

Traffic Sign Recognition using Capsule Network

Mutaqin Akbar¹, Agus Sidiq Purnomo¹, Supatman¹, Bernadete Deta²

¹Department of Informatics, Universitas Mercu Buana Yogyakarta, Indonesia

²Department of Informatics, Institut Keguruan dan Teknologi Larantuka, Indonesia

Article Info

Article history:

Received Apr 27, 2026

Revised June 24, 2026

Accepted July 23, 2026

Keywords:

Capsule network

Convolutional neural network

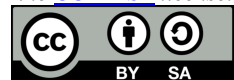
Recognition

Traffic sign

ABSTRACT

Traffic sign recognition (TSR) is critical for systems that rely on its output to inform downstream decisions. This study investigates TSR using Capsule Network (CapsNet), an advancement over the convolutional neural network (CNN) that captures spatial relationships between image features, conferring robustness to affine transformations. The proposed architecture comprises a convolutional layer (256 filters, 9×9 kernel, stride 1, ReLU activation), a primary capsule layer (32 channels of 6×6 capsules, each an 8-dimensional vector), and a class capsule layer (one 16-dimensional capsule per target class). The model was evaluated on both original and augmented datasets, the latter incorporating rotations ranging from -5 degrees to +5 degrees. On the original dataset, CapsNet achieved 100% training and testing accuracy with a training loss of 0,0048 at epoch 20. On the augmented dataset, the model achieved 100% training accuracy (loss: 0,0056) and 98% testing accuracy (5 misclassifications). Compared to multi-scale CNN (MS-CNN), which produced 7 misclassifications on the augmented dataset, CapsNet demonstrated superior consistency and robustness under affine transformations. These findings suggest that CapsNet is a viable and effective architecture for real-world TSR applications.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Mutaqin Akbar,

Department of Informatics, Universitas Mercu Buana Yogyakarta,

Jl. Jembatan Merah No.84C, Depok, Sleman, Daerah Istimewa Yogyakarta 55283, Indonesia.

Email: mutaqin@mercubuana-yogya.ac.id

1. INTRODUCTION

Traffic signs are road equipment elements that contain numbers, letters, sentences, symbols, and/or combinations that can function as advisory, prohibition, warning, command, and guide for road users. The Indonesian Ministry of Transportation regulates traffic signs through the Regulation of the Minister of Transportation concerning Traffic Signs (Number 13 of 2014). Recognition of traffic signs is crucial for certain technologies, such as traffic sign recognition (TSR). TSR used to recognize traffic signs by utilizing image processing. TSR has been applied to driver assistance systems (DAS) [1], [2], advanced driver assistance systems (ADAS) [3], [4], [5], autonomous driving systems (ADS) [6], [7], understanding urban conditions, road safety, and monitoring signs for maintenance purposes. The output from TSR can be input for DAS, ADAS, ADS, and other systems, so the accuracy of the output from TSR will significantly affect the decisions these systems take. Decision-making errors can have fatal consequences, with the worst outcome being traffic accidents that result in fatalities.

The field of TSR has attracted many researchers in recent years, especially the use of machine learning and deep learning to classify traffic signs accurately [8], [9], [10]. Some research studies the use of machine learning in TSR. Sapijaszko et al. use techniques comprised of preprocessing, feature extraction, compression, and classification (multilayer perceptron) [11]. They tested the model using three datasets, namely the Belgian traffic sign dataset (BTSD) with over 7000 images, the German traffic sign recognition benchmark (GTSRB) with over 50000

images, and the Chinese traffic sign database (TSRD) with over 6000 images. The accuracy results for each dataset are 96%, 95,7%, and 94,9%, respectively. By combining color, local binary patterns (LBP), and histograms of oriented gradients (HOG) within an artificial neural network (ANN), Kerim and Efe (2021) achieved a 95% accuracy rate on the GTSRB dataset [12]. Their multi-feature framework notably outperformed traditional methods that utilized HOG features in isolation. Another study uses the HOG feature with dimension reduction using a principal component analysis (PCA) algorithm and performs traffic sign recognition using a support vector machine (SVM) classifier [13]. They used the GTSRB dataset and obtained an accuracy level of 99,99%. Evaluating the GTSRB benchmark, Wang compared logistic regression, MLP, and SVM frameworks, which yielded respective accuracy rates of 97.75%, 98.88%, and 95.51% [14]. These findings highlight the superior predictive capabilities of the neural network approach over the traditional classifiers in this context.

Deep learning approaches for image classification typically rely on convolutional neural networks (CNN) because of their benchmark-setting accuracy. CNN itself has undergone many developments, first initiated by Yann LeCun et al. [15]. The CNN approach that has been carried out previously was able to obtain excellent accuracy values [16], [17]. CNN has a filter (kernel) that detects features such as edges, corners, or textures. However, CNN overlooks the spatial relationship between these features, such as whether the nose is positioned below the eyes. Therefore, in 2017, Geoffrey Hinton and his team initiated an artificial neural network architecture called the Capsule Network (CapsNet) [18]. CapsNet attempts to replicate how humans recognize objects' shapes and positions more accurately than CNN.

