



Comparison of ResNet50V2 and InceptionV3 with Adam, SGD, and RMSprop Optimizers for Road Image Classification

Rifka Anrahvi^{1*}, Stevani², Syahid Muhammad Hibban³

^{1,2}Department of Information System, Faculty of Science and Technology,
Universitas Islam Negeri Sultan Syarif Kasim Riau, Indonesia,

³International Relations Study Programme, Faculty of Administration and Economics,
Bandırma Onyedi Eylül Üniversitesi, Türkiye

E-Mail: ¹12250321100@students.uin-suska.ac.id,

²12250324245@students.uin-suska.ac.id, ³syahidhibban@ogr.bandirma.edu.tr

Received Jul 11th 2025; Revised Aug 25th 2025; Accepted Aug 28th 2025; Available Online Aug 31th 2025

Corresponding Author: Rifka Anrahvi

Copyright © 2025 by Authors, Published by Institut Riset dan Publikasi Indonesia (IRPI)

Abstract

This study compares two Convolutional Neural Network (CNN) architectures ResNet50V2 and InceptionV3 with three optimizers (Adam, RMSprop, and SGD) for road condition classification. Using a dataset of 1,000 images categorized into four classes, the models were evaluated based on accuracy, precision, recall, and F1-score. Based on the results, ResNet50V2 with Adam optimizer performed the best, achieving 99% accuracy, whereas SGD yielded less-than-ideal results. This study is interesting since it compares architecture-optimizer pairings, a topic that hasn't been extensively studied in other studies. The results provide valuable insights for developing automated and reliable road monitoring systems that support efficient infrastructure maintenance. To further enhance performance, future study might entail implementing regularization techniques and growing the dataset.

Keywords: Adam Optimizer, CNN, Image Classification, ResNet50V2, Road Condition Classification

1. INTRODUCTION

A pothole is a concave part of the road surface that cracks and eventually forms a round hole with a diameter of more than 15 cm [1]. Damage not only jeopardizes the safety of road users but is also difficult to detect quickly and thoroughly using manual methods. In addition, potholes on the road surface can indicate that the road has not received optimal maintenance and can signal the potential for deeper structural problems [2]. This is where Artificial Intelligence (AI) and techniques such as Deep Learning (DL) become important, as they can identify damage patterns in road images automatically and efficiently [3].

As deep learning technology develops, the Convolutional Neural Network (CNN) algorithm becomes one of the most effective methods, as it can automatically extract important features from images without needing a complex manual feature extraction process [4]. One widely used CNN architecture is Inception V3, which has been trained on millions of common images. Through a transfer learning approach, the knowledge learned by Inception V3 can be reused to recognize road defect patterns, such as potholes, by only retraining the final part of the model. This approach not only saves training time but also improves efficiency and accuracy in the road defect image classification process [5].

Previous studies in other domains have also confirmed the robustness of this architecture. For example, Rinanda et al [6] reported that InceptionV3 achieved 96.50% accuracy in classifying mango leaf diseases, while VGG16 slightly outperformed it with 96.87%. In the medical field, ResNet50V2 achieved 94%-96% accuracy in pneumonia detection from chest X-ray images and performed even better when combined with GAN [7]. Similarly, a study on weather image classification found InceptionV3 to be the most consistent model, outperforming DenseNet169 and NASNetMobile with accuracy above 93% on the test dataset. These results show that ResNet50V2 and InceptionV3 are reliable CNN architectures across a wide range of image classification tasks, making them promising candidates for pothole detection as well.

Beyond the road defect domain, both InceptionV3 and ResNet50V2 have shown outstanding performance in other fields. For example, InceptionV3 achieved 96.50% accuracy in plant disease detection and 93.81% in weather image classification, while ResNet50V2 achieved up to 96% accuracy in medical

image analysis. These consistent results demonstrate its potential effectiveness in road surface defect detection [8].

In addition to Inception V3, other CNN architectures are also used to detect road damage, one of which is ResNet50, which is known for its deep residual structure and efficiency in extracting image features, in the research of Saisree, et al (2018) ResNet50 produces 91.94% accuracy [9]. The Stochastic Gradient Descent (SGD) optimizer is used as the initial optimization method. However, SGD has limitations that can cause research results to be unstable or less than optimal. In the study, researchers replaced SGD with the Adam optimizer (Adaptive Moment Estimation). The results show that Adam's application is effective in pothole detection [10]. Besides Adam and SGD, other optimizers such as RMSprop have also been used in pothole detection model research. RMSprop is known to be effective in handling complex visual data. In a study by Parasnis et al (2023), RMSprop successfully improved the model's accuracy to more than 96% by maintaining training stability and avoiding overfitting [11].

This research aims to develop and evaluate an automatic pothole detection system using a deep learning method based on a CNN architecture [12]. Specifically, this research focuses on utilizing transfer learning models with Inception V3 and ResNet50 architectures to classify damaged road images [13]. In addition, this study also aims to compare the effectiveness of several optimizers, such as SGD, Adam and RMSprop in improving model accuracy and stability in the training process [14], and using the holdout validation method with a data sharing ratio of 80:20 from a total of 1000 road image datasets. The results of this research are expected to produce an accurate and reliable pothole detection model so that it can help in monitoring and repairing roads more efficiently.

This study is very important as it contributes to the development of a more efficient and automated road monitoring system. Unlike previous studies that focused on a single CNN architecture or relied on a single optimizer, this study provides a comprehensive comparative analysis across multiple configurations. Therefore, the findings offer valuable insights and can serve as a practical reference for researchers and practitioners in developing road defect detection systems.

2. MATERIAL AND METHOD

The purpose of the elements in the diagram is to compile and organize data, to improve model accuracy and performance, and to minimize bias and error in the analysis of results. Research methodology can be seen in Figure 1.

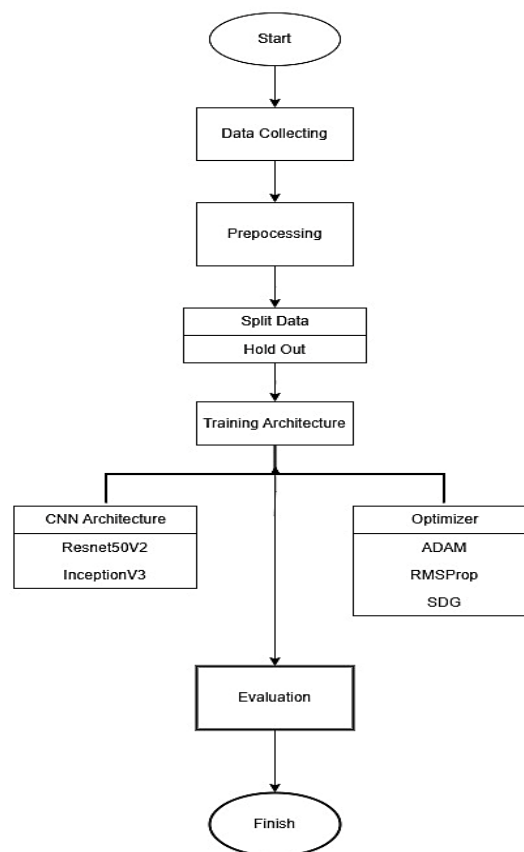


Figure 1. Research Methodology

2.1. Data Collecting

As training data, this study uses the Road Damage Classification and Assessment image dataset obtained through Kaggle. This dataset originally consisted of about 2,000 images categorized into four road condition classes: good, satisfactory, poor, and Very Poor. Only 1,000 images were used in this study, drawn proportionally from these four classes. Although this dataset has not been extensively used in prior academic research, its applicability and accessibility for assessing CNN-based picture classification tasks are what led to its selection. Compared to well-established benchmarks such as the dataset by Maeda et al [15] with over 9,000 annotated images and the RDD2022 dataset by Arya et al [16] with more than 47,000 images across multiple countries, the Kaggle dataset is smaller in scale but remains representative for proof-of-concept experiments. This subset was also chosen to balance computational efficiency and accuracy testing, enabling a focused analysis of CNN architectures and optimizers in classifying road damage conditions

