverse dataset for accurate traffic sign detection

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Haltakov, V.; Mayr, J.; Unger, C.; Ilic, S. Semantic Segmentation Based Traffic Light Detection at Day and at Night. In Proceedings of the Pattern Recognition, Lecture Notes in Computer Science, Vol. 9358, Springer, 2015; pp. 446–457.
2. Du, X.-P.; Xiong, H.; Li, X.-F. Traffic Light Recognition Based on Prior Knowledge and Optimized Threshold Segmentation. *J. Comput.* **2017**, *28*(2), 197–205.
3. Vitas, D.; Tomic, M.; Burul, M. Image Augmentation Techniques for Cascade Model Training. In Proceedings of the Zooming Innovations in Consumer Technologies Conference, May 2018; pp. 78–83.
4. Weber, M.; Huber, M.; Zollner, J.M. HDTLR: A CNN Based Hierarchical Detector for Traffic Lights. In Proceedings of the 21st International Conference on Intelligent Transportation Systems, 2018; pp. 255–260.
5. Saini, S.; Nikhil, S.; Konda, K.R.; Bharadwaj, H.S.; Ganeshan, N. An Efficient Vision-Based Traffic Light Detection and State Recognition for Autonomous Vehicles. In Proceedings of the IEEE Intelligent Vehicle Symposium, June 2017; pp. 606–611.
6. Vitas, D.; Tomic, M.; Burul, M. Traffic Light Detection in Autonomous Driving Systems. *IEEE Consumer Electronics Magazine* **2020**, *9*(4), 90–96, doi: 10.1109/MCE.2020.2969156.
7. Kulkarni, R.; Dhavalikar, S.; Bangar, S. Traffic Light Detection and Recognition for Self Driving Cars Using Deep Learning. 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), Pune, India, 2018; pp. 1–4, doi: 10.1109/ICCUBEA.2018.8697819.
8. Wang, X.; Wang, D.; Li, S.; Li, S.; Zeng, P.; Liang, X. Low-light Traffic Objects Detection for Automated Vehicles. 2022 6th CAA International Conference on Vehicular Control and Intelligence (CVCI), Nanjing, China, 2022; pp. 1–5, doi: 10.1109/CVCI56766.2022.9964586.