CapsNet has been applied in several fields [19]. First, CapsNet is used for image segmentation with the model name SegCaps, for example, for the segmentation of computed tomography (CT) scan images of the lungs, where the results are excellent even when an affine transformation occurs [20]. Meanwhile, the CapsNet architecture with multi-labels was also developed under the name Matwo-CapsNet, and the results were compared with SegCaps, where Matwo-CapsNet obtained results that could compete with SegCaps [21]. DRIP-Caps was then developed as an optimization of SegCaps, where this architecture has 37,85% fewer parameters than SegCaps, so that it can reduce computational complexity with competitive results [22]. In the field of image reconstruction, CapsNet delivers significantly better empirical results than traditional CNN models during digital holographic reconstruction tasks [23]. CapsNet is also developed as a discriminator in a generative adversarial network (GAN) [24]. CapsNet is also used in video segmentation for action detection [25], [26]. The development of CapsNet in the field of speech recognition has also been carried out by several researchers, where CapsNet is also able to outperform CNN [27], [28]. CapsNet models have additionally been deployed for text classification applications in natural language processing [29].

This study focuses on the development of CapsNet in traffic sign recognition, where the CapsNet architecture will be compared with previous research using a multi-scale CNN (MS-CNN). Furthermore, applying an affine transformation with rotation between -5 and 5 degrees will be carried out as a benchmark for the robustness of each architecture.

2. RESEARCH METHOD

The methodology used in this study shown in Figure 1, involves four main stages: dataset collection and preprocessing, network architecture design, model training, and finally testing and evaluation using a confusion matrix.

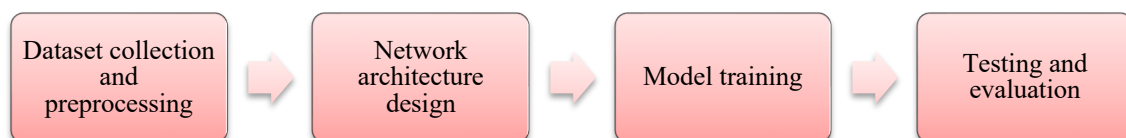
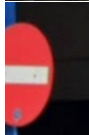
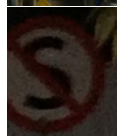


Figure 1. Research methodology

Table 1. Datasets

#	Example 1	Example 2	Example 3	#	Example 1	Example 2	Example 3
1				6			
2				7			
3				8			
4				9			
5				10			

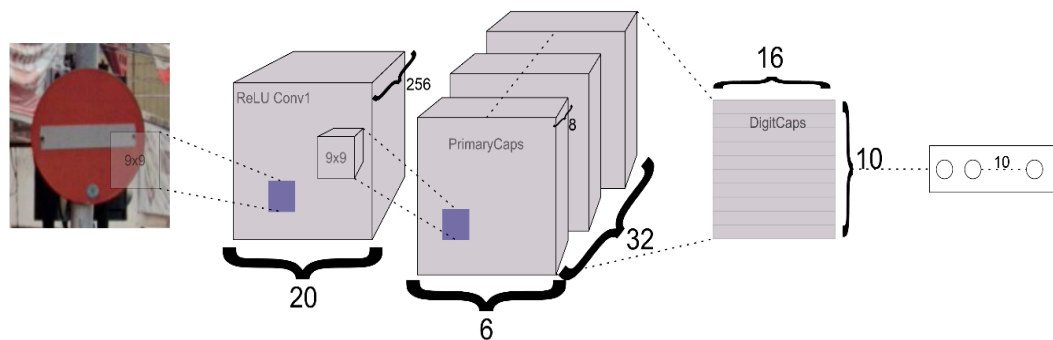


Figure 2. CapsNet architecture

The dataset used in this research is derived from a previous study shown in Table 1, which focused on recognizing Indonesian traffic signs using deep learning [16]. The dataset comprises 2050 color images (red-green-blue, RGB), categorized into 10 different types of traffic signs commonly found in Indonesia. These images were collected under varying lighting and background conditions to simulate real-world scenarios. The collected dataset initially consists of colored images, which are then processed by cropping to focus on the region of interest (ROI). The resulting cropped images have a uniform size of 128×128 pixels to maximize storage. The dataset is divided into two parts: 1750 images for training and 300 images for testing. Table I presents several sample images from the traffic sign dataset that were successfully gathered. Before being used, the dataset will go through preprocessing, which involves resizing to 28x28 and converting it into a grayscale image.

The capsule network architecture used in this study was presented in 2017 by Sabour and Hinton [18]. This network involves a traditional convolutional layer, a primary capsule layer, and a class capsule layer as shown in Figure 2 [30]. The convolutional layer serves to extract low-level features from the input image. This layer utilizes 256 filters with a kernel size of 9×9, a stride of 1, and employs the ReLU activation function. The output of this layer is a set of 2D feature maps that preserve the spatial structure of the input image, which are passed to the next layer to be organized

into capsules. The next layer is the primary capsule layer, the first layer in the network to output capsule vectors rather than scalar activations. This layer performs a convolutional operation but differs from the standard convolutional layer in that it produces multiple capsule channels at each spatial location. Specifically, this layer uses 32 capsule channels, where each capsule is an 8-dimensional vector. The convolution operation in this layer uses a kernel size of 9×9 and a stride of 2. As a result, this layer produces a grid of 6×6 capsules, with each capsule encoding different instantiation parameters such as pose, orientation, and deformation.

