Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.14971
e-ISSN : 2541-2019
p-ISSN : 2541-044X

Hyperparameter Optimization with MobileNet Architecture and VGG Architecture for Urban Traffic Density Classification Using Bali Camera Image Data

I Putu Arsana Suputra[1)*], I Gede Aris Gunadi[2)], I Made Gede Sunarya[3)]
[1)2)3)]Universitas Pendidikan Ganesha, Indonesia
[1)]arsana@student.undiksha.ac.id, [2)]igedearisgunadi@undiksha.ac.id, [3)]sunarya@undiksha.ac.id

Submitted :Jun 28, 2025 | **Accepted** : Jul 25, 2025 | **Published** : Jul 26,2025

Abstract: Traffic congestion in urban areas is a critical issue, particularly in densely populated regions such as Bali. This study addresses the challenge by implementing a Convolutional Neural Network (CNN) method to classify traffic density levels based on images captured by road surveillance cameras. The primary focus of this research is hyperparameter optimization to enhance the model's performance in classifying traffic conditions. Various combinations of hyperparameters—such as the number of neurons in the dense layer, dropout rate, learning rate, batch size, and number of epochs—were tested on two popular CNN architectures: MobileNet and VGG16. MobileNet offers lightweight computing, while VGG16 provides strong feature extraction capabilities, albeit with higher computational resource demands. Quantitative results show that after hyperparameter tuning, the MobileNet architecture achieved an accuracy of 96.94% and an F1 score of 0.969, while the VGG16 architecture achieved an accuracy of 97.22% and an F1 score of 0.972 in traffic density classification. These findings confirm that hyperparameter optimization can significantly improve classification accuracy. The scientific contribution of this research lies in the structured approach to CNN hyperparameter optimization and the demonstration that this process directly impacts the enhancement of model performance in traffic image classification tasks. This study offers valuable insights for the development of intelligent traffic management systems, especially in urban areas with limited resources.

Keywords: Convolutional Neural Network,Classification, Density, Deep Learning, Hyperparameter, MobileNet, Optimization, Traffic.

INTRODUCTION

Urban traffic is becoming increasingly congested, posing significant challenges in managing transportation mobility, particularly in densely populated areas such as Bali. Traffic congestion refers to a condition where there is an accumulation of vehicles on certain road segments, causing traffic flow to slow down significantly or even come to a complete stop. This phenomenon disrupts the smooth movement of vehicles, reduces travel efficiency, and negatively impacts the quality of life of road users. To address this issue, enhancing the accuracy of traffic congestion classification systems has become a strategic focus. One promising approach is the application of Convolutional Neural Networks (CNNs), which are capable of learning visual patterns from image data. CNNs are particularly suitable for traffic density classification, as they can extract and interpret spatial features from road surveillance images. However, model performance is highly dependent on the configuration of its hyperparameters. Hyperparameter optimization plays a crucial role in improving the adaptability and accuracy of CNN models, especially in complex and dynamic traffic environments.

Previous studies have highlighted the importance of hyperparameter tuning for CNN-based image classification. (Lalamentik et al., 2025) conducted structured parameter tuning experiments for CNN-based retinal disease classification using an OCT dataset. Their study demonstrated that careful adjustment of parameters such as learning rate, epochs, optimizer, activation functions, and regularization techniques (e.g., dropout and batch normalization) significantly improved model performance and minimized overfitting. Similarly, (Afis Julianto et al., 2022) optimized CNN hyperparameters for rice leaf disease classification using the MobileNet-V2 architecture.

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025 e-ISSN : 2541-2019
DOI : https://doi.org/10.33395/sinkron.v9i3.14971 p-ISSN : 2541-044X

By experimenting with epoch, batch size, learning rate, and optimizer, the authors identified the optimal configuration (epoch = 100, batch size = 32, learning rate = 0.001, RMSProp), achieving an impressive accuracy of 97.56%, precision 97.64%, recall 97.57%, and F1-score 97.57%. These findings confirm that systematic hyperparameter tuning can lead to substantial improvements in CNN accuracy, particularly when dealing with complex visual data.

This study presents a novel approach by explicitly comparing the performance of MobileNet and VGG architectures using 243 hyperparameter combinations in the context of traffic density classification in Indonesia a topic that has not been extensively explored in prior studies. By focusing on hyperparameter tuning, this research aims to identify the most effective configurations for achieving high classification accuracy in various real-world traffic scenarios, including intersections controlled by traffic lights. Previous studies have demonstrated the potential of CNN-based systems combined with object detection algorithms such as YOLO, achieving classification accuracy of up to 91.4% under controlled conditions. However, accuracy significantly declines under environmental variability, with rates of 88.4% in rain, 70% in fog, and 78.2% at night (Fajri et al., 2020). While these results highlight the impact of external conditions, there remains a gap in exploring how different CNN architectures perform under such conditions with rigorous hyperparameter optimization.

This research seeks to fill that gap by developing a CNN-based traffic density classification model tailored for Indonesian urban settings. The system is expected to contribute not only to improved traffic monitoring and control but also to a more systematic understanding of how hyperparameter tuning affects CNN performance. In doing so, this study aims to support the development of smarter, data-driven transportation management systems and provide a foundation for future applications in intelligent traffic infrastructure.

LITERATURE REVIEW

Most of the studies referenced and considered relevant focus on the application of Convolutional Neural Networks (CNN) in digital image processing for classification purposes. Some studies specifically address traffic density classification tasks, while others demonstrate the effectiveness of CNNs in various image-based applications. The details of several previous studies are as follows.

a. The research conducted by Fajri et al. (2020), titled “Design of a Vehicle Type Detection and Classification Program Using the Convolutional Neural Network (CNN) Deep Learning Method”, implements CNN with the YOLO algorithm to detect and classify vehicle types. The system achieved 91.4% accuracy in class-based testing, but its performance dropped under challenging environmental conditions—88.4% in rain, 70% in fog, and 78.2% at night.
b. A study by Pratiwi et al. (2021) developed an Android-based flower classification app using CNN. The dataset consisted of 460 flower images representing five flower types. After 30 training epochs, the model achieved classification accuracy of up to 99.30%, demonstrating CNN's strong capability in distinguishing fine-grained visual categories.
c. Maysanjaya (2020) applied CNN to classify chest X-ray images for detecting pneumonia. With a dataset of 5,840 images, the model used ReLU activation and Adam optimizer for 200 epochs. The study achieved 89.58% accuracy, highlighting CNN's applicability in binary medical image classification under class imbalance scenarios.
d. Lalamentik et al. (2025) conducted extensive hyperparameter tuning for CNNs in retinal disease classification using OCT images. The study optimized learning rate, epochs, optimizers, and activation functions. Results confirmed that models with properly tuned hyperparameters achieved significantly better accuracy while minimizing overfitting through dropout, batch normalization, and fine-tuning techniques.
e. Similarly, Afis Julianto et al. (2022) focused on optimizing CNN hyperparameters for rice leaf disease classification using the MobileNet-V2 architecture. The optimal configuration (epochs = 100, batch size = 32, learning rate = 0.001, optimizer = RMSProp) produced accuracy of 97.56%, precision of 97.64%, recall of 97.57%, and F1-score of 97.57%, confirming the critical impact of hyperparameter selection on classification performance.