2.2. Data Preprocessing

The preprocessing process to make the data ready for use in analysis with deep learning models begins after the data is collected. In this step, the image size is uniform, pixels are normalized by dividing them by 255, and the experimental data is enhanced by random rotation up to 45 degrees, vertical and horizontal inversion, and filling in empty pixels with the closest value. For now, the tested data is only validated through the normalization and division processes. Preprocessing ends with determining the batch size to control the number of image samples processed each time training is performed.

2.3. Deep Learning

Deep Learning (DL) is a branch of Machine Learning that utilizes layered neural networks to recognize complex patterns from data [17]. DL can automatically classify, predict, and detect [18]. Deep Learning models such as CNN, LSTM and Transformer have been widely used in the financial industry for credit prediction, fraud detection, and risk analysis [19]. However, challenges such as Interpretability, scalability, and efficiency still need to be overcome [20].

2.4. Convolutional Neural Network (CNN)

CNN are an evolution of the Multilayer Perceptron (MLP) made to handle two-dimensional data [21]. CNN was first developed under the name NeoCognitron by Kunihiko Fukushima, a researcher from NHK Broadcasting Science Research Laboratories, Kinuta, Setagaya, Tokyo, Japan. NeoCognitron became the forerunner of modern CNNs used today. CNN is very effective in understanding image structure because it automatically integrates features from small pixels into a more complex shape [22]. CNNs can identify objects within an image after receiving input. Image recognition and classification processes often use this algorithm [23]. CNN is considered one of the best models for solving object recognition problems in image data because it is able to recognize visual features [24]. Layer CNN can be seen in Figure 2.

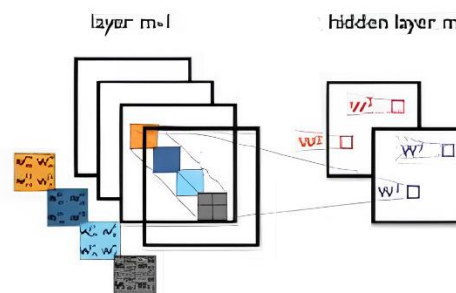


Figure 2. CNN

2.5. Arsitektur ResNet50

The ResNet-50 architecture model is one of the architecture models that has many layers, namely up to 50 layers in the hope of producing good accuracy [25]. ResNet50 is one of the architectures of CNN that introduces a new concept, namely shortcut connections [26]. Research from [27] found that the use of ResNet-50 obtained an accuracy rate of 96%. Arsitektur ResNet50 can be seen in Figure 3.

2.6. Arsitektur InceptionV3

Inception V3 is an extension of GoogLeNet (InceptionV1), which was later refined into InceptionV2 through the addition of batch normalization. The next iteration, InceptionV3, added factorization techniques for computational efficiency. The architecture is designed with dimensionality reduction at every stage, making it faster in processing, memory efficient, and still delivering high performance in image classification, even with limited resources [28]. Another advantage of Inception V3 in research [29] is its

ability to perform transfer learning, which allows the model to utilize knowledge from large datasets that have been previously trained so that the training process on new datasets becomes more efficient and accurate [30] found that the use of Inception V3 got an accuracy value of 92%. Figure 4 is an Arsitektur Inception V3.

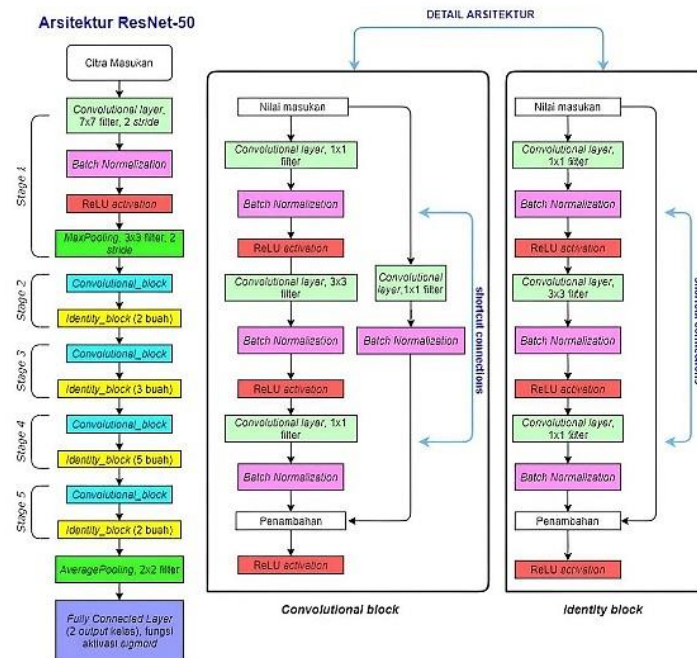


Figure 3. Arsitektur ResNet50

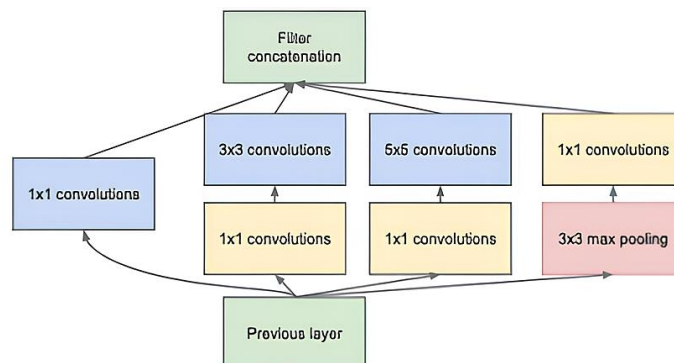


Figure 4. Arsitektur Inception V3

2.7. Optimizer Stochastic Gradient Descent (SGD)

SGD is a simple and fast algorithm, but its performance is suboptimal on complex datasets and deep networks. Although variants such as momentum and Nesterov help, its performance still lags behind adaptive algorithms such as Adam and AdaMax. SGD requires tuning the learning rate and is less stable in training large networks [31]. To determine how long the next step should take when moving forward in the direction of the gradient, SGD includes a parameter called the learning rate. Several contributions exist in specific areas of research on choosing the right learning rate. In general, a learning rate that is too low leads to slow convergence, while a learning rate that is too high can hinder convergence and cause the loss function to fluctuate around a minimum value or even diverge [32].

$$x_t \leftarrow x_{t-1} - \eta \nabla f_i(x_{t-1}) \quad (1)$$

The equation is the parameter update formula in SGD. This equation states that the parameter value x_t at iteration t is obtained by subtracting the previous value $x_{(t-1)}$ from the product of the learning rate (η) and the gradient of the loss function f_i concerning $x_{(t-1)}$. The goal is to minimize the loss function by moving in the direction of the negative derivative (fastest decreasing direction).