The final layer is the class capsule layer, also referred to as the digit capsule layer in earlier works [18]. This layer contains one capsule per target class. In this study, 10 capsules correspond to 10 traffic signs. Each class capsule outputs a 16-dimensional vector, where the length (norm) of the vector represents the probability of the input belonging to that class, and the direction encodes the class-specific instantiation parameters. The inputs to this layer are the outputs of the primary capsule layer, and a dynamic routing-by-agreement algorithm governs the connection between them. This algorithm iteratively updates the routing coefficients (weights) between the lower-level and higher-level capsules based on the agreement between predicted and actual capsule outputs. Typically, three routing iterations are used, which have been shown to provide a good trade-off between classification accuracy and computational cost. The model is trained using the margin loss function proposed by Sabour et al., which encourages the network to output capsule vectors with lengths close to one for the correct class and close to zero for the others. The model is optimized using the Adam optimizer with an initial learning rate of 0,001. The training is conducted over 20 epochs with a batch size of 2.

A confusion matrix is generated to assess classification performance by comparing the true and predicted labels across all 10 traffic sign classes. From this matrix, various performance metrics are computed, including overall accuracy, precision, recall, and F1-score, to evaluate the model's effectiveness comprehensively. A heatmap visualization of the confusion matrix is also used to highlight common misclassification patterns and provide insights into which traffic signs are most frequently confused by the model. In this study, two evaluations are conducted. The first test uses the original dataset as collected, providing a baseline performance measurement. The second test uses an augmented version of the original dataset, in which all images from the original dataset (training data and testing data) are artificially rotated by a random angle within a range of -5 to $+5$ degrees to simulate real-world variations in camera angle or road conditions. This second test aims to assess the model's robustness and ability to generalize under slight geometric transformations, which are common in practical traffic sign recognition scenarios. It is important to note that the dataset was first split into training and test sets, with rotation augmentation subsequently applied independently to each subset. Furthermore, the first and second tests were conducted entirely independently of one another, ensuring that the results of each evaluation reflect a distinct and unbiased measure of model performance. All test results were compared with models in previous studies that have proven to perform well with this dataset [17].

3. RESULTS AND DISCUSSION

3.1. Results

First, the CapsNet model will be trained using the original dataset over 20 epochs with a batch size of 2 and optimized using the Adam optimizer with an initial learning rate of 0,001.

The training results of the CapsNet model over 20 epochs, as shown in Table 2, demonstrate a significant reduction in loss and improvement in accuracy. Beginning with a relatively high loss of 0,8539 and a low accuracy of 11,31% in the first epoch, the model rapidly progressed through the training process. By the 5th epoch, accuracy had already surged to 95,26%, accompanied by a significant drop in loss to 0,0872. This upward trend continued, and by the 8th epoch, the model achieved an accuracy of over 99%. From epoch 12 onward, the model consistently maintained an outstanding accuracy of 99,89%, eventually reaching perfect classification accuracy (100%) starting from the 16th epoch through the 20th. During this final phase, the loss also remained minimal, fluctuating between 0,0047 and 0,0063, indicating strong model convergence and generalization.

Table 2. CapsNet's Training Results for Original and Augmented Datasets

Epoch	Original Dataset		Augmented Dataset	
	Loss	Accuracy (%)	Loss	Accuracy (%)
1	0,8539	11,31	0,8211	9,94
2	0,6532	13,03	0,7456	11,43
3	0,6101	13,49	0,6113	13,37
4	0,4008	47,09	0,4999	28,69
5	0,0872	95,26	0,1702	85,09
6	0,0479	97,49	0,0959	93,54
7	0,0338	98,29	0,0655	95,43
8	0,0243	99,26	0,0489	97,2
9	0,022	99,71	0,038	98,34
10	0,016	99,54	0,0286	99,14
11	0,013	99,89	0,0255	99,31
12	0,0101	99,89	0,0201	99,54
13	0,0081	99,83	0,0185	99,6
14	0,0084	99,94	0,0138	99,89
15	0,0063	99,94	0,0095	100
16	0,0078	100	0,0096	100
17	0,0047	100	0,0072	100
18	0,0028	100	0,0063	100
19	0,005	100	0,0082	99,94
20	0,0048	100	0,0056	100

