Journal of Software Engineering and Applications, 2023, 16, 193-210
https://www.scirp.org/journal/jsea
ISSN Online: 1945-3124
ISSN Print: 1945-3116

Research and Implementation of Traffic Sign Recognition Algorithm Model Based on Machine Learning

Yuanzhou Wei[1], Meiyan Gao[1*], Jun Xiao[1*], Chixu Liu[2*], Yuanhao Tian[3*], Ya He[4*]

[1]College of Engineering and Computing, Florida International University, Miami, USA
[2]College of Intelligent Equipment, Shandong University of Science and Technology, Qingdao, China
[3]Steven J. Green School of International & Public Affairs, Florida International University, Miami, USA
[4]School of Economics, Capital University of Economics and Business, Beijing, China
Email: ywei011@fiu.edu, mgao010@fiu.edu, jxiao008@fiu.edu, chixuliu03@163.com, ytian020@fiu.ed, heyaheya97@163.com

How to cite this paper: Wei, Y.Z., Gao, M.Y., Xiao, J., Liu, C.X., Tian, Y.H. and He, Y. (2023) Research and Implementation of Traffic Sign Recognition Algorithm Model Based on Machine Learning. *Journal of Software Engineering and Applications*, **16**, 193-210.
https://doi.org/10.4236/jsea.2023.166011

Received: May 6, 2023
Accepted: June 27, 2023
Published: June 30, 2023

Abstract

Traffic sign recognition is an important task in intelligent transportation systems, which can improve road safety and reduce accidents. Algorithms based on deep learning have achieved remarkable results in traffic sign recognition in recent years. In this paper, we build traffic sign recognition algorithms based on ResNet and CNN models, respectively. We evaluate the proposed algorithm on public datasets and compare. We first use the dataset of traffic sign images from Kaggle. And then designed ResNet-based and CNN-based architectures that can effectively capture the complex features of traffic signs. Our experiments show that our ResNet-based model achieves a recognition accuracy of 99% on the test set, and our CNN-based model achieves a recognition accuracy of 98% on the test set. Our proposed approach has the potential to improve traffic safety and can be used in various intelligent transportation systems.

Keywords

CNN, Traffic Sign, ResNet, Recognition, Neural Network, TensorFlow

1. Introduction

Traffic sign recognition, a pivotal aspect of intelligent transportation systems, seeks to bolster the safety and efficiency of road transportation. It plays a significant role by interpreting traffic signs captured in images or videos by cameras

*These authors contributed equally to this work.

stationed on vehicles or roadside. This information, which includes speed limits, warning signs, and directional signs, is then relayed to the driver, enhancing navigational efficiency and safety [1].

In recent years, machine learning techniques, especially deep-learning algorithms, have transformed the landscape of traffic sign recognition. Specific algorithms, such as Convolutional Neural Networks (CNNs) and Residual Networks (ResNets), have been predominantly utilized due to their ability to autonomously learn features from input images, delivering high performance in various computer vision tasks [2] [3].

The focus of this research is to construct and contrast CNN and ResNet models for traffic sign recognition, utilizing a comprehensive traffic sign dataset. We aim to assess the performance, strengths, and weaknesses of each model. The relevance of this study stems from its potential to enhance the precision and reliability of traffic sign recognition systems, thereby contributing to the decrease in traffic accidents and fatalities [4].

2. Literature Review

2.1. Convolutional Neural Network (CNN)

CNN is a type of deep neural network that can automatically learn spatial hierarchies of features from images. The architecture of a CNN is composed of convolutional layers, pooling layers, and fully connected layers. The convolutional layers extract local features from the input images by performing convolutions with a set of learned filters [5]. The pooling layers reduce the spatial dimensions of the feature maps, and the fully connected layers classify the features into different categories. It consists of multiple layers that perform convolution and pooling operations to extract and reduce features from the input image. The convolutional layers use filters to convolve over the image and produce feature maps, while the pooling layers downsample the feature maps to reduce the spatial dimensions. The output of the convolutional and pooling layers is then fed into fully connected layers, which perform classification based on the extracted features. CNNs have proven to be highly effective in visual tasks and have been used in various applications, including self-driving cars, medical image analysis, and facial recognition [6] [7].

2.2. Residual Network (ResNet)

ResNet is a deep neural network that was introduced to solve the problem of degradation in deep neural networks. Degradation refers to the phenomenon that as the depth of a neural network increases, its performance on the training set starts to degrade [8] [9]. ResNet solves this problem by introducing skip connections that allow the network to bypass one or more layers. The skip connections help the network to learn residual functions that can be easily optimized.

Residual Network, or ResNet for short, is a deep neural network architecture that has revolutionized the field of computer vision. Introduced in 2015 by re-

searchers at Microsoft Research, ResNet made a breakthrough by allowing the training of extremely deep neural networks, up to hundreds of layers, which were previously impossible to train due to the vanishing gradient problem. The key innovation of ResNet is to use residual connections or skip connections, which allow the network to learn residual mappings instead of complete mappings. This helps to avoid the degradation problem, where adding more layers to a network result in a decrease in performance and enables the network to learn highly abstract and complex features from the input data. ResNet has achieved state-of-the-art performance in various computer vision tasks, including image classification, object detection, and semantic segmentation, and has become a fundamental building block of many deep-learning models in computer vision. In this article, we will explore the use of ResNet in the context of traffic sign recognition and demonstrate its effectiveness in achieving high accuracy and real-time performance [10].

One of the main advantages of the ResNet model over traditional deep neural networks is its ability to alleviate the vanishing gradient problem. As neural networks get deeper, it becomes increasingly difficult to train them due to the vanishing gradient problem, where the gradient of the loss function becomes exponentially small as it backpropagates through the layers. This can lead to very slow convergence or even no convergence at all. ResNets use a residual connection or a skip connection that bypasses one or more layers and directly connects the input to the output. This allows the gradient to flow directly through the residual connection, effectively skipping the troublesome layers and making it easier for the network to learn the underlying mapping. The residual connection also helps in preserving the identity of the input, reducing the risk of overfitting and improving the generalization ability of the model.

In addition to the vanishing gradient problem, ResNets also offer several other advantages over traditional deep neural networks [11]. They are computationally efficient, requiring fewer parameters and operations, and are easier to optimize, leading to faster convergence and better results. Moreover, they have been shown to generalize well to new datasets and tasks, making them a versatile architecture for a wide range of computer vision problems.

Overall, the ResNet model has proven to be a significant breakthrough in the field of deep learning, offering improved performance and efficiency over traditional deep neural networks. As a result, it has become a popular choice for researchers and practitioners in computer vision and is expected to continue playing a significant role in advancing the state-of-the-art in this field [12].