2.8. Optimizer Adaptive Moment Estimation (Adam)

Adam uses an adaptive learning rate approach, which means calculating one's learning rate according to various conditions. It preserves the historical gradient while maintaining the average exponential decay of the previous squared gradient, such as momentum [33]. Adam Optimizer is proven to be an efficient and effective optimization method in training ANFIS models. Its main advantage lies in its ability to adaptively adjust the learning rate for each parameter so that the training process is faster, more stable, and able to achieve optimal convergence [34]. It is suitable for training complex models such as CNNs, Long Short Term Memory networks (LSTMs), and more recently, large language models such as BERT as it is very effective in handling problems with large scale and large parameter space [35].

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (2)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \quad (3)$$

The equation is part of Adam's algorithm, which calculates the exponential average of the gradient m_t and the square of the gradient v_t to adaptively adjust the learning rate during the model training process.

2.9. Optimizer RMSprop

RMSprop is known to be effective in handling unstable gradient fluctuations, by adjusting the learning rate based on the exponential moving average of the previous gradient. Therefore, this algorithm is suitable for data that has a high rate of change or dynamics [36]. Since it uses exponentially weighted gradient averaging, RMSprop is similar to gradient descent with momentum. However, the only difference is that the parameters must be updated [37]. By selecting different learning rates for each parameter, the learning rates are automatically aligned [33]. The RMSprop optimizer in the CNN model gives much higher accuracy than the Adam optimizer. This model can be further applied to identify Telugu handwritten characters more accurately [33].

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} \cdot g_t \quad (4)$$

The equation is part of the RMSprop algorithm used to optimize model parameters in deep learning training. RMSprop adjusts the learning rate based on the mean square of the previous gradient. By dividing the gradient by the root of the expected value plus ϵ , RMSprop helps maintain the stability of parameter updates, especially in unstable loss functions.

3. RESULTS AND ANALYSIS

In this stage, the results of the road condition image classification process using CNN are analyzed. The previous process began with data collection, preprocessing, and the use of the hold-out method to separate the data. Next, CNN models were built using two different architectures, ResNet50V2 and InceptionV3, and optimized with three types of optimizers, ADAM, RMSProp, and SGD. An evaluation is conducted to evaluate the performance of the combination of architecture and optimizer in classifying road condition images.

3.1. Data Set

After obtaining the road condition dataset, the Python programming process randomly displays the data by showing five images for each class. This dataset consists of four classes: good, poor, satisfactory, and very poor, as shown in Figure 5.

3.2. Pre-Processing

In this study, the ImageDataGenerator module of Keras was used to perform data preprocessing. Images were resized to 224 x 224 pixels and normalized to a scale of 1/255. Training data was added to increase data variety and prevent overfitting. These enhancements include random rotation of up to 30 degrees, zoom of up to 15 percent, horizontal and vertical shift of 20 percent, shear of up to 15 percent, and horizontal and vertical flip. Training, validation, and testing data were loaded from a directory organized by class name using the `flow_from_directory` method, which uses the category classification mode. The training data was only augmented, while the validation and testing data were only normalized. Augmentation visualization was also used to display the various transformed shapes of a single initial photo.

3.3. Data Augmentation

Data augmentation techniques are used to increase and expand the amount of training data by altering the original data without changing its name or meaning. In image processing, this technique is very helpful in

overcoming the limited amount of data and improving the generalization ability of the model. An original image and its nine augmented results are shown in Figure 6. Rotation, flipping, zooming, changing the brightness level (captioning), and spatial distortions such as shearing and warping are some of the augmentation techniques used. With the variations resulting from these augmentation processes, machine learning models can better adapt to various image conditions in the real world. This results in improved model performance and a reduction in the possibility of overfitting.

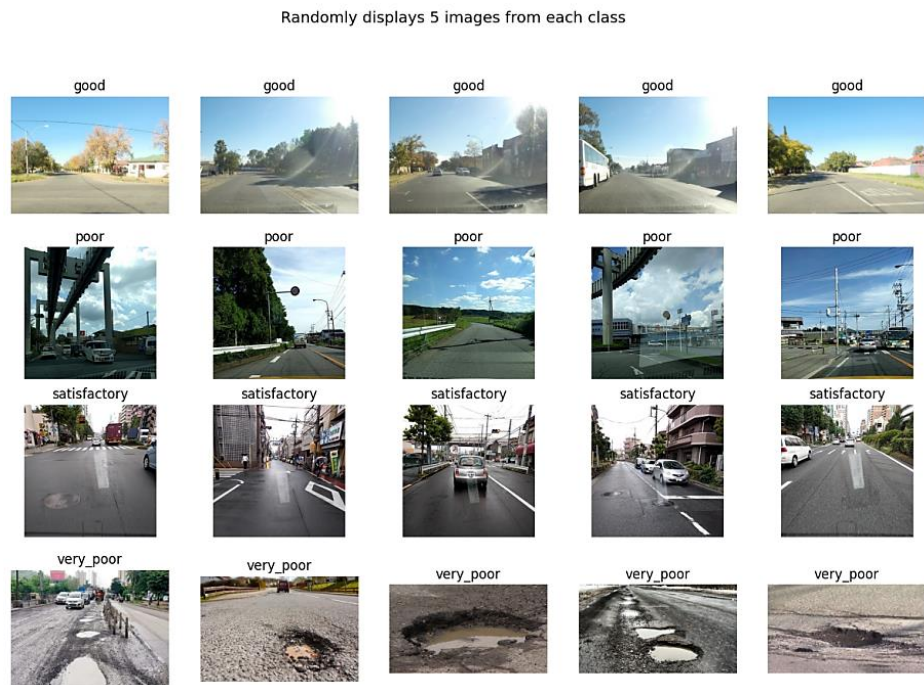


Figure 5. Road Dataset



Figure 6. Data Augmentation

3.4. Hasil Pengujian Confusion Matrix

3.4.1. Confusion Matrix ResNet50V2 & SGD

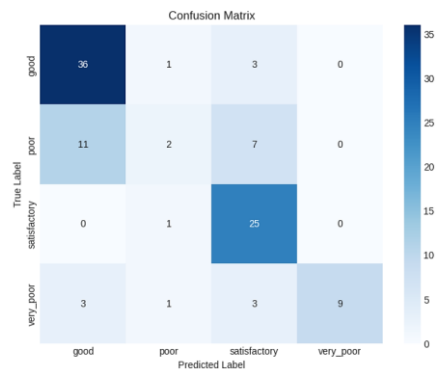


Figure 7. Confusion Matrix ResNet50V2 & SGD

3.4.2. Confusion Matrix ResNet50V2 & RMSprop

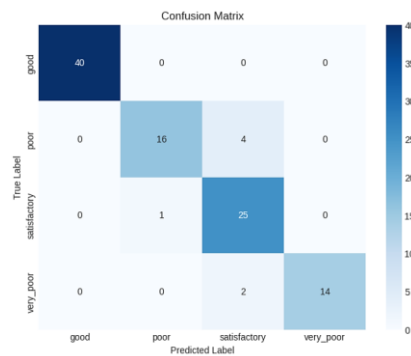


Figure 8. Confusion Matrix ResNet50V2 & RMSprop

The model in Figure 8 is able to classify the good class perfectly (40 correct data) and quite well in the satisfactory class (25 correct data). The poor and very poor classes still have some prediction errors, but overall, the model performs quite well and better than before.