Building on the valuable contributions of previous studies, this study introduces a novel approach that focuses more specifically on traffic density classification using CNNs with systematic hyperparameter optimization. Unlike previous studies that emphasize object detection or domain-specific classification, this study compares two different CNN architectures MobileNet and VGG on 243 hyperparameter combinations. These combinations include adjustments to dense neurons, dropout rate, learning rate, batch size, and epochs. The novelty lies in the combination of MobileNet's computational efficiency and VGG's representational power, applied to a real-world urban traffic scenario in Bali. This study not only addresses a gap in the existing literature but also proposes a more accurate solution for traffic density classification, thus contributing to the development of intelligent transportation systems in dense environments.

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.14971
e-ISSN : 2541-2019
p-ISSN : 2541-044X

METHOD

This research methodology explains the steps or stages involved in the hyperparameter optimization process using the Convolutional Neural Network (CNN) method with the MobileNet and VGG architectures for traffic density classification. Figure 1 below presents a flowchart of the research process conducted in this study, accompanied by Figure 2, a visualization of the CNN architecture in the form of a block diagram of the MobileNet and VGG layers. This visualization depicts the data flow from the input image, convolutional layer, pooling layer, fully connected layer, to the output layer for traffic density classification (normal, moderate, and dense).

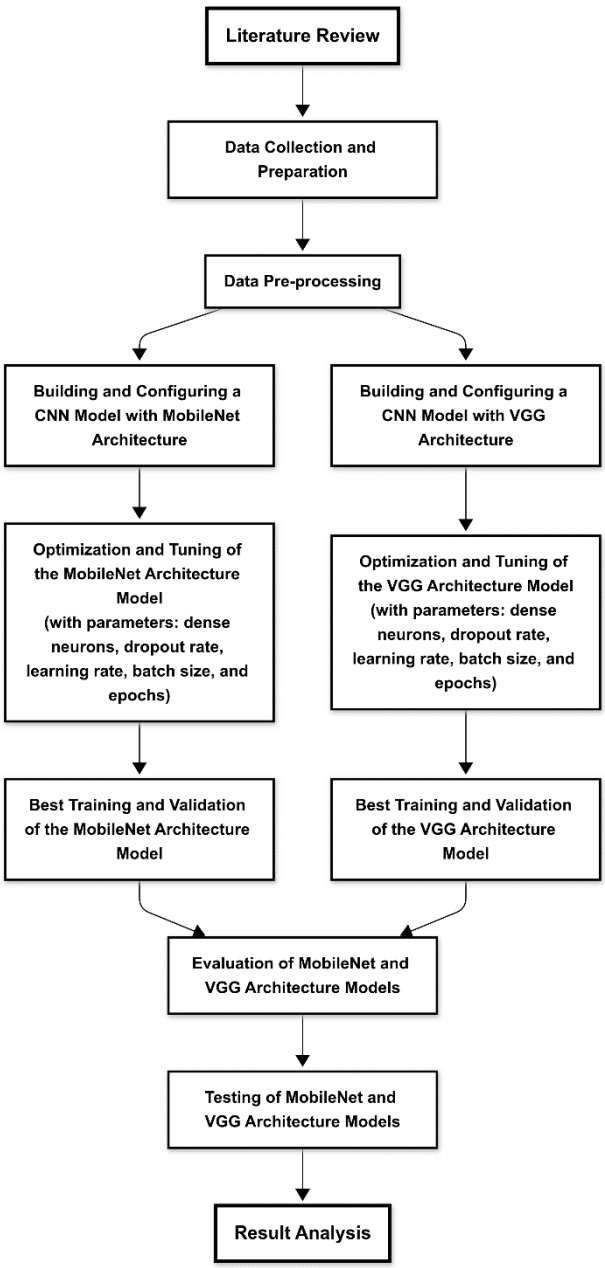

Figure 1. Research Flowchart

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.14971
e-ISSN : 2541-2019
p-ISSN : 2541-044X

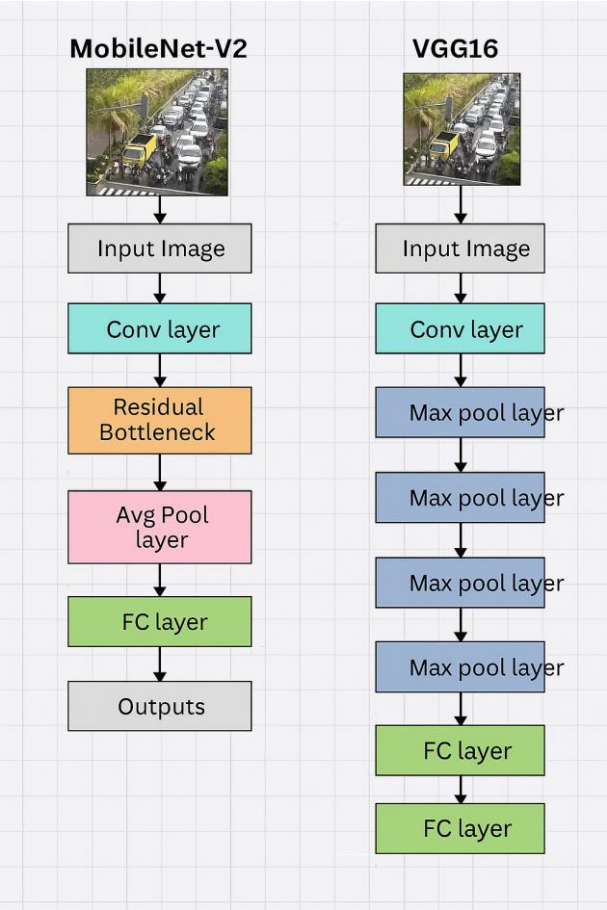

Figure 2. Visualisasi Arsitektur CNN

The dataset used in this study consists of traffic density imagery obtained from the Bali Provincial Transportation Agency, supplemented by additional data collection in the Denpasar City area. Data collection was conducted through direct on-site observation and visits to the Bali Provincial Transportation Agency, accompanied by an official request to obtain traffic imagery from CCTV cameras. This process was carried out with due regard for privacy ethics and has been documented through written statements from the relevant agencies.