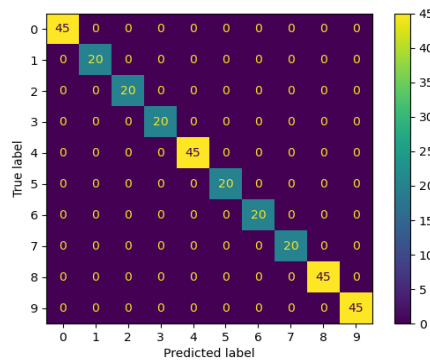


Figure 3. CapsNet's Confusion Matrix for Original Dataset

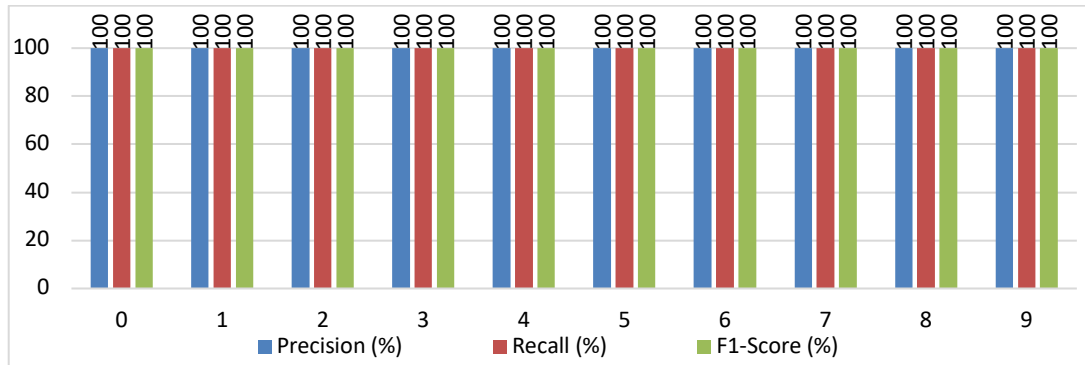


Figure 4. CapsNet's Classification Report for Original Dataset

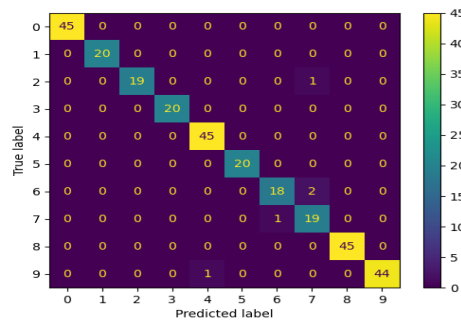


Figure 5. CapsNet's Confusion Matrix for Augmented/Affine-Transformed Dataset

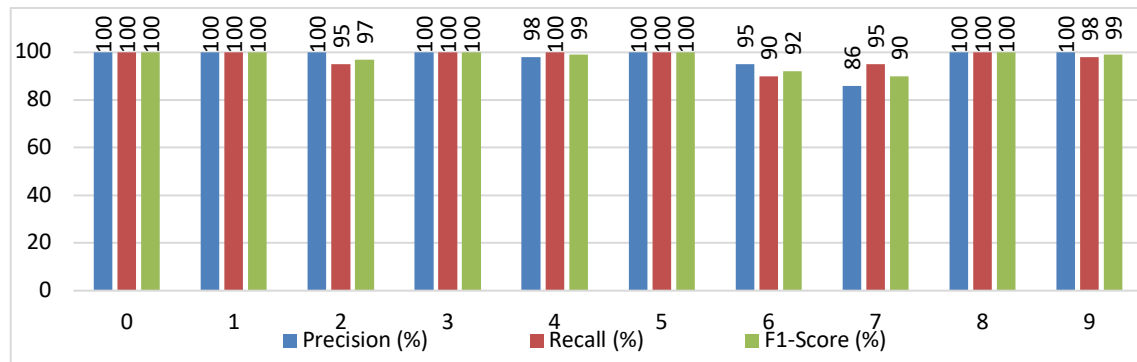


Figure 6. CapsNet's Classification Report for Augmented Dataset

The evaluation of the trained CapsNet model, as illustrated by the confusion matrix shown in Figure 3, reveals an exceptional level of performance across all classes. The confusion matrix displays a perfectly diagonal structure, indicating that every sample in the test set was correctly classified. Each class, whether it had 20 or 45 test samples, shows no instances of misclassification, resulting in zero off-diagonal values. This level of accuracy reflects the model's outstanding ability to distinguish between all ten classes with complete confidence. Further reinforcing the confusion matrix, the classification report provides detailed metrics, including precision, recall, and F1-score, with a score of 100% for each class. This means that for every label, the model made no false positives (precision), no false negatives (recall), and maintained perfect harmonic balance between the two (F1-score). The support values, ranging from 20 to 45 samples per class, demonstrate that this high performance was consistent even across classes with differing representations. The macro and weighted averages also recorded 100%, confirming that the model handled both majority and minority classes with equal precision and sensitivity.

For the second result, the CapsNet model will be trained using an augmented/affine transformed (rotation with a range of -5 to +5 degrees) dataset, employing the same parameters as in subsection 3.1. Table 2 also shows CapsNet model training results using the augmented dataset. In the initial epochs, performance started modestly, with the model achieving only 9,94% accuracy and a high loss of 0,8211 in the first epoch. However, by the 5th epoch, the accuracy surged to 85,09% and the loss dropped significantly to 0,1702, indicating that the model quickly adapted to the rotational variations introduced in the training data. This trend of improvement continued steadily, reaching 93,54% accuracy by epoch six and exceeding 97% by epoch 8. From epoch 10 onward, the model consistently achieved near-perfect performance, with accuracy values above 99% and continuously declining loss. Remarkably, starting from epoch 16, the model maintained a flawless 100% accuracy through to the 20th epoch, with the final loss value reaching a low of 0,0056. The confusion matrix of the CapsNet model trained with the augmented dataset, as shown in Figure 5, reveals a strong, though slightly imperfect, classification performance. Most classes were classified correctly with high accuracy, particularly classes 0, 1, 3, 4, 5, and 8, which were predicted flawlessly. However, a few minor misclassifications occurred: class 2 had one instance incorrectly predicted as class 9, class 6 had two samples misclassified as class 7, class 7 had one sample incorrectly predicted as class 3, and class 9 had one sample misclassified as class 4. These minor errors resulted in five misclassifications across all 300 test samples, thus the accuracy score reached 98,33%.