3.4.3. Confusion Matrix ResNet50V2 & Adam

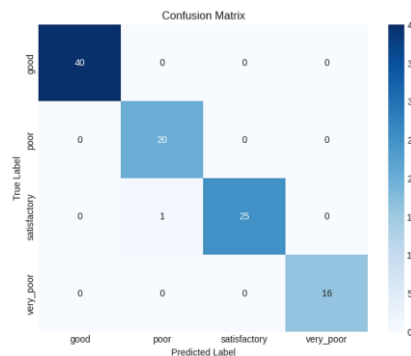


Figure 9. Confusion Matrix ResNet50V2 & Adam

The Confusion matrix in Figure 9 above shows excellent classification results, all data from the good, poor, and very poor classes are correctly classified, while the satisfactory class only has one error. This shows that the model is very accurate.

3.4.4. Confusion Matrix Inception V3 & RMSprop

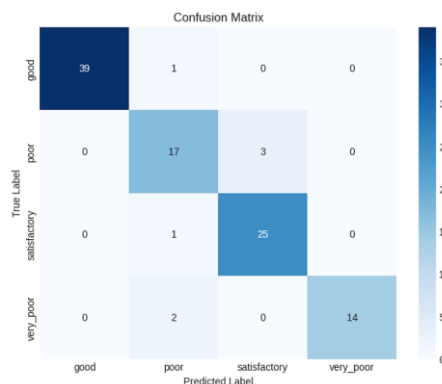


Figure 10. Confusion Matrix Inception V3 & RMSprop

The model's good output is shown by the confusion matrix in Figure 10. Almost all classes are classified accurately, especially the good and satisfactory classes. The poor and very poor classes have some errors, but overall the results are satisfactory.

3.4.5. Confusion Matrix InceptionV3 & Adam

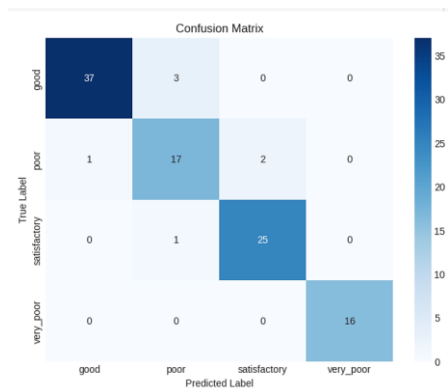


Figure 11. Confusion Matrix InceptionV3 & Adam

The confusion matrix in Figure 11 shows the model's good performance. The majority of predictions are on target, especially in the good, satisfactory, and very poor classes. Only a few errors occurred in the poor and good classes.

3.4.6. Confusion Matrix Inception V3 & SGD

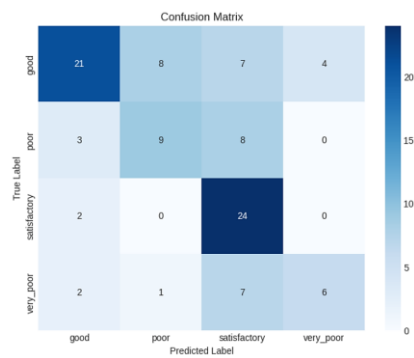


Figure 12. Confusion Matrix Inception V3 & SGD

Figure 12 shows the performance of the classification model in predicting the four classes of good, poor, satisfactory, and very poor. The true labels are shown in each row, and the labels predicted by the model are shown in the columns. For example, out of 30 actual data considered “satisfactory”, 24 were predicted to be successful correctly, while the others were misclassified to other classes. The darker the color of the diagonal (from left to right) the more accurate the prediction. Colors that fall outside the diagonal line indicate misclassification.

3.5. Performance Comparison

Table 1 shows that the combination of architecture and optimizer has a significant effect on classification performance. In general, the use of adaptive optimizers such as Adam and RMSprop produces much better performance than SGD. This can be seen in the combination of ResNet50V2 + Adam, which is able to achieve near-perfect accuracy and F1-score values in all classes, with only one prediction error in the satisfactory class. Conversely, SGD shows the lowest performance, especially in the minority classes (poor and very poor), with a much lower F1-score. Adam's superior performance can be explained by its mechanism, which uses an adaptive learning rate for each parameter and combines momentum with the estimated mean squared gradient. This approach makes Adam more stable and converges faster even on unbalanced datasets. RMSprop also shows quite good results because this algorithm is able to maintain the stability of weight updates by taking into account the average gradient square, thereby reducing fluctuations during training. Meanwhile, SGD has limitations because it uses a fixed learning rate, which makes the training process more difficult to achieve optimal convergence, and is prone to local minima on complex datasets. These findings confirm that the choice of optimizer greatly affects classification results. Adaptive optimizers are more suitable for classifying road images with unbalanced class distributions, as they are able to adjust the learning rate to the characteristics of the data, thereby improving model stability and accuracy.

Table 1. Performance Comparison

Architecture	Optimizer	Class Name	Precision	Recall	F1-Score	Support
Resnet50V2	SGD	Good	0.72	0.90	0.80	40
		Satisfactory	0.66	0.96	0.78	26
		Poor	0.40	0.10	0.16	20
		Very Poor	1.00	0.56	0.72	16
Resnet50V2	RMSProp	Good	1.00	1.00	1.00	40
		Satisfactory	0.81	0.96	0.88	26
		Poor	0.94	0.80	0.86	20
		Very Poor	1.00	0.88	0.93	16
Resnet50V2	Adam	Good	1.00	1.00	1.00	40
		Satisfactory	1.00	0.96	0.98	26
		Poor	0.95	1.00	0.98	20
		Very Poor	1.00	1.00	1.00	16
InceptionV3	RMSProp	Good	1.00	0.97	0.99	40
		Satisfactory	0.89	0.96	0.93	26
		Poor	0.81	0.85	0.83	20
		Very Poor	1.00	0.88	0.93	16
InceptionV3	Adam	Good	0.97	0.93	0.95	40
		Satisfactory	0.93	0.96	0.94	26
		Poor	0.81	0.85	0.83	20
		Very Poor	1.00	1.00	1.00	16
InceptionV3	SGD	Good	0.75	0.53	0.62	40
		Satisfactory	0.52	0.92	0.67	26
		Poor	0.50	0.45	0.47	20
		Very Poor	0.60	0.38	0.46	16

3.6. Evaluasi

3.6.1. ResNet50V2 & SGD

Figure 13, Training Accuracy illustrates how the model accuracy has evolved over 30 epochs of training and validation data. In general, both training and validation accuracies increase consistently from around 35% to over 70%. The validation accuracy line (red) tends to be slightly higher than the training accuracy (blue), indicating that the model does not suffer from overfitting and is able to generalize well to data that has not been seen before. This trend indicates that the training process went smoothly and that the model was able to use the available data successfully.