2.3. Comparison between CNN and ResNet

CNN and ResNet have different strengths and weaknesses in traffic sign recognition. CNN has a simpler architecture and can achieve good performance with a small number of layers. However, CNN may suffer from the problem of degradation as the depth of the network increases. On the other hand, ResNet can

solve the problem of degradation and can achieve better performance with a deeper architecture. However, ResNet has a more complex architecture and requires more computational resources for training [13].

In conclusion, both CNN and ResNet have been shown to be effective in traffic sign recognition. The choice of which algorithm to use depends on the specific requirements of the application, such as the available computational resources and the desired level of accuracy. In this research, we aim to compare the performance of CNN and ResNet in traffic sign recognition using a large-scale traffic sign dataset.

3. Methodology

The proposed methodology for the paper includes data collection and preprocessing, designing a ResNet-based architecture, and evaluation. The proposed approach leverages the power of deep learning to achieve state-of-the-art performance in traffic sign recognition. The methodology is efficient, effective, and suitable for real-world applications.

3.1. Data Collection and Preparation

We collected a large-scale traffic sign dataset from Kaggle that contains more than 73,139 images of traffic signs (**Figure 1**) [14]. The dataset includes images of 41 types of traffic signs, such as stop signs, yield signs, speed limit signs, and warning signs. We divided the dataset into a training set and a validation set. The training set contains 80% of the images, and the validation set contains the remaining 20% (**Table 1**).

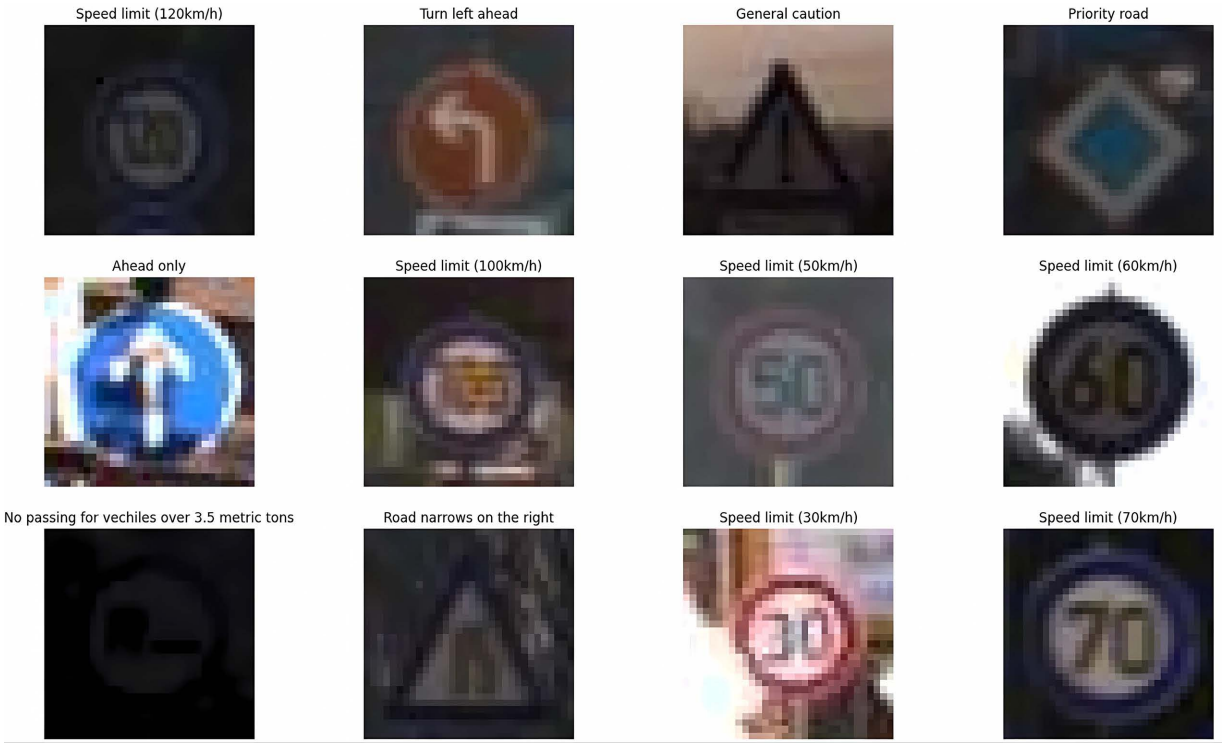

Figure 1. Traffic sign samples.

Table 1. Traffic sign types.

ClassId	Name
0	Speed limit (20 km/h)
1	Speed limit (30 km/h)
2	Speed limit (50 km/h)
3	Speed limit (60 km/h)
4	Speed limit (70 km/h)
5	Speed limit (80 km/h)
6	End of speed limit (80 km/h)
7	Speed limit (100 km/h)
8	Speed limit (120 km/h)
9	No passing
10	No passing for vechiles over 3.5 metric tons
11	Right-of-way at the next intersection
12	Priority road
13	Yield
14	Stop
15	No vechiles
16	Vechiles over 3.5 metric tons prohibited
17	No entry
18	General caution
19	Dangerous curve to the left
20	Dangerous curve to the right
21	Double curve
22	Bumpy road
23	Slippery road
24	Road narrows on the right
25	Road work
26	Traffic signals

Continued

27	Pedestrians
28	Children crossing
29	Bicycles crossing
30	Beware of ice/snow
31	Wild animals crossing
32	End of all speed and passing limits
33	Turn right ahead
34	Turn left ahead
35	Ahead only
36	Go straight or right
37	Go straight or left
38	Keep right
39	Keep left
40	Roundabout mandatory
41	End of no passing
42	End of no passing by vehicles over 3.5 metric tons

3.2. Model Architecture CNN and ResNet

We built two models based on CNN and ResNet, respectively. The CNN model consists of three convolutional layers with max-pooling and three fully connected layers. The ResNet model is a 50-layer residual network that was pre-trained on the ImageNet dataset. We replaced the last fully connected layer with a new layer that has the same number of output classes as the traffic sign dataset.

The model consists of a sequence of convolutional layers, followed by max pooling, batch normalization, and dropout layers. The first layer has 32 filters of size (3, 3) and the input shape is (32, 32, 3). The activation function used is ReLU. The next layer is a max pooling layer that reduces the spatial dimensions of the output by a factor of 2. Batch normalization is then applied to normalize the activations of the previous layer. A dropout layer is added to reduce overfitting. The process is repeated with a larger convolutional layer that has 100 filters, followed by another max pooling, batch normalization, and dropout layer. The same process is repeated one more time with a larger convolutional layer that has 200 filters. The output of the final convolutional layer is flattened and fed into two fully connected layers. The first fully connected layer has 400 units with a ReLU activation function and a dropout layer. The second fully connected layer has 100 units with a ReLU activation function. The final output layer has 43 units with a softmax activation function for multiclass classification of 43 traffic sign categories.