The classification report shown in Figure 6 provides a detailed evaluation of the CapsNet model trained using the augmented dataset with slight image rotations. The overall accuracy is 98%, with both macro and weighted averages of 98% across precision, recall, and F1-score, indicating strong and balanced model performance across all ten classes. Most classes, such as classes 0, 1, 3, 5, and 8, achieved perfect scores (100%) in all three metrics, reflecting the model's confidence and consistency in correctly identifying these classes. Slight declines in performance are observed in a few specific classes. For instance, class 2 had a recall of 95%, meaning one of the actual instances was misclassified, even though precision remained at 100%. Similarly, class 4 had a precision of 98% and a recall of 100%, suggesting one predicted sample from another class was incorrectly labeled as class 4. More significant drops are observed in classes 6 and 7: class 6

achieved a precision of 95% and a recall of 90%, while class 7 scored 86% precision and 95% recall, with corresponding F1-scores of 92% and 90%, respectively. These lower scores align with the confusion matrix results, where some samples were misclassified between these visually similar or rotationally ambiguous classes.

3.2. Discussions

First discussion is a comparison of the use of original and augmented datasets in the training process is presented in Table 2.

The original model's began with low accuracy (11,31% in epoch 1) and made a dramatic leap by epoch 5 (95,26%). In contrast, the model trained on the augmented dataset started at a slightly lower accuracy of 9,94% in epoch 1, and by epoch 5 reached 85,09%, still impressive but slightly behind the original version in raw percentage terms at that point. However, from epoch six onward, the performance of the augmented model caught up quickly, reaching 93,54% (compared to 97,49% in the original), and by epoch 8, it had achieved 97,2% accuracy, which still trails behind the original dataset's 99,26%. From epoch 10 onward, both models perform similarly, with over 99% accuracy and gradually decreasing loss. While both show a steady reduction in loss over time, the model trained with augmented data achieved a final loss of 0,0056 at epoch 20, compared to 0,0048 in the original dataset. Although the original model achieved a lower loss slightly earlier, the difference is marginal. However, the augmented model's loss reduction is smoother, with no sudden jumps, suggesting more stable training and better regularization due to the diversity introduced by slight image rotations. From a testing perspective, the difference becomes clear as shown in Figure 2 for the original dataset and Figure 3 for the augmented dataset. The original model achieved a perfect confusion matrix with all predictions aligning precisely with the true labels, indicating 100% accuracy and zero misclassifications. This suggests that the original model learned the training data very well and was capable of classifying static, unaltered images with complete precision. However, the model trained with augmented data faced more challenging conditions due to the slight rotations introduced during training, which better simulate real-world variations. While these rotations caused a few misclassifications, they also tested the model's robustness and generalization ability. These results highlight the trade-off between raw accuracy and real-world applicability. Although the model trained with rotated images did not achieve perfect accuracy, the small number of errors is acceptable given the added benefit of increased robustness to orientation changes. This aligns with CapsNet's design goal: to capture spatial relationships and part-whole hierarchies.

The second discussion is about a comparison of CapsNet results and previous research results using augmented Datasets.

Table 3. CapsNet and MS-CNN Training Results for Augmented Datasets

Epoch	CapsNet		MS-CNN	
	Loss	Accuracy (%)	Loss	Accuracy (%)
1	0,8211	9,94	1,7431	47,14
2	0,7456	11,43	0,6504	83,66
3	0,6113	13,37	0,332	91,43
4	0,4999	28,69	0,2167	94,06
5	0,1702	85,09	0,1543	95,49
6	0,0959	93,54	0,1157	97,2
7	0,0655	95,43	0,0893	97,94
8	0,0489	97,2	0,0666	98,86
9	0,038	98,34	0,0613	98,57
10	0,0286	99,14	0,043	99,43
11	0,0255	99,31	0,0351	99,66
12	0,0201	99,54	0,03	99,77
13	0,0185	99,6	0,0215	100
14	0,0138	99,89	0,0169	99,94
15	0,0095	100	0,0152	99,94
16	0,0096	100	0,0132	100
17	0,0072	100	0,011	100
18	0,0063	100	0,0096	100
19	0,0082	99,94	0,0089	100
20	0,0056	100	0,0078	100