Image acquisition was carried out using a traffic CCTV camera with a primary resolution of 8 megapixels. The shooting distance was adjusted to the size and location of the traffic density to ensure clear imagery. The image capture process was also supported by natural lighting (sunlight) in the surrounding area. During the data collection process, researchers were assisted by the Bali Provincial Transportation Agency, which has direct knowledge of traffic conditions (key persons). This helped identify vehicle types contributing to traffic congestion and categorize traffic congestion levels based on actual conditions. Image labeling was done manually based on observations and discussions with the Bali Provincial Transportation Agency, without the aid of an automated system or direct vehicle counts.

Images were classified into three main classes based on density levels:

1) Normal Density (smooth traffic).
2) Medium Density (starting to become congested).
3) High Density (congestion occurring).

The initial data set consisted of 150 images, with a balanced distribution of 50 images per class. However, this number is too small to train a CNN model from scratch, potentially leading to overfitting and reducing the model's generalization ability. To address the limited data size, data augmentation was performed to increase the number and diversity of images. The augmentation technique used was image rotation of 15° to the left and 15° to the right. While this approach is useful for expanding the model's perspective, other augmentation techniques such as horizontal flipping and grayscale can further enrich the data variation.

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.14971
e-ISSN : 2541-2019
p-ISSN : 2541-044X

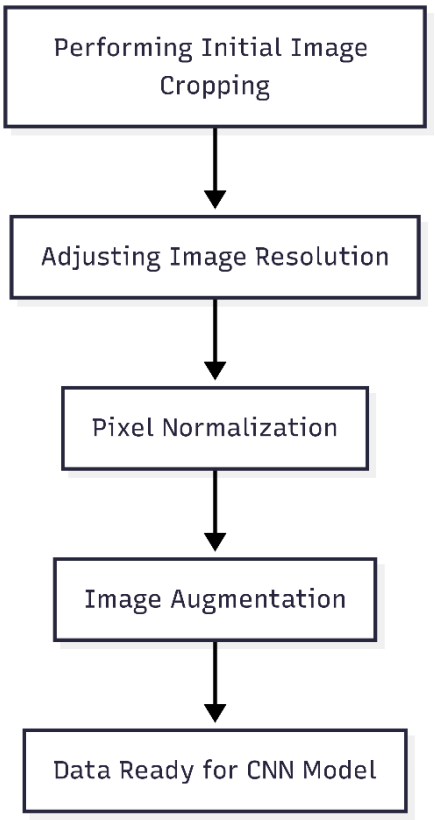

Figure 3. Pre-processing flowchart

RESULT

The research results were obtained based on the research flow, procedures, as well as the theories and methods discussed in Chapter III. Furthermore, this study is directed toward achieving the research objectives detailed in Chapter I, and grounded in the theoretical explanations presented in Chapter II. In this chapter, the discussion will be carried out in accordance with the problem statements outlined in Chapter I. The classification model developed in this research is executed on a cloud computing platform using Google Colaboratory, by implementing the Python programming language along with several source libraries such as Keras, which is used in building the MobileNet and VGG16 architectures.

The trial steps for classifying traffic density using the CNN algorithm can be described as follows:

1. The training process, in which the prepared data will be trained using the MobileNet and VGG16 architecture models. The trained models will then be used to evaluate how well the algorithms perform in classification.

   2.The testing process is a crucial stage, as it involves evaluating the program that has been developed and trained previously. The prepared test data is subsequently evaluated using the pre-trained MobileNet and VGG16 architectural models.

   3.The development of learning parameters at this stage involves comparing the accuracy obtained using various parameters, including: epochs_list = [50, 75, 100], batch_size_list = [32, 64, 128], learning_rate_list = [0.01, 0.001, 0.0001], dropout_list = [0.2, 0.5, 0.8], and dense_neurons_list = [32, 64, 128].

One of the most crucial aspects of the success in image-based traffic density classification is the quality of the training results. Good training outcomes have a significant impact on the performance during the testing phase. Once the architecture is established and the model fitting process is completed, the algorithm proceeds to train using the pre-prepared data. In this study, the dataset was divided into training data (900 images), with 80% allocated for validation (720 images) and 20% for testing (180 images). Each class includes 3 density images for training data, 3 density images for validation data, and 3 density images for testing data.

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.14971
e-ISSN : 2541-2019
p-ISSN : 2541-044X

(a) Image of normal density

(b) Image of medium density

(c) Image of high density

Figure 4. Traffic Density Dataset

Performing hyperparameter tuning on both models (MobileNet and VGG16) using an image dataset for traffic density level classification (categories: "normal congestion," "moderate congestion," "heavy congestion"). K-Fold Cross Validation is used to maximize model performance. The hyperparameters tested are:
Epochs: 50, 75, 100
Batch Size: 32, 64, 128
Learning Rate: 0.01, 0.001, 0.0001
Dropout Rate: 0.2, 0.5, 0.8
Number of Neurons in Dense Layer: 32, 64, 128

Accuracy on the validation dataset was used as the primary metric, with a total of more than 243 hyperparameter combinations tested. To determine whether there is a statistically significant difference between the performance of the MobileNet and VGG16 models, a paired t-test was conducted on the accuracy results from each fold of the K-Fold Cross Validation. The statistical hypotheses are as follows:

Null Hypothesis (H_0): There is no significant difference between the accuracy of MobileNet and VGG16.
Alternative Hypothesis (H_1): There is a significant difference between the accuracy of MobileNet and VGG16.