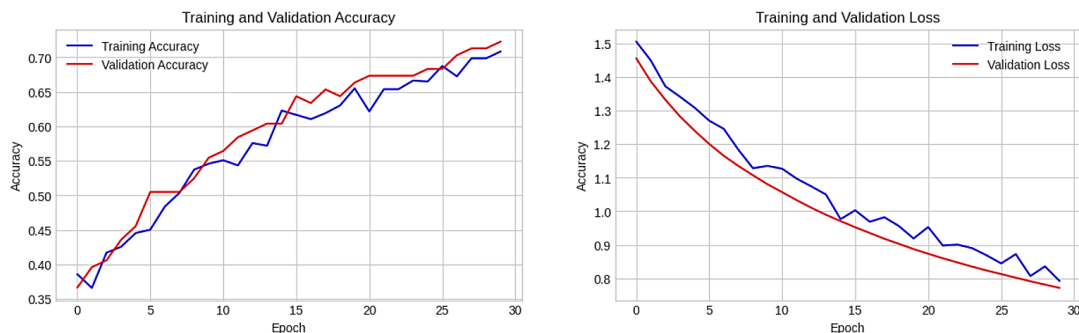
**Figure 13.** Accurasi & Loss ResNet50V2,SGD

Figure 13, Training Loss shows the loss values on the training and validation data. From the beginning to the end of training, the training and validation loss both drop dramatically, dropping from about 1.5 to about 0.8. The balanced decrease between the two curves indicates that the model fits well with the training and validation data. There are no signs of overfitting or significant differences between the two loss curves, so it can be concluded that the model learns well and the instruction goes well.

3.6.2. ResNet50V2 & RMSprop

Figure 14, Training Accuracy shows the loss values on the training and validation data. From the beginning to the end of training, the training and validation loss both drop dramatically, dropping from about 1.5 to about 0.8. The balanced decrease between the two curves indicates that the model fits well with both training and validation data. There are no signs of overfitting or significant differences between the two loss curves, so it can be concluded that the model learns well and the instruction goes well.

Figure 14, Training Loss shows that the loss value on the training data continues to decrease consistently until it approaches 0.1, while the loss value on the validation data decreases initially but then stabilizes and tends to increase after the fifteenth epoch. This pattern indicates that although the learning model works well with the training data, its ability to generalize to new (validation) data does not improve evenly. Therefore, in situations like this, the use of additional regularization methods such as dropout, model complexity reduction, or early delay can help prevent overfitting.

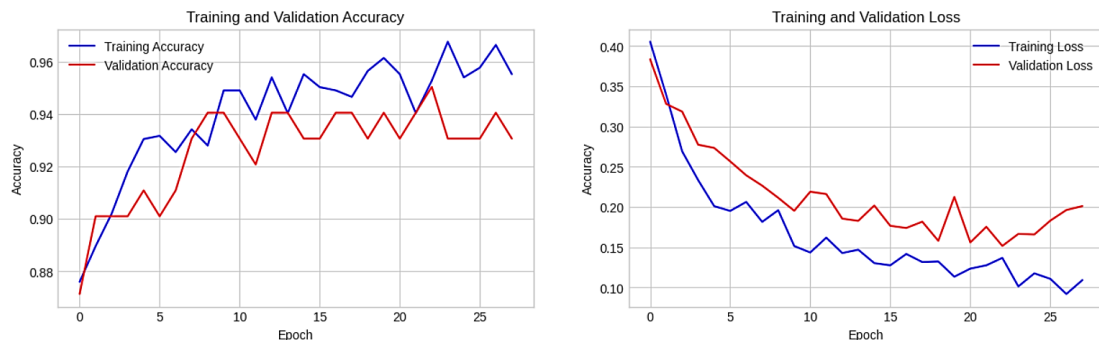


Figure 14. Accurasi & Loss ResNet50V2,RMSprop

3.6.3. ResNet50V2 & Adam

Figure 15, Training Accuracy shows excellent performance, as indicated by the graph showing the accuracy increasing sharply at first and then stabilizing at a high number after some time. The training and validation accuracies are both above 90%, indicating that the model is able to learn from the data effectively without losing generalization to new data. There is no significant difference between the two lines, indicating that the model is not overfitting.

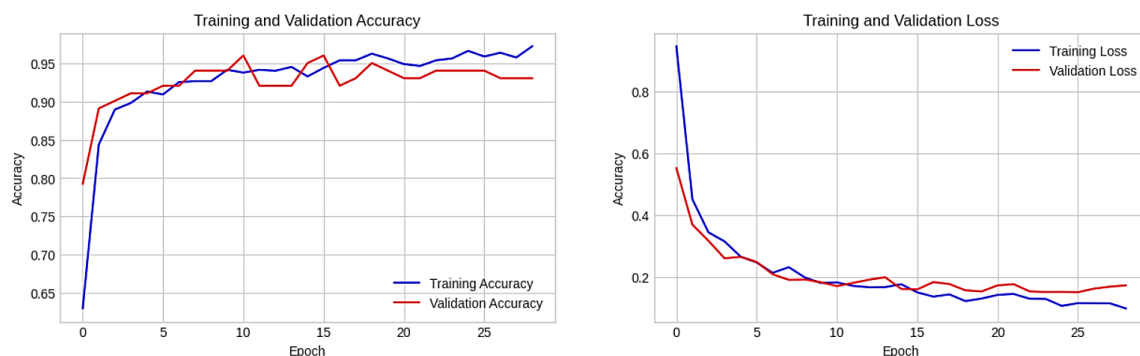


Figure 15. Accurasi & Loss ResNet50V2,Adam

Figure 15, Training Loss also supports this. The validation and training loss values are both decreasing with a stable trend, but the validation loss changes slightly. The small difference between the two shows that the model is quite balanced in learning. Overall, the model is already good enough. You can even stop training early to save time without compromising performance.

3.6.4. InceptionV3 & SGD

Figure 16, Training Accuracy shows a fairly consistent improvement in performance, although it is still in the early learning stage. The accuracy graph shows an upward trend for both training and validation data, but the numbers are still low (only around 55%), indicating that the model is starting to understand the data patterns, although not yet fully optimized. Overfitting is not yet a big problem, as there is a small gap between training and validation.

Figure 16, Training Loss, shows that the validation loss and assessment continue to decrease gradually. This consistent decrease indicates that the model is still learning and stabilizing. Although the performance is not ideal, the learning direction is appropriate, and the model can still be improved by changing the architecture, number of epochs, or other optimization techniques.