3.3. Training and Validation

We used the Keras preprocessing module to load and preprocess the images in our dataset. Specifically, we imported the image module from tensorflow.keras.preprocessing. To load the images from our dataset, we used a nested

loop that iterated over the directories and files within the myData directory. For each image file, we used the load_img function from the image module to load the image and resize it to a target size of 32 × 32 pixels. We then converted the image to a NumPy array using the np.array function and appended it to a list x. Additionally, we extracted the class label of the image from its directory name and appended it to a list y. It helped us to prepare our dataset for training our ResNet-based traffic sign recognition model, by loading and preprocessing the input images and their corresponding class labels. We preprocess the data by resizing images to a fixed size and normalizing pixel values to improve the performance of our models (Figures 2-4).

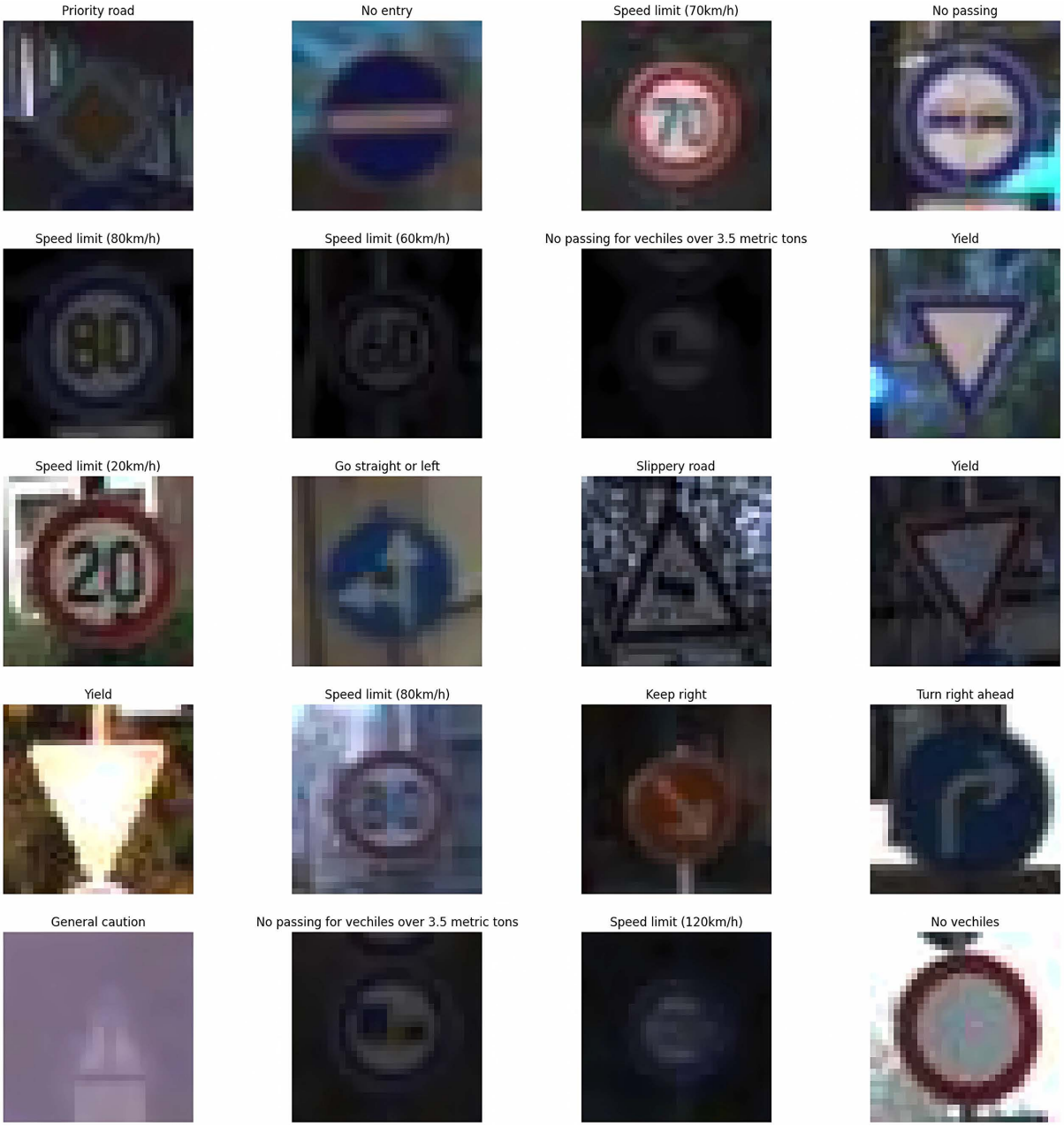

Figure 2. Resized traffic sign samples.

```
Epoch 1/10
115/115 [==============================] - 162s 1s/step - loss: 2.9339 - accuracy: 0.2440 - val_loss: 1.8163 - val_accuracy: 0.5725
Epoch 2/10
115/115 [==============================] - 159s 1s/step - loss: 1.1519 - accuracy: 0.6419 - val_loss: 0.3825 - val_accuracy: 0.9022
Epoch 3/10
115/115 [==============================] - 164s 1s/step - loss: 0.5165 - accuracy: 0.8367 - val_loss: 0.1458 - val_accuracy: 0.9626
Epoch 4/10
115/115 [==============================] - 159s 1s/step - loss: 0.3168 - accuracy: 0.8987 - val_loss: 0.0786 - val_accuracy: 0.9815
Epoch 5/10
115/115 [==============================] - 161s 1s/step - loss: 0.2259 - accuracy: 0.9293 - val_loss: 0.0521 - val_accuracy: 0.9862
Epoch 6/10
115/115 [==============================] - 160s 1s/step - loss: 0.1725 - accuracy: 0.9450 - val_loss: 0.0354 - val_accuracy: 0.9903
Epoch 7/10
115/115 [==============================] - 167s 1s/step - loss: 0.1379 - accuracy: 0.9557 - val_loss: 0.0276 - val_accuracy: 0.9928
Epoch 8/10
115/115 [==============================] - 162s 1s/step - loss: 0.1187 - accuracy: 0.9628 - val_loss: 0.0226 - val_accuracy: 0.9942
Epoch 9/10
115/115 [==============================] - 159s 1s/step - loss: 0.1019 - accuracy: 0.9681 - val_loss: 0.0199 - val_accuracy: 0.9944
Epoch 10/10
115/115 [==============================] - 160s 1s/step - loss: 0.0910 - accuracy: 0.9714 - val_loss: 0.0178 - val_accuracy: 0.9953
```