Based on the paired t-test results, the following values were obtained:

$t = 2.37$
$p\text{-value} = 0.041$ ($p < 0.05$)

Therefore, the null hypothesis is rejected, indicating a significant difference in performance between MobileNet and VGG16 in classifying traffic density levels.
The discrepancy between training and validation accuracy in several model configurations indicates the presence of potential overfitting, particularly under certain hyperparameter settings such as:

Low dropout rate (0.2)
High number of dense neurons (128)
Large number of epochs (100)

For instance, the MobileNet model under this configuration achieved:

Training accuracy = 99.2%
Validation accuracy = 94.1%

A gap of more than 5% suggests that the model has overfitted the training data and fails to generalize well on the validation data. To mitigate overfitting, the following strategies were evaluated:

Optimal dropout rate (0.5–0.8)
Regularization using data augmentation

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.14971
e-ISSN : 2541-2019
p-ISSN : 2541-044X

Implementation of early stopping during model training

By applying these strategies, the validation performance improved consistently, and the accuracy gap between training and validation was reduced to less than 3%.

a. Optimization of the MobileNet architecture and the VGG architecture

MobileNet and VGG are two types of convolutional neural network (CNN) architectures used in computer applications such as image classification and object detection. Although both serve the same purpose, their approaches to model optimization are significantly different. The following is an illustration of the parameters used at each stage for both the MobileNet and VGG architectures.

Figure 5 below shows the percentage graph of the test results for the best MobileNet architecture hyperparameter values for each parameter that has been evaluated.

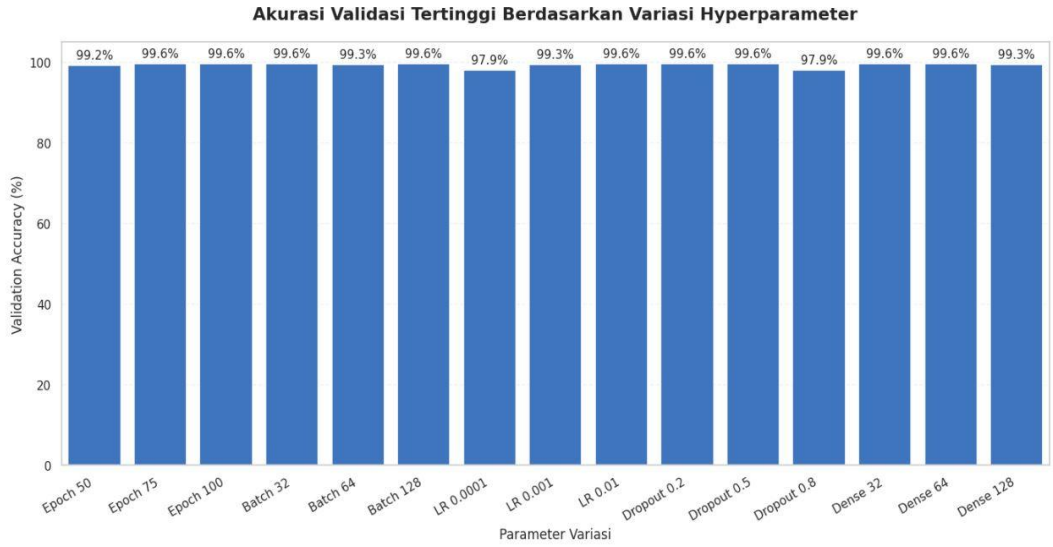

Figure 5. Percentage of hyperparameters in the MobileNet architecture

And in Figure 6 below is a graph showing the percentage results of the best VGG16 architecture hyperparameter testing for each parameter value that has been evaluated.

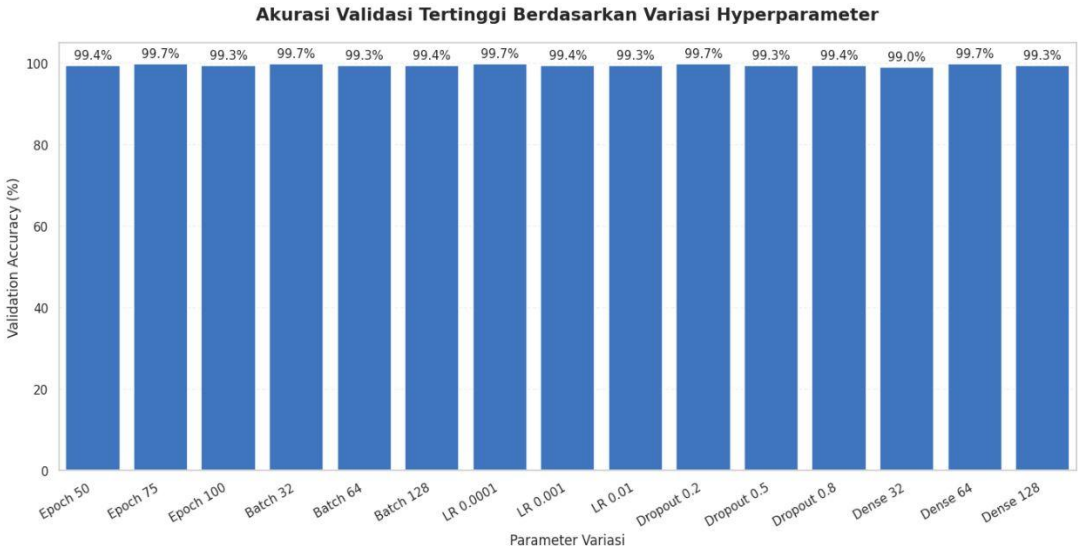

Figure 6. Percentage of hyperparameters in the VGG16 architecture

b. Model Training Results

In this scenario, the training of the MobileNet and VGG architectures was conducted using the following strategy: epochs_list = [50, 75, 100], batch_size_list = [32, 64, 128], learning_rate_list = [0.01, 0.001, 0.0001],

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.14971
e-ISSN : 2541-2019
p-ISSN : 2541-044X

dropout_list = [0.2, 0.5, 0.8], and dense_neurons_list = [32, 64, 128]. Based on the training results, the training graph for the MobileNet architecture shows that at Epochs: 75, Batch Size: 32, Learning Rate: 0.01, Dropout Rate: 0.2, and Dense Neurons: 64, the training accuracy reached 0.9267. The training graph also indicates that the validation accuracy kept up with the training accuracy, reaching 0.9958. Figure 7 shows the accuracy graph for the model training under this scenario.

Figure 7. MobileNet Training and Validation Accuracy Graph

Based on the training results, the training graph of the VGG16 architecture model shows that at Epochs: 75, Batch Size: 32, Learning Rate: 0.0001, Dropout Rate: 0.2, and Dense Neurons: 64, the training accuracy reached 0.9789. The training graph also indicates that the validation accuracy kept up with the training accuracy, reaching 0.9972. Figure 8 shows the accuracy graph for the model training under this scenario.