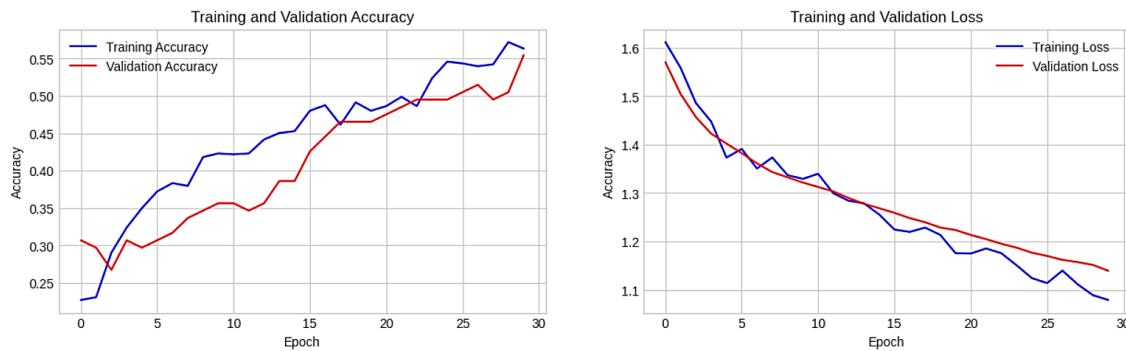


Figure 16. Accurasi & Loss InceptionV3,SGD

3.6.5. InceptionV3 & RMSprop

Figure 17, Training Accuracy It is evident that the model works very well. Training and validation accuracies both increase sharply at the start of training and then stabilize at a high level, reaching over 90 percent. This shows that validation is slightly better than training, indicating that the model can perform excellent generalization to data that has never been seen before.

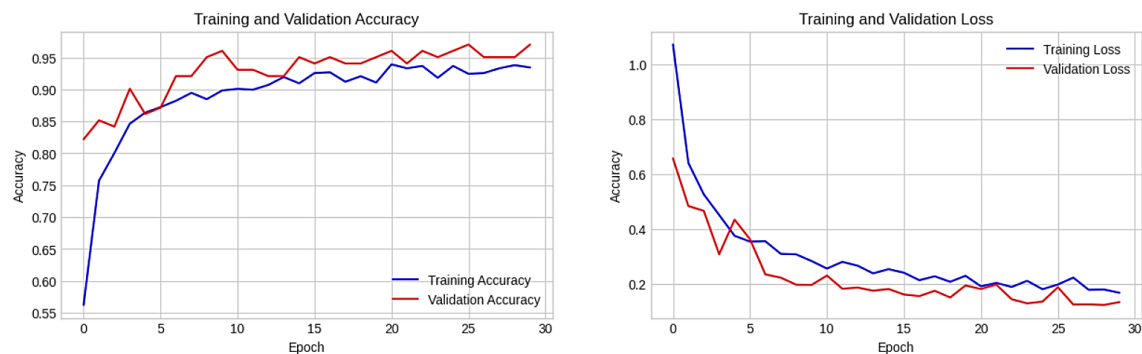


Figure 17. Accurasi & Loss InceptionV3,RMSprop

Figure 17, Training Loss also supports the results. After a drastic drop at the beginning of the period, the loss value continues to shrink until it stabilizes at a low number with no significant spikes. Interestingly, the validation loss is usually smaller than the instruction loss. This may indicate good regularization or slightly easier validation data. In general, the model shows balanced and optimal training results.

3.6.6. InceptionV3 & Adam

Figure 18, Training Accuracy shows a fairly good achievement. The training and validation accuracies both increase rapidly at the beginning of training and then tend to stabilize at a high level, around 85-90%. Although the validation accuracy drops slightly under training, the difference is not very large, indicating that the model can still perform good generalization to new data.

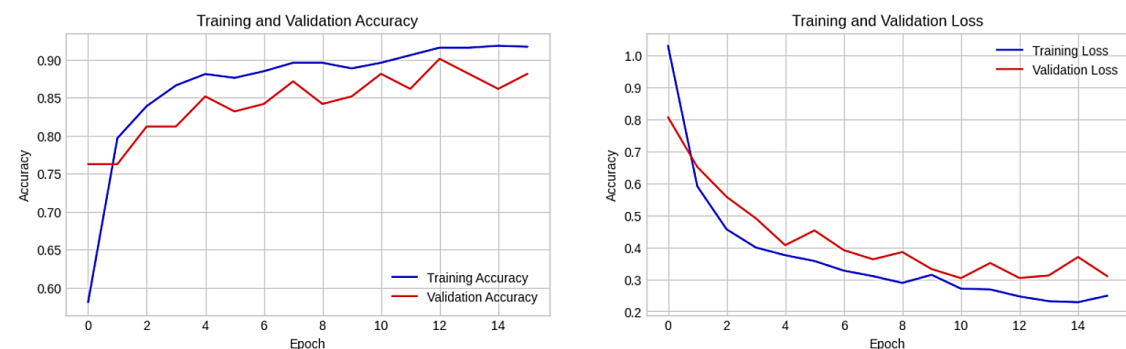


Figure 18. Accurasi & Loss InceptionV3,Adam

Figure 18, Training Loss shows that the training and validation loss values both decrease steadily, but the validation loss seems to be slightly more dynamic after the fifth period. This may be an early indication of slight overfitting, but it is still within reasonable limits. Overall, the model is good enough; to improve the stability of the validation loss, some additional adjustments or regularization methods may be needed.

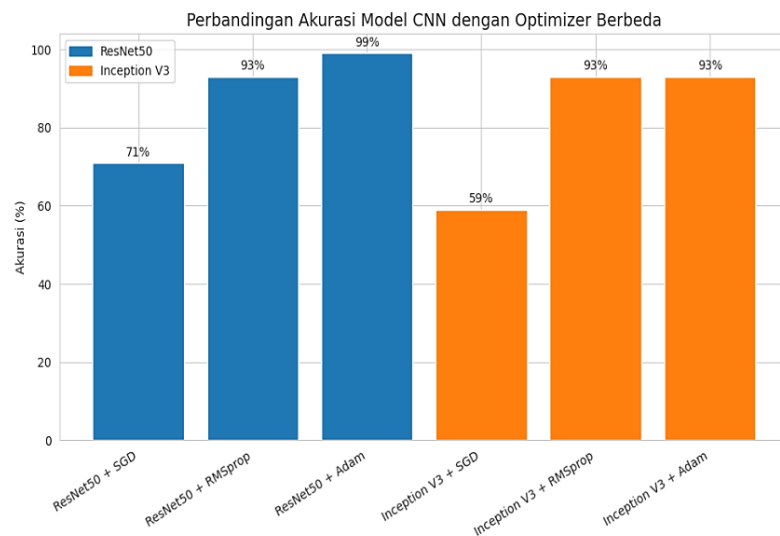


Figure 19. Comparison of the Accuracy

The visualization illustrates the comparison of the accuracy levels of two CNN model architectures (see Figure 19), namely ResNet50 and Inception V3, which were each tested with three types of optimizers: SGD, RMSprop, and Adam. Based on the evaluation results, the combination of ResNet50 with the Adam optimizer recorded the highest accuracy of 99%. Meanwhile, ResNet50 with RMSprop produced an accuracy of 93%, and with SGD it only reached 71%. For the Inception V3 model, the use of RMSprop and Adam both produced 93% accuracy, while the combination with SGD showed the lowest performance with 59% accuracy. From the overall results, it can be concluded that ResNet50 performs better than Inception V3, especially when combined with the Adam optimizer.