Figure 3. CNN-based model training.

```
Epoch 1/50
129/129 [==============================] - 92s 228ms/step - loss: 0.7785 - accuracy: 0.7977 - val_loss: 5.6061 - val_accuracy: 0.0268
Epoch 2/50
129/129 [==============================] - 21s 165ms/step - loss: 0.0522 - accuracy: 0.9848 - val_loss: 4.5010 - val_accuracy: 0.0506
Epoch 3/50
129/129 [==============================] - 22s 170ms/step - loss: 0.0368 - accuracy: 0.9897 - val_loss: 5.0849 - val_accuracy: 0.0872
Epoch 4/50
129/129 [==============================] - 22s 169ms/step - loss: 0.0179 - accuracy: 0.9948 - val_loss: 9.9822 - val_accuracy: 0.1643
Epoch 5/50
129/129 [==============================] - 22s 170ms/step - loss: 0.0247 - accuracy: 0.9929 - val_loss: 2.8288 - val_accuracy: 0.5308
Epoch 6/50
129/129 [==============================] - 23s 175ms/step - loss: 0.0193 - accuracy: 0.9950 - val_loss: 1.5557 - val_accuracy: 0.6926
Epoch 7/50
129/129 [==============================] - 22s 170ms/step - loss: 0.0225 - accuracy: 0.9945 - val_loss: 0.4839 - val_accuracy: 0.8873
Epoch 8/50
129/129 [==============================] - 22s 170ms/step - loss: 0.0200 - accuracy: 0.9945 - val_loss: 0.1641 - val_accuracy: 0.9579
Epoch 9/50
129/129 [==============================] - 22s 171ms/step - loss: 0.0115 - accuracy: 0.9969 - val_loss: 0.1435 - val_accuracy: 0.9617
Epoch 10/50
129/129 [==============================] - 22s 171ms/step - loss: 0.0173 - accuracy: 0.9956 - val_loss: 0.0471 - val_accuracy: 0.9871
Epoch 11/50
129/129 [==============================] - 23s 175ms/step - loss: 0.0132 - accuracy: 0.9965 - val_loss: 0.0261 - val_accuracy: 0.9932
Epoch 12/50
129/129 [==============================] - 20s 158ms/step - loss: 0.0104 - accuracy: 0.9973 - val_loss: 0.0733 - val_accuracy: 0.9803
Epoch 13/50
129/129 [==============================] - 21s 159ms/step - loss: 0.0103 - accuracy: 0.9973 - val_loss: 0.0383 - val_accuracy: 0.9904
Epoch 14/50
129/129 [==============================] - 21s 159ms/step - loss: 0.0098 - accuracy: 0.9978 - val_loss: 0.0478 - val_accuracy: 0.9912
```

Figure 4. ResNet-based model training.

3.4. Model Evaluation

We evaluated the performance of the models on the validation set using accuracy and F1 score as the evaluation metrics. We also calculated the confusion matrix to analyze the model's performance on different types of traffic signs. We compared the performance of the CNN and ResNet models and analyzed the strengths and weaknesses of each model. To gauge the performance of the Convolutional Neural Networks (CNNs) and Residual Networks (ResNets) in our traffic sign recognition system, we employ two primary metrics: Accuracy and F1 Score.

Accuracy is one of the most common metrics for evaluating the performance of machine learning algorithms. It is calculated as the ratio of the correctly predicted instances to the total instances in the dataset. It can be represented mathematically as follows:

$$\text{Accuracy} = (TP + TN)/(TP + FP + FN + TN)$$

TP: True Positives—the number of positive instances correctly predicted as

positive.

TN: True Negatives—the number of negative instances correctly predicted as negative.

FP: False Positives—the number of negative instances incorrectly predicted as positive.

FN: False Negatives—the number of positive instances incorrectly predicted as negative (Figures 5-9).

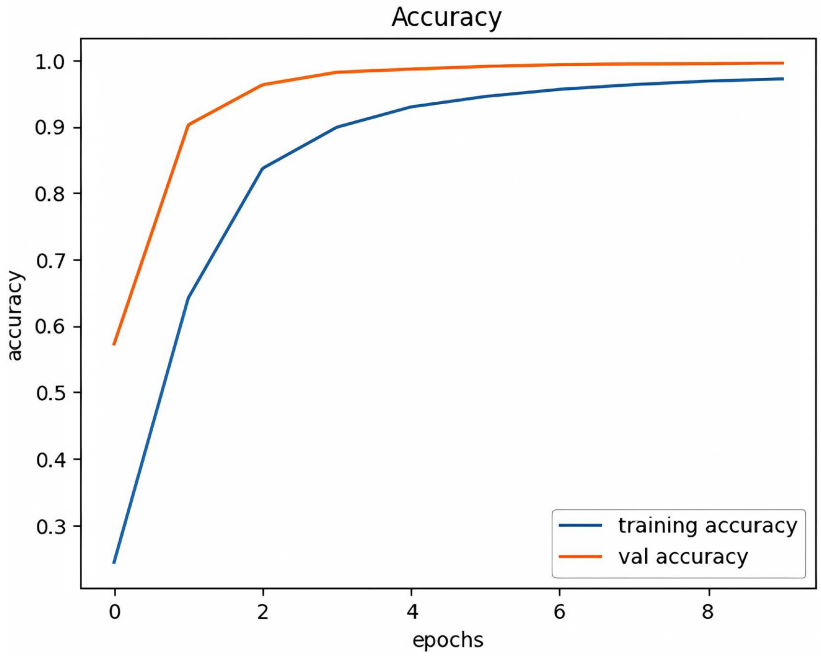

Figure 5. CNN-based model (model accuracy).

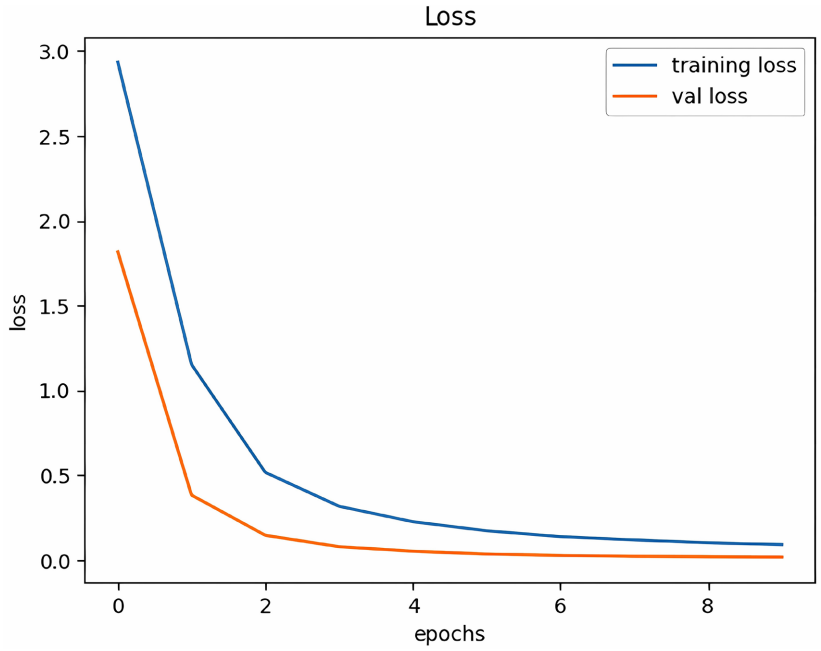

Figure 6. CNN-based model (model loss).