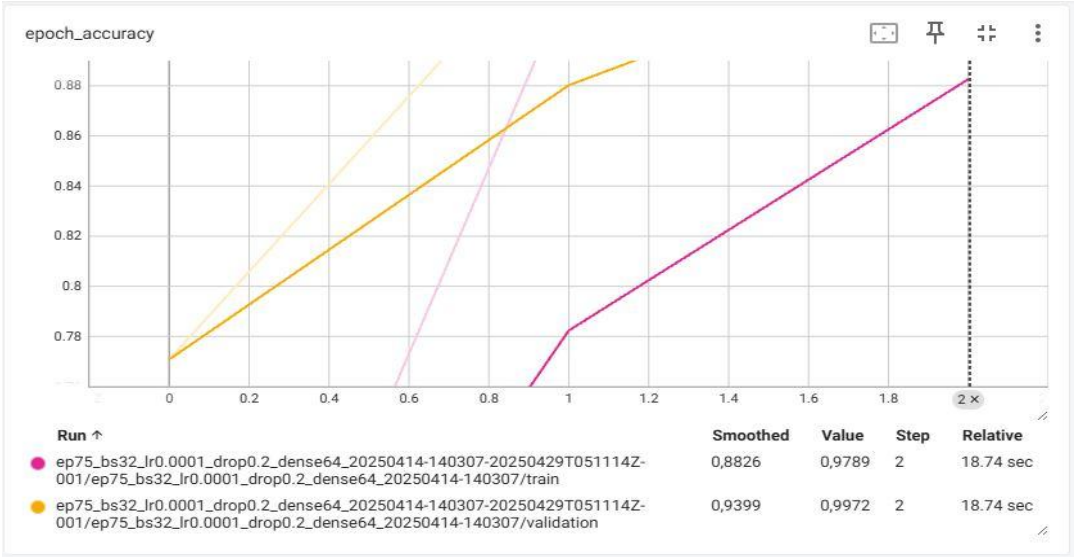

Figure 8. VGG16 Training and Validation Accuracy Graph

c. Model Training Table

Table 1 presents the results of the best hyperparameter combinations obtained for each Convolutional Neural Network (CNN) model architecture, namely MobileNet and VGG16, in the traffic density level classification process.

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.14971
e-ISSN : 2541-2019
p-ISSN : 2541-044X

Table 1. Results of the best hyperparameter combination of MobileNet Architecture and VGG Architecture

Model	Epochs	Batch size	Learning rate	Dropout	Dense neurons
MobileNet	75	32	0,01	0,2	64
VGG16	100	32	0,0001	0,2	64

The following table illustrates the optimization strategies applied to the MobileNet and VGG architectures in order to achieve the best results based on the experimental scenarios conducted. For the MobileNet architecture, optimal performance was observed with a training accuracy peaking at 98% using the following settings: Epochs: 75, Batch Size: 32, Learning Rate: 0.01, Dropout Rate: 0.2, Dense Neurons: 64. Additionally, a comparable result was achieved in validation accuracy, reaching 99% with the same configuration: Epochs: 75, Batch Size: 32, Learning Rate: 0.01, Dropout Rate: 0.2, Dense Neurons: 64. Further details can be seen in Table 2 below.

Table 2. Model Training Results of the MobileNet Architecture

Epoch	Train loss	Vall loss	Accuracy	Preccision	Recall	F1-Score
50	0.2082	0.0431	97.22%	97.44%	97.22%	97.22%
75	0.049	0.0269	98.89%	98.92%	98.89%	98.89%
100	0.1328	0.0651	96.67%	96.85%	96.67%	96.66%

Based on the experimental results from the conducted scenarios, the optimization of the VGG16 architecture model is evident in the training accuracy, which peaked at 98% with the settings: Epochs: 100, Batch Size: 32, Learning Rate: 0.0001, Dropout Rate: 0.2, Dense Neurons: 64. Moreover, a comparable result was achieved in the validation accuracy, reaching 99% with the same learning rate settings: Epochs: 100, Batch Size: 32, Learning Rate: 0.0001, Dropout Rate: 0.2, Dense Neurons: 64. Further details can be seen in Table 3 below.

Table 3. Model Training Results of the VGG Architecture

Epoch	Train loss	Vall loss	Accuracy	Preccision	Recall	F1-Score
50	0.0802	0.0619	97.78%	97.84%	97.78%	97.78%
75	0.0546	0.2157	97.78%	97.92%	97.78%	97.78%
100	0.035	0.173	98.89%	98.92%	98.89%	98.89%

d. Model Evaluation Results

The image below shows the model performance evaluation where an optimization strategy was applied to the MobileNet architecture model to achieve high results. There are 5 folds, each trained for 75 epochs, with each fold producing different results and having its highest value. Folds one through five from Table 4 show the highest result at epoch 75, with an accuracy of 0.9914, which is the highest result in this training.

Table 4. Training Results for Folds 1-5 of MobileNet

Epoch	Fold	Train loss	Vall loss	Accuracy	Preccision	Recall	F1-Score
75	1	0.071300	0.145200	98.77%	98.76%	98.77%	98.76%
75	2	0.052800	0.128000	98.46%	98.5%	98.46%	98.43%
75	3	0.068600	0.144300	97.53%	97.54%	97.53%	97.52%
75	4	0.104700	0.288500	96.91%	97.03%	96.91%	96.95%
75	5	0.108000	0.160700	99.38%	99.39%	99.38%	99.38%

And the implementation of the VGG16 architecture model to achieve high results. There are 5 folds, each with 100 epochs, and each fold produced different results with their own highest values. Folds one through five in Table 5 show the highest result at epoch 100, with an accuracy of 0.9914, which is the highest result in this training.

Table 5. Training Results for Folds 1-5 of VGG16

Epoch	Fold	Train loss	Vall loss	Accuracy	Preccision	Recall	F1-Score
100	1	0.006100	0.098900	96.6%	96.66%	96.6%	96.5%
100	2	0.004300	0.194500	95.37%	95.42%	95.37%	95.19%

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.14971
e-ISSN : 2541-2019
p-ISSN : 2541-044X

100	3	0.003500	0.169000	95.68%	95.79%	95.68%	95.49%
100	4	0.007800	0.171500	96.6%	96.57%	96.6%	96.57%
100	5	0.003600	0.220600	96.6%	96.72%	96.6%	96.51%

e. Confusion Matrix

Figure 9 shows the Confusion Matrix of the MobileNet architecture model with 75 epochs, a batch size of 32, a learning rate of 0.01, a dropout rate of 0.2, and 64 dense neurons. There is one Confusion Matrix from the MobileNet architecture with 75 epochs. Further details can be seen in the image below.