3.7. Discussion

The model evaluation results show that the optimizer selection greatly affects the classification performance of road defect images. The Adam and RMSprop optimizers consistently provide more stable and accurate results than SGD, as indicated by high accuracy, precision, recall, and F1 score metrics on all classes, especially when used on the ResNet50V2 and InceptionV3 architectures. In contrast, combining the model with SGD tends to result in suboptimal performance, especially in classes with lower amounts of data, such as the very poor, tagged class.

In addition, the displayed accuracy and loss graphs show that the model with the Adam optimizer and RMSprop exhibits a stable training trend, high accuracy, and significantly decreased loss values. This indicates that the model is capable of good generalization to new data. Although the accuracy of SGD increased gradually, the final value was still much lower than that of Adam and RMSprop. Therefore, it can be concluded that to achieve the best performance in road condition classification, the use of adaptive optimizers such as Adam and RMSprop is recommended.

The study's findings demonstrate that, in comparison to SGD, the use of adaptive optimizers—specifically, Adam and RMSprop—produces more accurate and consistent results. These results are consistent with a work by Parasnis et al [11] that found that RMSprop maintained training stability while increasing road image detection models' accuracy to over 96%. Nonetheless, the work by Swain & Tripathy (2024), which employed the VGG-16 architecture with an accuracy of about 87%, was significantly outperformed by the 99% accuracy attained with the ResNet50V2 + Adam combo. This discrepancy demonstrates how adaptive optimizers, in conjunction with deeper topologies like ResNet50V2, can yield significant performance gains.

4. CONCLUSION

This study successfully achieved its objective of comparing ResNet50V2 and InceptionV3 architectures with three optimizers (Adam, RMSprop, and SGD) in classifying road conditions. The results show that the combination of ResNet50V2 with Adam yielded the best performance, reaching 99% accuracy, which confirms the effectiveness of adaptive optimizers in handling class imbalance. These findings

demonstrate the practical benefit of the proposed approach for supporting automated road monitoring systems that can be applied by public works or transportation agencies to improve infrastructure management. However, this study has a number of drawbacks, such as the short dataset size (only 1,000 images), the concentration on two CNN architectures, and the lack of adequate validation techniques. It is therefore advised that future studies investigate bigger and more varied datasets, use extra regularization strategies like dropout or early stopping, and compare the model to other CNN architectures like DenseNet or EfficientNet in order to improve the model's robustness and generalization.

REFERENCES

- [1] A. Paramarthalingam, J. Sivaraman, and P. Theerthagiri, "A deep learning model to assist visually impaired in pothole detection using computer vision," *Decis. Anal. J.*, vol. 12, no. August, p. 100507, 2024, doi: 10.1016/j.dajour.2024.100507.
- [2] S. Swain and A. K. Tripathy, "Automatic detection of potholes using VGG-16 pre-trained network and Convolutional Neural Network," *Heliyon*, vol. 10, no. 10, p. e30957, 2024, doi: 10.1016/j.heliyon.2024.e30957.
- [3] B. Omarov and B. Kulambayev, "ScienceDirect ScienceDirect Deep Learning-Based Image Processing for Real-Time Detection of Deep Learning-Based Image Processing for Real-Time Detection of Road Surface Damage Road Surface Damage," *Procedia Comput. Sci.*, vol. 251, no. 2023, pp. 609–614, 2024, doi: 10.1016/j.procs.2024.11.157.
- [4] D. Li, Z. Duan, X. Hu, D. Zhang, and Y. Zhang, "Automated classification and detection of multiple pavement distress images based on deep learning," *J. Traffic Transp. Eng. (English Ed.)*, vol. 10, no. 2, pp. 276–290, 2023, doi: 10.1016/j.jtte.2021.04.008.
- [5] H. Song, K. Baek, and Y. Byun, "Pothole Detection using Machine Learning," no. February, pp. 151–155, 2018, doi: 10.14257/astl.2018.150.35.
- [6] P. D. Rinanda, D. N. Aini, T. A. Pertiwi, S. Suryani, and A. J. Prakash, "Implementation of Convolutional Neural Network (CNN) for Image Classification of Leaf Disease In Mango Plants Using Deep Learning Approach," *Public Res. J. Eng. Data Technol. Comput. Sci.*, vol. 1, no. 2, pp. 56–61, 2024, doi: 10.57152/predatecs.v1i2.872.
- [7] D. Lestari, A. Mulya, A. Tatamara, R. R. Haiban, and H. D. Khalifah, "Deep Learning for Pneumonia Detection in Chest X-Rays using Different Algorithms and Transfer Learning Architectures," *Public Res. J. Eng. Data Technol. Comput. Sci.*, vol. 3, no. 1, pp. 10–21, 2025.
- [8] V. Wulandari, W. J. Sari, Z. H. Al-sawaff, and S. Manickam, "Comparative Analysis of Weather Image Classification Using CNN Algorithm with InceptionV3 , DenseNet169 and NASNetMobile Architecture Models," *Inst. Res. Publ. Indones.*, vol. 2, no. January, pp. 81–92, 2025, doi: <https://doi.org/10.57152/predatecs.v2i2.1609>.
- [9] C. Saisree and U. Kumaran, "Pothole Detection Using Deep Learning Classification Method," *Procedia Comput. Sci.*, vol. 218, pp. 2143–2152, 2022, doi: 10.1016/j.procs.2023.01.190.
- [10] C. Lin and Y. Liu, "Research on Potholes Detection Based on Improved Mask RCNN Algorithms".
- [11] G. Parasnis, A. Chokshi, V. Jain, and K. Devadkar, "RoadScan: A Novel and Robust Transfer Learning Framework for Autonomous Pothole Detection in Roads," 2023 IEEE 7th Conf. Inf. Commun. Technol. CICT 2023, 2023, doi: 10.1109/CICT59886.2023.10455143.
- [12] S. Dutta, "International Journal of Scientific Research and Reviews," *Int. J. Sci. Res. Rev.*, vol. 8, no. 2, pp. 2475–2483, 2019.
- [13] R. K. Ibrahim, R. Mustapha, and K. Haruna, "Enhanced Deep learning for Pothole Detection in Autonomous Vehicles .," vol. 1, no. 1, pp. 91–100, 2024.
- [14] D. Anggara, N. Suarna, and Y. Arie Wijaya, "Performance Comparison Analysis Of Optimizer Adam, SGD, and RMSPROP on The H5 Model," *J. Ilm. NERO*, vol. 8, no. 1, p. 2023, 2023.
- [15] H. Maeda, Y. Sekimoto, T. Seto, T. Kashiya, and H. Omata, "Road Damage Detection and Classification Using Deep Neural Networks with Smartphone Images," *Comput. Civ. Infrastruct. Eng.*, vol. 33, no. 12, pp. 1127–1141, 2018, doi: 10.1111/mice.12387.
- [16] D. Arya, H. Maeda, S. K. Ghosh, D. Toshniwal, and Y. Sekimoto, "RDD2022: A multi-national image dataset for automatic road damage detection - Arya - Geoscience Data Journal - Wiley Online Library." pp. 846–862, 2023. doi: 10.1002/gdj3.260.
- [17] M. Fasounaki, E. B. Yüce, S. Öncül, and G. Ince, "CNN-based Text-independent Automatic Speaker Identification Using Short Utterances," *Proc. - 6th Int. Conf. Comput. Sci. Eng. UBMK 2021*, vol. 01, pp. 413–418, 2021, doi: 10.1109/UBMK52708.2021.9559031.
- [18] S. S. Sulaiman, I. Nadher, and S. M. Hameed, "Credit Card Fraud Detection Using Improved Deep Learning Models," *Comput. Mater. Contin.*, vol. 78, no. 1, pp. 1049–1069, 2024, doi: 10.32604/cmc.2023.046051.
- [19] M. Tavakoli, R. Chandra, F. Tian, and C. Bravo, "Multi-modal deep learning for credit rating prediction using text and numerical data streams," *Appl. Soft Comput.*, vol. 171, no. October 2024, p.