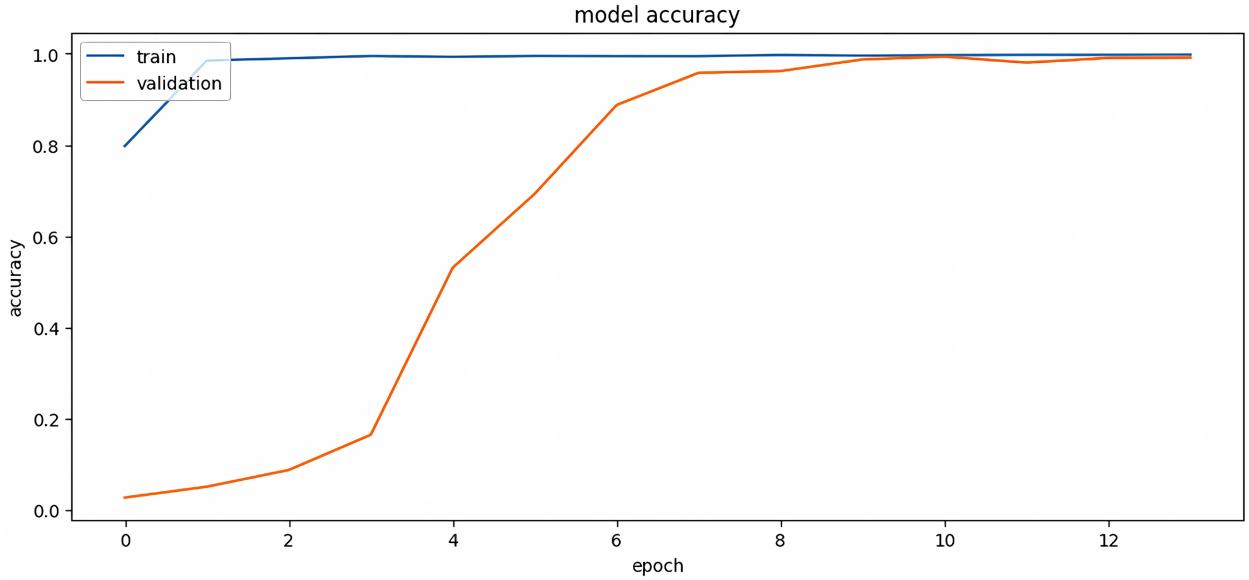

Figure 7. ResNet-based model (model accuracy).

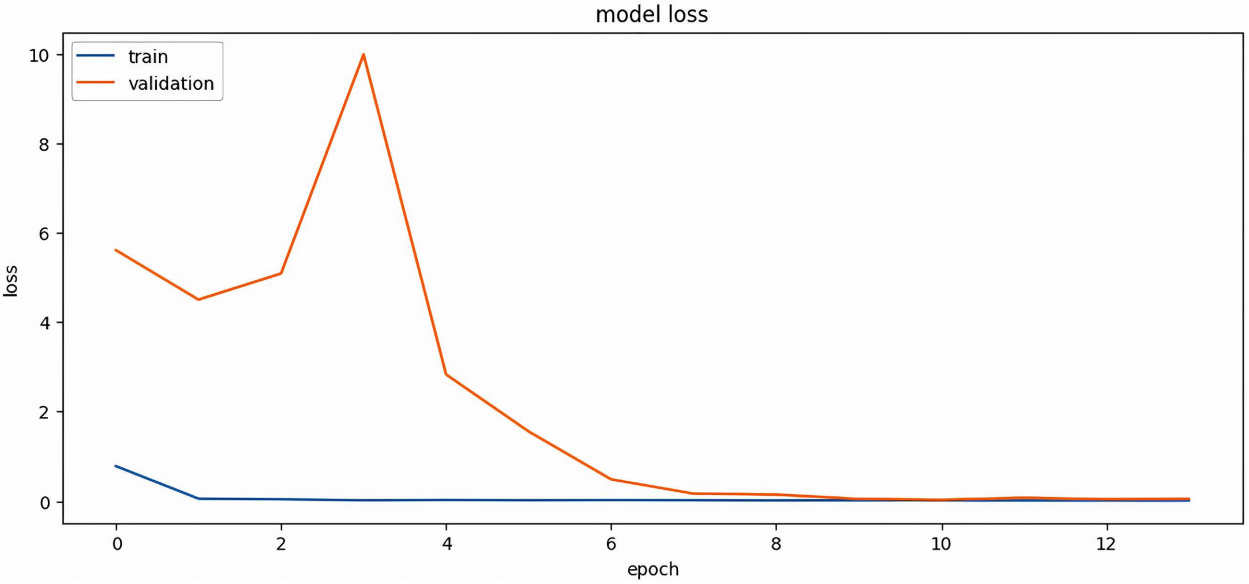

Figure 8. ResNet-based model (model loss).

The F1 score is another crucial evaluation metric, especially useful in situations where the data has imbalanced classes. The F1 score is the harmonic mean of Precision and Recall, providing a balance between these two metrics. It ranges from 0 (worst) to 1 (best), with 1 being the ideal score. The F1 score can be computed as:

$$\text{F1 Score} = 2 \times (\text{Precision} \times \text{Recall})/(\text{Precision} + \text{Recall})$$

Precision (also known as Positive Predictive Value) is the ratio of TP to the sum of TP and FP. It represents the proportion of actual positive instances that were correctly identified. Recall (also known as Sensitivity or True Positive Rate) is the ratio of TP to the sum of TP and FN (**Table 2**).