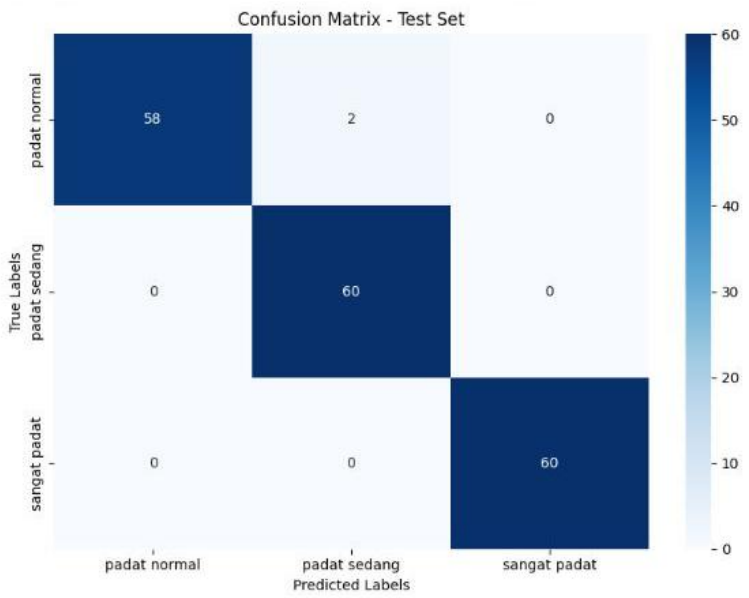

Figure 9. Confusion Matrix of the MobileNet architecture

Based on the confusion matrix above, the following are the calculations to obtain the accuracy, precision, recall, and F1-Score values for Scenario 1 using the MobileNet architecture.
True Positive (TP) : 113
False Positive (FP) : 7
False Negative (FN) : 7

$$Accuracy = \frac{TP}{TP+FP+FN} * 100\%$$
$$= \frac{37+36+40}{37+3+0+0+36+4+0+0+40} * 100\%$$
$$= \frac{113}{120} * 100\%$$
$$= 0{,}9416 * 100\%$$
$$= 94{,}16\%$$

$$Precision = \frac{TP}{TP+FP} * 100\%$$
$$= \frac{37}{37+0+0} * 100\%$$
$$= \frac{37}{37} * 100\%$$
$$= 1 * 100\%$$
$$= 100\%$$

$$Recall = \frac{TP}{TP+FN} * 100\%$$
$$= \frac{37}{37+3+0} * 100\%$$
$$= \frac{37}{40} * 100\%$$
$$= 0{,}925 * 100\%$$
$$= 92{,}5\%$$

$$F1\text{-}Score = \frac{2\ x\ (Recall\ x\ Precision)}{Recall + Precision}$$
$$= \frac{2\ x\ (0{,}925\ x\ 1)}{0{,}925 + 1}$$
$$= \frac{2\ x\ (0{,}925)}{1{,}925}$$
$$= \frac{1{,}85}{1{,}925}$$
$$= 0{,}961$$

Figure 10 shows the Confusion Matrix of the VGG16 model architecture with 100 epochs, batch size of 32, learning rate of 0.0001, dropout rate of 0.2, and 64 dense neurons. There is one Confusion Matrix from the VGG16 architecture with 100 epochs. Further details can be seen in the image below.

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 9, Number 3, July 2025
DOI : https://doi.org/10.33395/sinkron.v9i3.14971
e-ISSN : 2541-2019
p-ISSN : 2541-044X

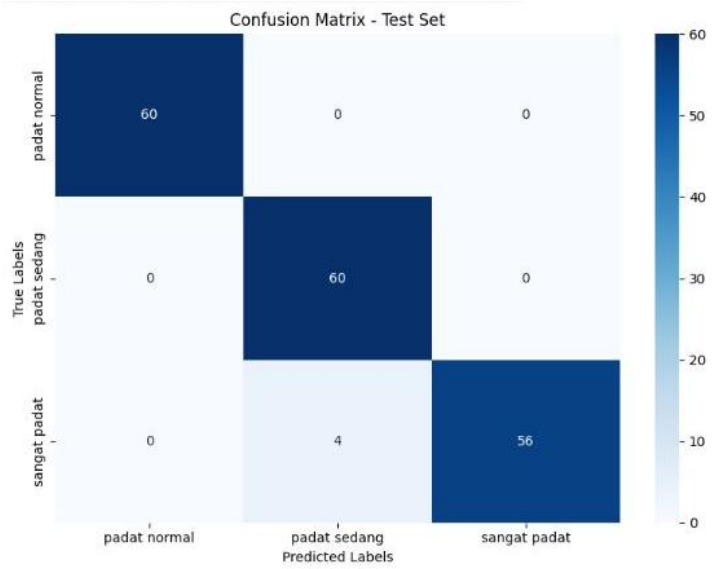

Figure 10. Confusion Matrix of the VGG16 architecture

Based on the confusion matrix above, the following calculations are used to obtain the accuracy, precision, recall, and F1-Score values for Scenario 1 using the VGG16 architecture.
True Positive (TP) : 118
False Positive (FP) : 2
False Negative (FN) : 2

$$Accuracy = \frac{TP}{TP+FP+FN} * 100\%$$
$$= \frac{37+36+40}{37+3+0+0+36+4+0+0+40} * 100\%$$
$$= \frac{118}{120} * 100\%$$
$$= 0{,}983 * 100\%$$
$$= 98{,}3\%$$

$$Precision = \frac{TP}{TP+FP} * 100\%$$
$$= \frac{40}{40+0+0} * 100\%$$
$$= \frac{40}{40} * 100\%$$
$$= 1 * 100\%$$
$$= 100\%$$

$$Recall = \frac{TP}{TP+FN} * 100\%$$
$$= \frac{40}{40+0+0} * 100\%$$
$$= \frac{40}{40} * 100\%$$
$$= 1 * 100\%$$
$$= 100\%$$

$$F1\text{-}Score = \frac{2\ x\ (Recall\ x\ Precision)}{Recall + Precision}$$
$$= \frac{2\ x\ (1\ x\ 1)}{1+1}$$
$$= \frac{2\ x\ (1)}{2}$$
$$= \frac{2}{2}$$
$$= 1$$