- 112771, 2025, doi: 10.1016/j.asoc.2025.112771.
- [20] S. Hong, Y. Zhou, J. Shang, C. Xiao, and J. Sun, "Opportunities and challenges of deep learning methods for electrocardiogram data: A systematic review," *Comput. Biol. Med.*, vol. 122, 2020, doi: 10.1016/j.combiomed.2020.103801.
- [21] Verdy and Ery Hartati, "Klasifikasi Penyakit Mata Menggunakan Convolutional Neural Network Model Resnet-50," *J. Rekayasa Sist. Inf. dan Teknol.*, vol. 1, no. 3, pp. 199–206, 2024, doi: 10.59407/jrsit.v1i3.529.
- [22] A. Zalvadila, "Klasifikasi Penyakit Tanaman Bawang Merah Menggunakan Metode SVM dan CNN," *J. Inform. J. Pengemb. IT*, vol. 8, no. 3, pp. 255–260, 2023, doi: 10.30591/jpit.v8i3.5341.
- [23] D. Iswanto and D. Handayani UN, "Klasifikasi Penyakit Tanaman Jagung Menggunakan Metode Convolutional Neural Network (CNN)," *J. Ilm. Univ. Batanghari Jambi*, vol. 22, no. 2, p. 900, 2022, doi: 10.33087/jiubj.v22i2.2065.
- [24] S. Dewi, F. Ramadhani, and S. Djasmayena, "Klasifikasi Jenis Jerawat Berdasarkan Gambar Menggunakan Algoritma CNN (Convolutional Neural Network)," *Hello World J. Ilmu Komput.*, vol. 3, no. 2, pp. 68–73, 2024, doi: 10.56211/helloworld.v3i2.518.
- [25] J. Alberto and D. Hermanto, "Klasifikasi Jenis Burung Menggunakan Metode CNN ResNet-50 [1]," *J. Tek. Inform. dan Sist. Inf.*, vol. 10, no. 3, pp. 34–36, 2023.
- [26] M. L. Septipalan, M. S. Hibrizi, N. Latifah, R. Lina, and F. Bimantoro, "Klasifikasi Tumor Otak Menggunakan CNN Dengan Arsitektur Resnet50," *Semin. Nas. Teknol. Sains*, vol. 3, no. 1, pp. 103–108, 2024, doi: 10.29407/stains.v3i1.4357.
- [27] J. Algoritme et al., "Pengenalan Motif Songket Palembang Menggunakan Convolutional Neural Network dengan Arsitektur ResNet-50," vol. 5, no. 1, pp. 78–87, 2024, doi: 10.35957/algoritme.xxxx.
- [28] E. Andreas and W. Widhiarso, "Klasifikasi Penyakit Mata Katarak Menggunakan Convolutional Neural Network Dengan Arsitektur Inception V3," *MDP Student Conf.*, vol. 2, no. 1, pp. 107–113, 2023, doi: 10.35957/mdp-sc.v2i1.3660.
- [29] K. Zairan Maulana and A. Susanto, "Implementasi Arsitektur CNN Inception V3 untuk Identifikasi Spesies Burung Endemik di Indonesia," *J. Pustaka Robot Sister (Jurnal Pus. Akses Kaji. Robot. Sist. Tertanam, dan Sist. Terdistribusi)*, vol. 2, no. 1, pp. 22–27, 2024, doi: 10.55382/jurnalpustakarobotsister.v2i1.775.
- [30] U. UNGKAWA and G. AL HAKIM, "Klasifikasi Warna pada Kematangan Buah Kopi Kuning menggunakan Metode CNN Inception V3," *ELKOMIKA J. Tek. Energi Elektr. Tek. Telekomun. Tek. Elektron.*, vol. 11, no. 3, p. 731, 2023, doi: 10.26760/elkomika.v11i3.731.
- [31] D. Soydaner, "A Comparison of Optimization Algorithms for Deep Learning," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 34, no. 13, 2020, doi: 10.1142/S0218001420520138.
- [32] I. Chakroun, T. Haber, and T. J. Ashby, "SW-SGD: The Sliding Window Stochastic Gradient Descent Algorithm," *Procedia Comput. Sci.*, vol. 108, pp. 2318–2322, 2017, doi: 10.1016/j.procs.2017.05.082.
- [33] B. Soujanya and T. Sitamahalakshmi, "Optimization with ADAM and RMSprop in Convolution neural Network (CNN): A Case study for Telugu Handwritten Characters," *Int. J. Emerg. Trends Eng. Res.*, vol. 8, no. 9, pp. 5116–5121, 2020, doi: 10.30534/ijeter/2020/38892020.
- [34] A. Pasha et al., "Leveraging ANFIS with Adam and PSO optimizers for Parkinson's disease," *Heliyon*, vol. 10, no. 9, p. e30241, 2024, doi: 10.1016/j.heliyon.2024.e30241.
- [35] L. Yang, "Theoretical Analysis of Adam Optimizer in the Presence of Gradient Skewness," vol. 7, no. 2, pp. 27–42, 2024.
- [36] K. Penyakit, D. Anggur, W. Rohman, N. Rifa, D. Erwanto, and I. Yanuartanti, "Evaluasi Kinerja CNN dengan Optimizer RMSprop, Adam dan," pp. 91–104.
- [37] P. R. Kiran and N. C. Naveen, "Op-RMSprop (Optimized-Root Mean Square Propagation) Classification for Prediction of Polycystic Ovary Syndrome (PCOS) using Hybrid Machine Learning Technique," *Int. J. Adv. Comput. Sci. Appl.*, vol. 13, no. 6, pp. 588–596, 2022, doi: 10.14569/IJACSA.2022.0130671.