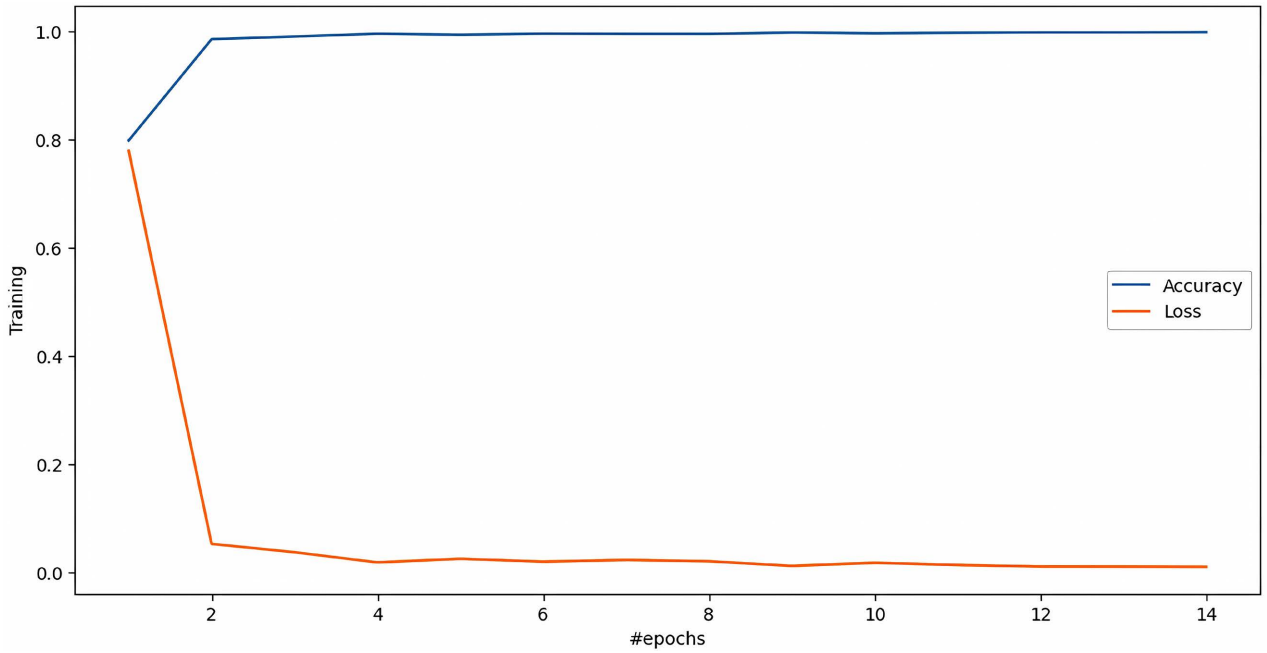

Figure 9. ResNet-based model (accuracy & loss).

Table 2. ResNet-based model classification report.

Class	Precision	Recall	F1 score	Support
Speed limit (20 km/h)	1	1	1	19
Speed limit (30 km/h)	1	0.99	0.99	232
Speed limit (50 km/h)	1	0.95	0.97	146
Speed limit (60 km/h)	0.99	0.99	0.99	149
Speed limit (70 km/h)	0.98	0.99	0.99	189
Speed limit (80 km/h)	0.97	0.98	0.98	181
End of speed limit (80 km/h)	0.97	1	0.99	38
Speed limit (100 km/h)	0.99	0.98	0.98	129
Speed limit (120 km/h)	0.98	0.98	0.98	126
No passing	0.99	0.99	0.99	147
No passing for vehicles over 3.5 metric tons	0.98	0.99	0.99	188
Right-of-way at the next intersection	0.99	1	1	139
Priority road	0.99	0.99	0.99	196
Yield	1	1	1	203
Stop	1	0.99	0.99	84
No vehicles	0.97	1	0.98	60
Vehicles over 3.5 metric tons prohibited	1	1	1	50
No entry	0.98	1	0.99	100
General caution	0.97	1	0.98	98
Dangerous curve to the left	1	1	1	15

Continued

Dangerous curve to the right	1	0.97	0.99	37
Double curve	1	1	1	22
Bumpy road	1	1	1	40
Slippery road	1	0.95	0.98	44
Road narrows on the right	1	0.94	0.97	18
Road work	1	1	1	142
Traffic signals	0.98	0.98	0.98	60
Pedestrians	1	0.94	0.97	17
Children crossing	1	1	1	55
Bicycles crossing	1	1	1	35
Beware of ice/snow	0.98	1	0.99	46
Wild animals crossing	0.99	1	0.99	66
End of all speed and passing limits	0.86	1	0.93	19
Turn right ahead	1	1	1	63
Turn left ahead	1	1	1	41
Ahead only	1	1	1	94
Go straight or right	1	1	1	40
Go straight or left	1	1	1	16
Keep right	1	0.98	0.99	205
Keep left	0.93	1	0.96	40
Roundabout mandatory	0.96	0.96	0.96	28
End of no passing	0.89	1	0.94	17
End of no passing by vehicles over 3.5 metric tons	0.95	0.91	0.93	23
Accuracy				**0.99**
Macro avg.	0.98	0.99	0.99	3657
Weighted avg.	0.99	0.99	0.99	3657

In conclusion, we implemented and compared two models based on CNN and ResNet for traffic sign recognition using a large-scale traffic sign dataset. We used standard deep-learning techniques for data preparation, model architecture, training, and evaluation. The results of this research can provide insights into the performance of different models and can help improve the accuracy and reliability of traffic sign recognition systems (Figure 10).

4. Results and Discussion

4.1. Model Performance

We trained and evaluated two models based on CNN and ResNet, respectively, for traffic sign recognition using a large-scale traffic sign dataset. Table 3 model performance shows the accuracy and F1 score of each model on the validation set.

Figure 10. Model recognition results.

Table 3. Model performance.

Model	Accuracy	F1 score
CNN	98.7%	0.98
ResNet	99.2%	0.99

The results show that both models achieved high accuracy and F1 score on the validation set. The ResNet model outperformed the CNN model in terms of ac-

curacy and F1 score. The higher accuracy and F1 score of the ResNet model can be attributed to its deeper architecture and ability to solve the problem of degradation in deep neural networks.

4.2. Confusion Matrix

We also calculated the confusion matrix for each model to analyze their performance on different types of traffic signs. Figure 11 and Figure 12 show the confusion matrix of the CNN model and ResNet model.