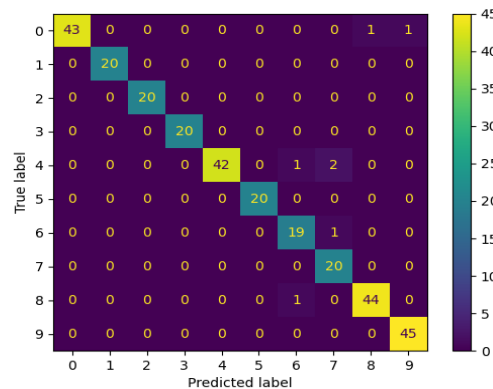


Figure 7. MS-CNNs' Confusion Matrix for Augmented Dataset

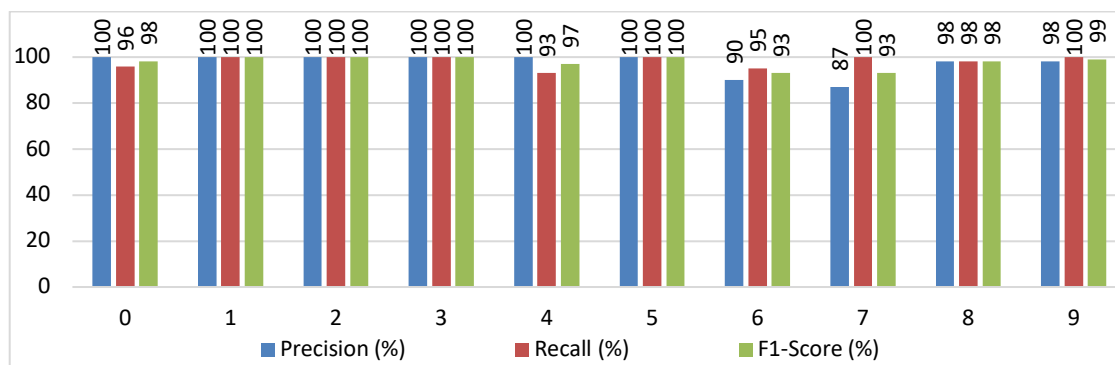


Figure 8. MS-CNNs' Classification Report for Augmented Dataset

Table 3 shows a comparison of the training results of CapsNet and MS-CNN. The MS-CNN model uses the same architecture as in previous study[17] (only produces 1 output instead of 2), utilizing the Adam optimizer and a learning rate of 0,001. In the early epochs, MS-CNN shows significantly faster learning progress. At epoch 1, it already reaches 47,14% accuracy with a loss of 1,7431, while CapsNet lags behind at 9,94% accuracy and a loss of 0,8211. By epoch 3, MS-CNN jumps to over 91% accuracy, indicating a rapid feature learning process, thanks to its multi-scale convolutional structure. In contrast, CapsNet achieves only 13,37% accuracy at the same point, indicating a slower start. However, by epoch 5, CapsNet begins to close the gap, reaching an accuracy of 85,09%, while MS-CNN remains at 95,49%. From epoch eight onward, both models show consistently high accuracy above 97%, with MS-CNN maintaining a slight lead in loss reduction. By epoch 12, both models achieve 100% accuracy and retain it through epoch 20. However, CapsNet's final loss is slightly lower at 0,0056, compared to MS-CNN's 0,0078, indicating a marginally better fit to the training data. MS-CNN learns faster and converges earlier, benefiting from its efficient multi-scale feature extraction. CapsNet, although slower in the early stages of training, catches up and performs equally well in terms of final accuracy, with a slightly lower final loss. This suggests that while MS-CNN learns faster and converges earlier, CapsNet generalizes better, particularly in tasks that require essential spatial relationships and transformation handling.

Figure 7 shows the confusion matrix which demonstrates strong performance, achieving an accuracy of 97,67% (7 misclassifications out of 300 images). The confusion matrix indicates that the model accurately classified most samples, with minor misclassifications occurring in a few classes. For example, class 0 had two samples misclassified (one each as class 8 and 9), and class 4 had three misclassified samples (one as class 5, one as class 6, and one as class 8). Similar minor errors are observed in classes 6 and 8, where one sample was incorrectly predicted as class 7, and one sample was misclassified as class 6. Despite these misclassifications, many classes, such as 1, 2, 3, 5, 7, and 9, were classified perfectly, showing the model's strong ability to generalize across most categories.

According to the classification report shown in Figure 8, most classes achieved perfect scores (100%) in precision, recall, and F1-score. However, a few classes exhibited lower performance, particularly class 0, which had a recall of 96%, and class 4, which had a slightly reduced precision and recall of 93% and 95%, respectively. Notably, class 7 had the lowest precision at 87%, reflecting some confusion with neighboring or visually similar classes. Nevertheless, the macro and weighted averages of 98% across all metrics indicate that the model performs well overall, with only minor deviations in a few edge cases. When compared to the CapsNet model trained on the same augmented dataset, which also achieved 98% accuracy, the results are similar in terms of overall performance. However, CapsNet had fewer misclassifications (5 errors total) than MS-CNN (7 errors total), and its classification report showed slightly more balanced precision and recall across all classes. For instance, the lowest F1-score for CapsNet was 90%, while MS-CNN's lowest dropped to 90% for class 7 and included slightly lower recall in other classes. This suggests that CapsNet demonstrates slightly better consistency and robustness in handling rotated inputs, likely due to its architectural design, which effectively captures spatial relationships and part-whole hierarchies.