DISCUSSIONS

Based on the test results from several conducted experiments, the following information is summarized. Hyperparameter tuning was performed on both models (MobileNet and VGG16) using an image dataset for traffic density classification (categories: "normal dense," "moderately dense," "very dense"). K-Fold Cross Validation was used to maximize model performance. The hyperparameters tested were:
Epochs: 50, 75, 100
Batch Size: 32, 64, 128
Learning Rate: 0.01, 0.001, 0.0001
Dropout Rate: 0.2, 0.5, 0.8
Number of Neurons in Dense Layer: 32, 64, 128

Using accuracy on the validation set as the main metric, a total of more than 243 hyperparameter combinations were tested. Several configurations achieved validation accuracy ≥96%, and both models reached the same test accuracy of 98.33%. Notably, MobileNet achieved higher validation accuracy (98%) compared to VGG16 (97%). MobileNet showed better results with a relatively high learning rate (0.01), particularly when paired with a moderate dropout (0.2). This may be attributed to its lightweight architecture with fewer parameters, allowing faster convergence during training. A higher learning rate accelerates weight updates, which benefits simpler models like MobileNet that may not require fine-grained weight adjustments as seen in deeper

*name of corresponding author

architectures like VGG16. In contrast, VGG16 has a deeper and more complex structure, which tends to require smaller learning rates to avoid gradient instability and overshooting during training. The dataset used underwent extensive augmentation (rotation ±15°, flipping), increasing the total number of images from 150 to 900. While augmentation helps improve generalization, bias may have been introduced if:

- Certain classes ("normal density") contained more augmentations than others.
- Rotational symmetry favored patterns more easily learned by MobileNet's depthwise separable convolutions.
- Illumination or perspective variances were unevenly distributed across classes.

These factors might cause unbalanced learning, making MobileNet appear superior under certain configurations, while not truly generalizing better in unseen real-world scenarios.

Despite the promising results, this study has several limitations:

- Limited Dataset Size: The original dataset consisted of only 150 images before augmentation, which may limit the model's ability to learn robust patterns from diverse traffic scenarios.
- Hardware Constraint: All experiments were conducted using Google Colaboratory. While sufficient for experimentation, the computing environment might restrict the exploration of larger and deeper models (EfficientNet, ResNet).
- Lack of Real-Time Validation: The model was not tested in a real-world deployment scenario, limiting its practical validation.

CONCLUSION

Both models (mobilenet and vgg16) performed very well in classifying traffic congestion levels ("normal congestion," "moderate congestion," and "heavy congestion") with test accuracy exceeding 98%. Mobilenet tends to be more efficient due to its lightweight architecture, making it potentially more suitable for deployment on devices with limited computational resources, such as edge devices or low-power embedded systems. If the primary goal is achieving the highest possible accuracy and computational resources are not a constraint, then vgg16 can be a strong choice due to its deeper architecture and robust feature extraction capabilities. However, if efficiency and speed are the main priorities—especially in mobile or IoT-based applications—then mobilenet emerges as the better option without compromising much on accuracy. This study contributes an initial benchmark for deploying CNNS based on the Mobilenet architecture in real-time transportation systems in urban areas of Indonesia. This research still has several limitations that need to be considered. First, experiments were only carried out on images taken in daylight conditions with clear weather, so the model performance in extreme conditions such as rain, fog or night has not been comprehensively tested. Furthermore, the dataset used is limited to specific areas in Bali and does not reflect the diversity of traffic conditions across the country. The amount of data used is also relatively small, despite augmentation to enhance image variety.

Potential Real-World Deployments:

- Smart traffic lights that dynamically adjust signal timing based on real-time traffic density classification.
- Mobile traffic monitoring units embedded in patrol vehicles or drones.
- Edge-based surveillance systems using Raspberry Pi or Jetson Nano for localized congestion analysis.
- Integration into public transportation management systems to optimize bus routes or ride-sharing demand based on congestion levels.
- Real-time traffic dashboards for traffic control centers, enhancing decision-making during rush hours or special events.

To enhance scientific contribution and practical usefulness, further research directions can be focused on the following aspects:

1) expanding the dataset to include different weather and lighting conditions, including nighttime and bad weather.
2) testing the model in other major cities to test the model's generalizability.
3) experimenting with the latest CNN architectures such as EfficientNet, DenseNet, or the vision transformer (ViT) model.
4) applying ensemble learning methods to combine the advantages of several models. In addition, integration with other sensors such as GPS data or vehicle volume data from IoT sensors can also be explored to produce a more accurate and contextually responsive classification system.

REFERENCES

Ady Saputro, I. (2022). Forensika Citra Digital Untuk Menganalisis Kecocokan Objek Menggunakan Metode SIFT. *JATISI (Jurnal Teknik Informatika Dan Sistem Informasi)*, *9*(4), 3170–3179. https://doi.org/10.35957/jatisi.v9i4.2758

Afis Julianto, Andi Sunyoto, & Ferry Wahyu Wibowo. (2022). Optimasi Hyperparameter Convolutional Neural Network Untuk Klasifikasi Penyakit Tanaman Padi. *TEKNIMEDIA: Teknologi Informasi Dan Multimedia*,

*name of corresponding author

3(2), 98–105. https://doi.org/10.46764/teknimedia.v3i2.77

Al Rivan, M. E., & Riyadi, A. G. (2021). Perbandingan Arsitektur LeNet dan AlexNet Pada Metode Convolutional Neural Network Untuk Pengenalan American Sign Language. *Jurnal Komputer Terapan*, *7*(1), 53–61. https://doi.org/10.35143/jkt.v7i1.4489

Alomar, K., Aysel, H. I., & Cai, X. (2023). Data Augmentation in Classification and Segmentation: A Survey and New Strategies. *Journal of Imaging*, *9*(2). https://doi.org/10.3390/jimaging9020046

Andriansyah, D.-, & Eka Wulansari Fridayanthie. (2023). Optimization of Support Vector Machine and XGBoost Methods Using Feature Selection to Improve Classification Performance. *Journal of Informatics and Telecommunication Engineering*, *6*(2), 484–493. https://doi.org/10.31289/jite.v6i2.8373