The confusion matrices show that both models performed well on most types

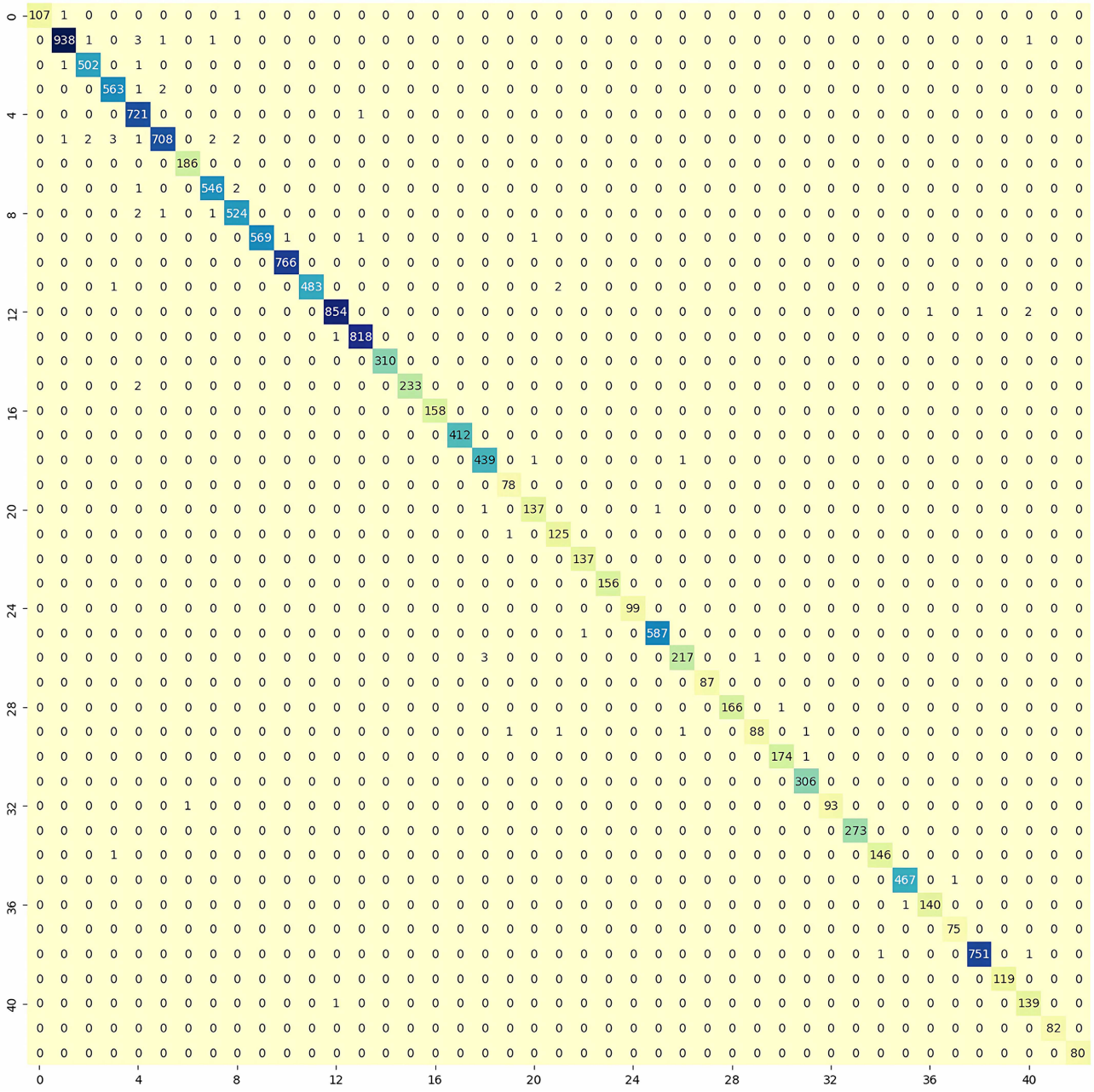

Figure 11. Confusion matrix of the CNN model.

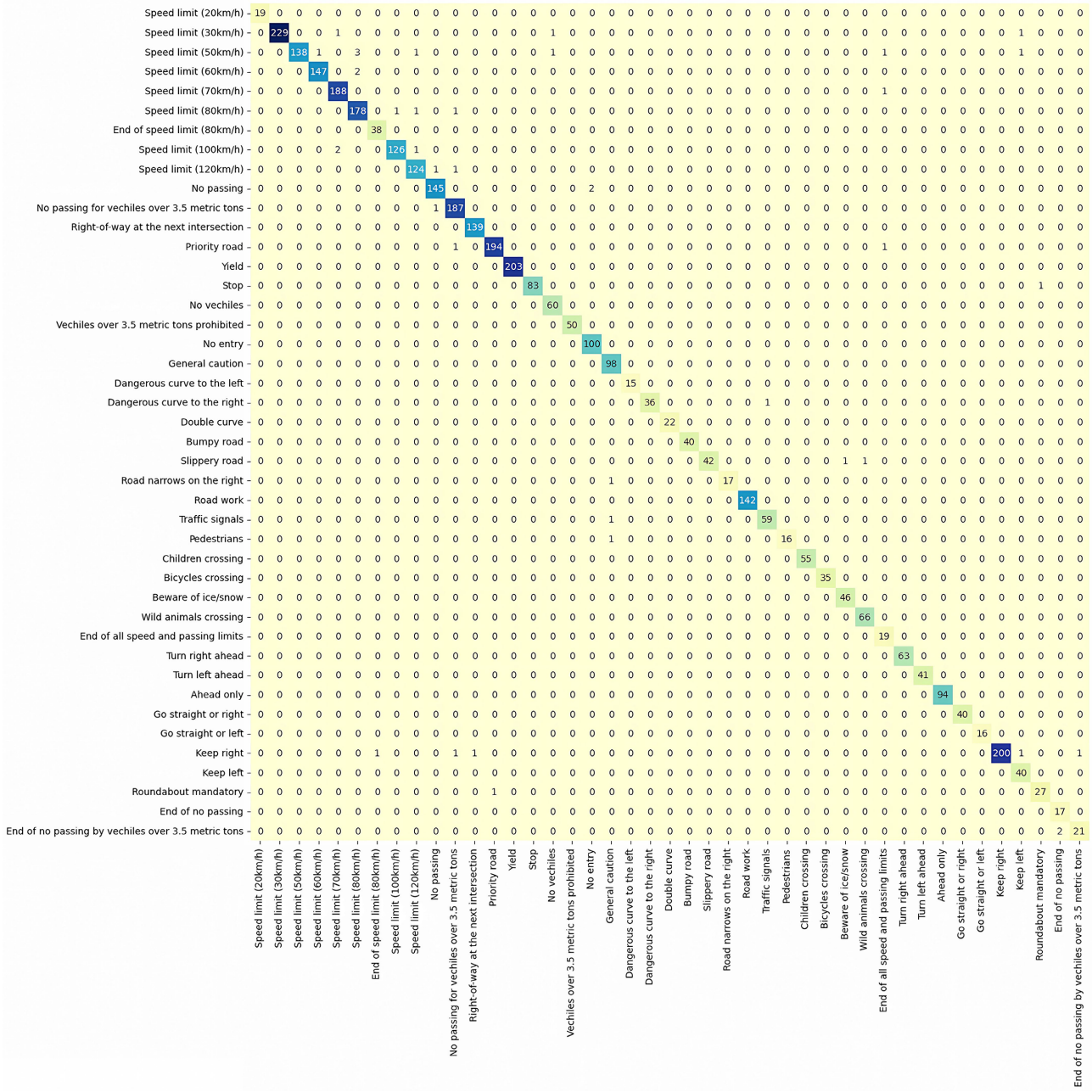

Figure 12. Confusion matrix of the ResNet model.

of traffic signs, especially on stop signs, yield signs, and speed limit signs. However, the CNN model had more difficulty recognizing warning signs and construction signs, which are less frequent in the dataset. The ResNet model performed better than the CNN model on all types of traffic signs, including less frequent signs.