4. CONCLUSION

This paper presents the Capsule Network (CapsNet) used in traffic sign recognition, specifically for Indonesian traffic signs. The dataset used in this study is taken from previous research conducted by Akbar et al., and the MS-CNN model will also be used for comparison. The CapsNet model is trained and tested using both the original and augmented datasets (rotated with a range of -5 to +5 degrees). Using the original dataset, the CapsNet model achieved a training loss of 0,0048 and an accuracy of 100% at the 20th epoch. From a testing perspective, the CapsNet model achieves 100% accuracy, indicating its ability to generalize from the original dataset. Using the augmented dataset, the CapsNet model achieved a training loss of 0,0056 and an accuracy of 100% at the 20th epoch. From a testing perspective, the CapsNet model achieves an accuracy of 98%. Compared to MS-CNN, CapsNet had fewer misclassifications (5 errors total) than MS-CNN (7 errors total), and its classification report showed slightly more balanced precision and recall across all classes. For instance, the lowest F1-score for CapsNet was 90%, while MS-CNN's lowest dropped to 90% for class 7 and included slightly lower recall in other classes. This suggests that CapsNet demonstrates slightly better consistency and robustness in handling augmented inputs, likely due to its architectural design, which captures spatial relationships and part-whole hierarchies more effectively.

CONFLICT OF INTEREST

The authors declare that they have no conflict of interest.

DATA AVAILABILITY

The data associated with this study are publicly available online in the replication package. [<https://doi.org/10.5281/zenodo.21194058>]

AUTHOR CONTRIBUTIONS

Mutaqin Akbar: Conceptualization; Data Curation; Data Labelling; Methodology; Writing-Original Draft; Visualization; Writing-Review & Editing. **Agus Sidiq Purnomo:** Conceptualization; Data Curation; Data Labelling; Methodology; Visualization; Writing-Review & Editing. **Supatman:** Conceptualization; Methodology; Supervision; Writing-Review & Editing. **Bernadete Deta.:** Methodology; Data Curation; Data Labelling; Writing-Review & Editing.

REFERENCES

- [1] K. Bengler, K. Dietmayer, B. Farber, M. Maurer, C. Stiller, and H. Winner, "Three Decades of Driver Assistance Systems: Review and Future Perspectives," *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 4, pp. 6–22, 2014, doi: 10.1109/MITS.2014.2336271.

- [2] T. Günthner and H. Proff, "On the way to autonomous driving: How age influences the acceptance of driver assistance systems," *Transp. Res. Part F Traffic Psychol. Behav.*, vol. 81, pp. 586–607, Aug. 2021, doi: 10.1016/j.trf.2021.07.006.
- [3] V. K. Kukkala, J. Tunnell, S. Pasricha, and T. Bradley, "Advanced Driver-Assistance Systems: A Path Toward Autonomous Vehicles," *IEEE Consum. Electron. Mag.*, vol. 7, no. 5, pp. 18–25, Sep. 2018, doi: 10.1109/MCE.2018.2828440.
- [4] A. Ziebinski, R. Cupek, D. Grzechca, and L. Chruszczyk, "Review of advanced driver assistance systems (ADAS)," presented at the PROCEEDINGS OF THE INTERNATIONAL CONFERENCE OF COMPUTATIONAL METHODS IN SCIENCES AND ENGINEERING 2017 (ICCMSE-2017), Thessaloniki, Greece, 2017, p. 120002. doi: 10.1063/1.5012394.
- [5] M. M. Narkhede and N. B. Chopade, "Review of Advanced Driver Assistance Systems and Their Applications for Collision Avoidance in Urban Driving Scenario," in *Machine Learning and Big Data Analytics (Proceedings of International Conference on Machine Learning and Big Data Analytics (ICMLBDA) 2021)*, vol. 256, R. Misra, R. K. Shyamasundar, A. Chaturvedi, and R. Omer, Eds., in Lecture Notes in Networks and Systems, vol. 256, Cham: Springer International Publishing, 2022, pp. 253–267. doi: 10.1007/978-3-030-82469-3_23.
- [6] J. Levinson *et al.*, "Towards fully autonomous driving: Systems and algorithms," in *2011 IEEE Intelligent Vehicles Symposium (IV)*, Baden-Baden, Germany: IEEE, Jun. 2011, pp. 163–168. doi: 10.1109/IVS.2011.5940562.
- [7] A. Suriya Prakash, D. Vigneshwaran, R. Seenivasaga Ayyalu, and S. Jayanthi Sree, "Traffic Sign Recognition using Deeplearning for Autonomous Driverless Vehicles," in *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*, Erode, India: IEEE, Apr. 2021, pp. 1569–1572. doi: 10.1109/ICCMC51019.2021.9418437.
- [8] Y. Zhu and W. Q. Yan, "Traffic sign recognition based on deep learning," *Multimed. Tools Appl.*, vol. 81, no. 13, pp. 17779–17791, May 2022, doi: 10.1007/s11042-022-12163-0.
- [9] X. R. Lim, C. P. Lee, K. M. Lim, T. S. Ong, A. Alqahtani, and M. Ali, "Recent Advances in Traffic Sign Recognition: Approaches and Datasets," *Sensors*, vol. 23, no. 10, p. 4674, May 2023, doi: 10.3390/s23104674.
- [10] A. Hamza and S. Nawal, "Traffic Sign Classification using Deep Learning Comparative Study," *Procedia Comput. Sci.*, vol. 233, pp. 939–949, 2024, doi: 10.1016/j.procs.2024.03.283.
- [11] G. Sapijaszko, T. Alobaidi, and W. B. Mikhael, "Traffic Sign Recognition Based on Multilayer Perceptron Using DWT and DCT," in *2019 IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS)*, Dallas, TX, USA: IEEE, Aug. 2019, pp. 440–443. doi: 10.1109/MWSCAS.2019.8884897.
- [12] A. Kerim and M. O. Efe, "Recognition of Traffic Signs with Artificial Neural Networks: A Novel Dataset and Algorithm," in *2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIIC)*, Jeju Island, Korea (South): IEEE, Apr. 2021, pp. 171–176. doi: 10.1109/ICAIIIC51459.2021.9415238.
- [13] W. Li, H. Song, and P. Wang, "Finely Crafted Features for Traffic Sign Recognition," *Int. J. Circuits Syst. Signal Process.*, vol. 16, pp. 159–170, Jan. 2022, doi: 10.46300/9106.2022.16.20.
- [14] B. Wang, "Research on the Optimal Machine Learning Classifier for Traffic Signs," *SHS Web Conf.*, vol. 144, p. 03014, 2022, doi: 10.1051/shsconf/202214403014.
- [15] Y. Le Cun *et al.*, "Handwritten Digit Recognition: Applications of Neural Net Chips and Automatic Learning," in *Neurocomputing*, F. F. Soulié and J. Hérault, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 1990, pp. 303–318. doi: 10.1007/978-3-642-76153-9_35.
- [16] M. Akbar, "Traffic sign recognition using convolutional neural networks," *J. Teknol. Dan Sist. Komput.*, vol. 9, no. 2, pp. 120–125, Apr. 2021, doi: 10.14710/jtsiskom.2021.13959.