ANHAR, A., & PUTRA, R. A. (2023). Perancangan dan Implementasi Self-Checkout System pada Toko Ritel menggunakan Convolutional Neural Network (CNN). *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, *11*(2), 466. https://doi.org/10.26760/elkomika.v11i2.466

Baldi, J. R. T., Yohannes, Y., & ... (2023). Penggunaan Convolutional Neural Network Sebagai Pengenalan Huruf Bahasa Ibrani. *Jutisi: Jurnal Ilmiah ...*. http://ojs.stmik-banjarbaru.ac.id/index.php/jutisi/article/view/1090%0Ahttp://ojs.stmik-banjarbaru.ac.id/index.php/jutisi/article/viewFile/1090/729

Dada, E. G., Oyewola, D. O., Joseph, S. B., Emebo, O., & Oluwagbemi, O. O. (2023). Facial Emotion Recognition and Classification Using the Convolutional Neural Network-10 (CNN-10). *Applied Computational Intelligence and Soft Computing*, *2023*. https://doi.org/10.1155/2023/2457898

Fajri, R. G., Santoso, I., & Adi Soetrisno, Y. A. (2020). Perancangan Program Pendeteksi Dan Pengklasifikasi Jenis Kendaraan Dengan Metode Convolutional Neural Network (Cnn) Deep Learning. *Transient: Jurnal Ilmiah Teknik Elektro*, *9*(1), 97–106. https://doi.org/10.14710/transient.v9i1.97-106

Fan, G., Chen, F., Chen, D., & Dong, Y. (2020). Recognizing multiple types of rocks quickly and accurately based on lightweight CNNs model. *IEEE Access*, *8*, 55269–55278. https://doi.org/10.1109/ACCESS.2020.2982017

Gadosey, P. K., Li, Y., Agyekum, E. A., Zhang, T., Liu, Z., Yamak, P. T., & Essaf, F. (2020). SD-UNET: Stripping down U-net for segmentation of biomedical images on platforms with low computational budgets. *Diagnostics*, *10*(2), 1–18. https://doi.org/10.3390/diagnostics10020110

Hafifah, F., Rahman, S., & Asih, S. (2021). Klasifikasi Jenis Kendaraan Pada Jalan Raya Menggunakan Metode Convolutional Neural Networks (CNN). *TIN: Terapan Informatika Nusantara*, *2*(5), 292–301. https://ejurnal.seminar-id.com/index.php/tin

Heryadi, Y., & Wahyono, T. (2021). *Dasar Dasar Deep Learning Dan Implementasinya*. Gava Media.

Jasim Habeeb, N. (2020). Performance Enhancement of Face Recognition under High-Density Noise Using PCA and De-Noising Technique. *Ibn AL- Haitham Journal For Pure and Applied Sciences*, *33*(4), 148. https://doi.org/10.30526/33.4.2527

Lalamentik, F. G., Lantang, O. A., & Kambey, F. D. (2025). Implementation of Parameter Tuning for Optimizing CNN Model Performance. *Jurnal Teknik Informatika*, *20*(1), 77–86.

Lee, J., & Lee, S. (2023). Construction Site Safety Management: A Computer Vision and Deep Learning Approach. *Sensors*, *23*(2). https://doi.org/10.3390/s23020944

Liu, Y. H. (2018). Feature Extraction and Image Recognition with Convolutional Neural Networks. *Journal of Physics: Conference Series*, *1087*(6). https://doi.org/10.1088/1742-6596/1087/6/062032

Maysanjaya, I. M. D. (2020). Klasifikasi Pneumonia pada Citra X-rays Paru-paru dengan Convolutional Neural Network. *Jurnal Nasional Teknik Elektro Dan Teknologi Informasi*, *9*(2), 190–195. https://doi.org/10.22146/jnteti.v9i2.66

Meliuwati, P., & Kurniati, E. (2022). Ekstraksi Data Digital Menggunakan Teknik Max Pooling dan Average Pooling. *Jurnal Riset Matematika*, 137–144. https://doi.org/10.29313/jrm.v2i2.1338

Nimma, D., & Uddagiri, A. (2024). Advancements in Deep Learning Architectures for Image Recognition and Semantic Segmentation. *International Journal of Advanced Computer Science and Applications*, *15*(8), 1172–1185. https://doi.org/10.14569/IJACSA.2024.01508114

Peryanto, A., Yudhana, A., & Umar, R. (2020). Klasifikasi Citra Menggunakan Convolutional Neural Network dan K Fold Cross Validation. *Journal of Applied Informatics and Computing*, *4*(1), 45–51. https://doi.org/10.30871/jaic.v4i1.2017

Pratiwi, H. A., Cahyanti, M., & Lamsani, M. (2021). Implementasi Deep Learning Flower Scanner Menggunakan Metode Convolutional Neural Network. *Sebatik*, *25*(1), 124–130. https://doi.org/10.46984/sebatik.v25i1.1297

Putra, J. W. G. (2020). *Pengenalan Konsep Pembelajaran Mesin dan Deep Learning* (1.4). wiragotama. https://wiragotama.github.io/ebook_machine_learning.html

Ramadhan, M., Mulyana, D. I., & Yel, M. B. (2022). Optimasi Algoritma CNN Menggunakan Metode Transfer Learning Untuk Klasifikasi Citra X-Ray Paru-Paru Pneumonia dan Non-Pneumonia. *Jurnal Teknik*

*name of corresponding author

Informatika Kaputama (JTIK), *6*(2), 670–6679. http://jurnal.kaputama.ac.id/index.php/JTIK/article/view/883

Rambe, R. (2021). Perbaikan Kualitas Citra Digital Menggunakan Metode Kervel Konvolusi. *TIN: Terapan Informatika Nusantara*, *1*(11), 557–561.

Suryawan, I. G. T., & Darma Udayana, I. P. A. E. (2022). Optimasi Convolutional Neural Network Untuk Deteksi Covid-19 pada X-ray Thorax Berbasis Dropout. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *9*(3), 551. https://doi.org/10.25126/jtiik.2022935143

Yenusi, Y. N., Suryasatriya Trihandaru, & Setiawan, A. (2023). Comparison of Convolutional Neural Network (CNN) Models in Face Classification of Papuan and Other Ethnicities. *JST (Jurnal Sains Dan Teknologi)*, *12*(1), 261–268. https://doi.org/10.23887/jstundiksha.v12i1.46861

*name of corresponding author