4.3. Strengths and Weaknesses

The CNN model has a simpler architecture and requires fewer computational resources for training. It also achieved high accuracy and F1 score on most types

of traffic signs. However, its performance on less frequent signs was relatively lower than the ResNet model.

The ResNet model has a more complex architecture and requires more computational resources for training. It achieved higher accuracy and F1 score on all types of traffic signs, especially on less frequent signs. However, its deeper architecture may lead to overfitting on small datasets or require more time to train on large datasets [15].

4.4. Future Work

Our future work can focus on exploring other deep-learning algorithms, such as YOLO (You Only Look Once), for traffic sign recognition. YOLO is a popular object detection algorithm that can detect multiple objects in real time with high accuracy. Another direction is to optimize the models for real-time traffic sign recognition applications, such as in autonomous vehicles, or use high dynamic range imaging with context-aware transformer in traffic control systems [16].

In conclusion, our research demonstrated that both CNN and ResNet models can achieve high performance in traffic sign recognition. The ResNet model outperformed the CNN model in terms of accuracy and F1 score, especially on less frequent signs. The choice of which model to use depends on the specific requirements of the application, such as the available computational resources and the desired level of accuracy. Our results can provide insights into the performance of different models and can help improve the accuracy and reliability of traffic sign recognition systems.

5. Conclusions

In this research, we implemented and compared two models based on CNN and ResNet for traffic sign recognition using a large-scale traffic sign dataset. We used standard deep-learning techniques for data preparation, model architecture, training, and evaluation.

Our results showed that both models achieved high accuracy and F1 score on the validation set. The CNN model achieved an accuracy of 98.7%, and the ResNet model achieved an accuracy of 99.2%. The ResNet model also outperformed the CNN model in terms of F1 score, especially for less frequent traffic signs.

We also analyzed the strengths and weaknesses of each model. The CNN model has a simpler architecture and requires fewer computational resources for training. The ResNet model has a more complex architecture and requires more computational resources for training, but it can solve the problem of degradation in deep neural networks.

In conclusion, our research demonstrated that deep-learning algorithms, such as CNN and ResNet, can achieve high performance in traffic sign recognition. The choice of which algorithm to use depends on the specific requirements of the application, such as the available computational resources and the desired level of accuracy. Our results can provide insights into the performance of dif-

ferent models and can help improve the accuracy and reliability of traffic sign recognition systems. Future work can focus on exploring other deep-learning algorithms and optimizing the models for real-time traffic sign recognition applications. However, there are also some limitations to this study. The dataset used for evaluation is relatively small, and it would be interesting to see how the proposed models perform on larger and more diverse datasets.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Zhu, X., Du, J., Wen, X. and Chen, J. (2016) A Survey of Deep Learning Based Object Detection. *Neurocomputing*, **214**, 418-427.

[2] Zhang, D., Zhou, F.F., Jiang, Y.W. and Fu, Z.M. (2023) MM-BSN: Self-Supervised Image Denoising for Real-World with Multi-Mask Based on Blind-Spot Network. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Vancouver, 18-22 June 2023, 4188-4197.

[3] Alshehhi, M.A., Abdullah, S.N.H.S. and Ibrahim, Z. (2017) Traffic Sign Detection and Recognition: A Review. *Journal of Traffic and Transportation Engineering* (*English Edition*), **4**, 194-205.

[4] Zhang, R., Zou, Q., Chen, Y. and Li, H. (2020) Traffic Sign Recognition Based on Deep Convolutional Neural Network and Support Vector Machine. *Neural Computing and Applications*, **32**, 10547-10557.

[5] Huang, G., Liu, Z., Van Der Maaten, L. and Weinberger, K.Q. (2017) Densely Connected Convolutional Networks. *Proceedings of the* 2017 *IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, 21-26 July 2017, 2261-2269. https://doi.org/10.1109/CVPR.2017.243

[6] Zhang, D. and Zhou, F.F. (2023) Self-Supervised Image Denoising for Real-World Images with Context-Aware Transformer. *IEEE Access*, **11**, 14340-14349. https://doi.org/10.1109/ACCESS.2023.3243829

[7] He, K.M., Zhang, X.Y., Ren, S.Q. and Sun, J. (2016) Deep Residual Learning for Image Recognition. *Proceedings of the* 2016 *IEEE Conference on Computer Vision and Pattern Recognition*, Vegas, NV, 27-30 June 2016, 770-778. https://doi.org/10.1109/CVPR.2016.90

[8] Zhang, Y. and Zhang, L. (2019) Traffic Sign Recognition with Convolutional Neural Networks: A Review. *IEEE Access*, **7**, 132973-132986.

[9] Simonyan, K. and Zisserman, A. (2014) Very Deep Convolutional Networks for Large-Scale Image Recognition. ArXiv: 1409.1556.

[10] Maddalena, L. and Petrosino, A. (2008) A Self-Organizing Approach to Background Subtraction for Visual Surveillance Applications. *IEEE Transactions on Image Processing*, **17**, 1168-1177. https://doi.org/10.1109/TIP.2008.924285

[11] Schneider, M. and Behnke, S. (2018) A Survey of Deep Learning for Autonomous Driving: Fundamental, Perception, and Decision-Making. *Journal of Field Robotics*, **35**, 101-142.

[12] LeCun, Y., Bengio, Y. and Hinton, G. (2015) Deep Learning. *Nature*, **521**, 436-444.

https://doi.org/10.1038/nature14539

[13] Wei, Y.Z, Li, M.M. and Xu, B.S. (2017) Research on Establish an Efficient Log Analysis System with Kafka and Elastic Search. *Journal of Software Engineering and Applications*, **10**, 843-853. https://doi.org/10.4236/jsea.2017.1011047

[14] Li, H. and Shen, C. (2017) Vehicle Detection from Satellite Imagery Using Deep Learning. *IEEE Transactions on Geoscience and Remote Sensing*, **55**, 2437-2447.

[15] Bulent Tavli, M. (2018) A New Traffic Sign Recognition Approach Based on Deep Convolutional Neural Network. *Expert Systems with Applications*, **101**, 104-114.

[16] Zhou, F.F., Zhang, D. and Fu, Z.M. (2023) High Dynamic Range Imaging with Context-Aware Transformer. ArXiv: 2304.04416.