- [17] M. Akbar, I. Susilawati, B. S. Jati, and N. Alamsyah, "Multi-Task Learning for Traffic Sign Recognition using Multi-Scale Convolutional Neural Networks," *Int. J. Adv. Data Inf. Syst.*, vol. 6, no. 2, pp. 391–402, Aug. 2025, doi: 10.59395/ijadis.v6i2.1406.
- [18] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic Routing Between Capsules," 2017, doi: 10.48550/ARXIV.1710.09829.
- [19] S. J. Pawan and J. Rajan, "Capsule networks for image classification: A review," *Neurocomputing*, vol. 509, pp. 102–120, Oct. 2022, doi: 10.1016/j.neucom.2022.08.073.
- [20] R. LaLonde, Z. Xu, I. Irmakci, S. Jain, and U. Bagci, "Capsules for Biomedical Image Segmentation," 2020, *arXiv*. doi: 10.48550/ARXIV.2004.04736.
- [21] S. Bonheur, D. Štern, C. Payer, M. Pienn, H. Olschewski, and M. Urschler, "Matwo-CapsNet: A Multi-label Semantic Segmentation Capsules Network," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2019*, vol. 11768, D. Shen, T. Liu, T. M. Peters, L. H. Staib, C. Essert, S. Zhou, P.-T. Yap, and A. Khan, Eds., in Lecture Notes in Computer Science, vol. 11768. , Cham: Springer International Publishing, 2019, pp. 664–672. doi: 10.1007/978-3-030-32254-0_74.
- [22] S. J. Pawan *et al.*, "Capsule Network-based architectures for the segmentation of sub-retinal serous fluid in optical coherence tomography images of central serous chorioretinopathy," *Med. Biol. Eng. Comput.*, vol. 59, no. 6, pp. 1245–1259, Jun. 2021, doi: 10.1007/s11517-021-02364-4.
- [23] T. Zeng, H. K.-H. So, and E. Y. Lam, "RedCap: residual encoder-decoder capsule network for holographic image reconstruction," *Opt. Express*, vol. 28, no. 4, p. 4876, Feb. 2020, doi: 10.1364/OE.383350.
- [24] A. Jaiswal, W. AbdAlmageed, Y. Wu, and P. Natarajan, "CapsuleGAN: Generative Adversarial Capsule Network," 2018, *arXiv*. doi: 10.48550/ARXIV.1802.06167.
- [25] K. Duarte, Y. S. Rawat, and M. Shah, "VideoCapsuleNet: A Simplified Network for Action Detection," 2018, *arXiv*. doi: 10.48550/ARXIV.1805.08162.
- [26] K. Duarte, Y. S. Rawat, and M. Shah, "CapsuleVOS: Semi-Supervised Video Object Segmentation Using Capsule Routing," 2019, *arXiv*. doi: 10.48550/ARXIV.1910.00132.
- [27] J. Bae and D.-S. Kim, "End-to-End Speech Command Recognition with Capsule Network," in *Interspeech*, 2018. [Online]. Available: <https://api.semanticscholar.org/CorpusID:52192557>
- [28] J. Poncelet, V. Renkens, and H. Van Hamme, "Low resource end-to-end spoken language understanding with capsule networks," *Comput. Speech Lang.*, vol. 66, p. 101142, Mar. 2021, doi: 10.1016/j.csl.2020.101142.
- [29] J. Kim, S. Jang, E. Park, and S. Choi, "Text classification using capsules," *Neurocomputing*, vol. 376, pp. 214–221, Feb. 2020, doi: 10.1016/j.neucom.2019.10.033.
- [30] M. U. Haq, M. A. J. Sethi, and A. U. Rehman, "Capsule Network with Its Limitation, Modification, and Applications—A Survey," *Mach. Learn. Knowl. Extr.*, vol. 5, no. 3, pp. 891–921, Aug. 2023, doi: 10.3390/make5030047.