

## Article

Egidio Bagus Sudewo, Muhammad Kunta Biddinika\*, Rusydi Umar and Abdul Fadlil

# Evaluating the Impact of Optimizer Hyperparameters on ResNet in Hanacaraka Character Recognition

<https://doi.org/10.1515/pdtc-2024-0061>

Received September 21, 2024; accepted January 26, 2025;

published online February 24, 2025

**Abstract:** This study evaluates the performance of various optimizers on a ResNet-18 based Convolutional Neural Network (CNN) model for the task of recognizing Hanacaraka Javanese script characters. The image dataset is divided into three sets: training, validation, and test, with image sizes of  $64 \times 64$  pixels and a batch size of 64. The tested optimizers include SGD, Adam, RMSprop, Adagrad, Adadelta, NAdam, and Adamax, all with a learning rate of 0.001 and trained for 10 epochs. The results show that NAdam provides the best performance with accuracy, precision, recall, and F1-Score values reaching 100 %, followed by Adamax with metrics above 97 %. Adam and Adagrad also demonstrate high performance with metric values above 97 %. Meanwhile, SGD shows fairly good performance with an accuracy of 93.72 %, and Adadelta shows adequate performance with an accuracy of 86.58 %. RMSprop yields the lowest performance with an accuracy of 81.74 %. Accuracy and loss graphs indicate that Adam and Adadelta offer the best balance between training and validation performance, while RMSprop and NAdam exhibit significant instability. This study highlights the importance of selecting the appropriate optimizer to achieve optimal performance in Hanacaraka character classification, with NAdam and Adamax being the best choices.

**Keywords:** Hanacaraka Javanese script; CNN; optimizer; character recognition; ResNet-18

## 1 Introduction

Character recognition is a crucial field in image processing and computer vision, with various applications such as Optical Character Recognition (OCR), handwriting recognition, and document digitization. In the context of Hanacaraka character recognition, the traditional Javanese script, the challenges are significant due to its complex and diverse characters.

ResNet (Residual Network) is a convolutional neural network architecture that has proven to be highly effective in various image recognition tasks (Han-wen et al. 2021; Singh and Schicker 2021). One of the main advantages of ResNet is its ability to overcome the vanishing gradient problem through the use of residual blocks (Nicholas et al. 2022; Sudewo, Biddinika, and Fadlil 2024b). This allows for the creation of very deep networks without losing critical information during training.

Recent studies demonstrate the capability of Convolutional Neural Networks (CNN) in solving diverse classification problems with high accuracy and efficiency. Murinto and Melany (2023) showed that MobileNetV2 combined with transfer learning improved classification accuracy for coffee beans up to 96 %, outperforming conventional CNN approaches (Murinto and Melany 2023). Similarly, Cahya and Murinto (2021) achieved remarkable results in classifying batik motifs, with a training accuracy of 100 % and testing accuracy of 99 %, underscoring the potential of CNN for intricate pattern recognition tasks (Cahya and Murinto 2021). In another study, Rosyda (2022) successfully applied the Log-DIWPSO optimization algorithm to improve CNN performance on the CIFAR-10 dataset, demonstrating a significant accuracy increase from 28.07 % to 69.3 % within 10 epochs (Rosyda 2022). Lei, Pan, and Huang (2019) introduced the Hybrid Dilated CNN (HDC) model, reducing training time while enhancing accuracy by over 14 %, showcasing advancements in architectural innovations to improve CNN efficiency and precision (Lei, Pan, and Huang 2019).

This study aims to evaluate the impact of optimizer hyperparameters on the performance of ResNet in the task

\*Corresponding author: Muhammad Kunta Biddinika, Master of Informatics Engineering, Universitas Ahmad Dahlan, Yogyakarta, Indonesia, E-mail: muhammad.kunta@mti.uad.ac.id. <https://orcid.org/0000-0003-4104-3755>

Egidio Bagus Sudewo, Rusydi Umar and Abdul Fadlil, Master of Informatics Engineering, Universitas Ahmad Dahlan, Yogyakarta, Indonesia

of Hanacaraka character recognition. By conducting systematic experiments and analyses on various optimizers and their settings, we aim to provide practical guidelines for developing more effective and efficient character recognition models. The integration of state-of-the-art optimization techniques and architectural insights is expected to address the challenges posed by the intricate nature of Hanacaraka script, paving the way for advancements in traditional script recognition technologies.

This study aims to evaluate the impact of optimizer hyperparameters on the performance of ResNet in the task of Hanacaraka character recognition. By conducting systematic experiments and analyses on various optimizers and their settings, we can gain a better understanding of how each optimizer affects the model's convergence, accuracy, and generalization ability. The results of this research are expected to provide practical guidelines for developing more effective and efficient character recognition models.

## 2 Materials and Methods

This study uses a quantitative methodology, utilizing image data as the computational subject using the CNN method with ResNet-18 architecture and a learning rate of 0.001, with the optimizer as an additional comparison component. This research utilizes Google Colab and various libraries as supporting tools.

### 2.1 Research Steps

The flowchart of this research, shown in Figure 1, begins with a literature review to understand the basics and developments in Hanacaraka character recognition using CNN algorithms. Next, a dataset of Javanese script character images is collected and resized to  $64 \times 64$  pixels. The pre-processing step is then performed by converting images to tensors and applying data augmentation if necessary. The ResNet-18 based CNN algorithm is implemented and the model is trained using various optimizers such as SGD, Adam, and others for 10 epochs. After model training, the model is tested with the validation dataset during the model testing phase to measure its performance. The testing results are evaluated based on accuracy, precision, recall, and F1-Score, determining the best optimizer. Finally, the output in the form of predicted images from the CNN model demonstrates the model's performance in recognizing Javanese script characters.

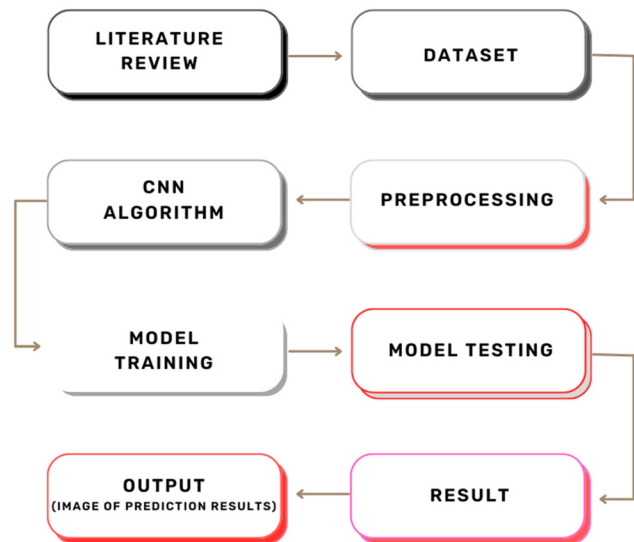


Figure 1: Research steps.

### 2.2 Literature Review

The initial stage of this research involves a literature review to understand the basic concepts and recent developments in Hanacaraka Javanese script character recognition using Convolutional Neural Network (CNN) algorithms. This process helps identify the most effective methods and appropriate strategies for model implementation. The literature used in this research includes several studies relevant to the use of CNN algorithms in various image classification applications. Murinto and Melany (2023) utilized MobileNetV2 and transfer learning on VGG16 and MobileNetV2 models to classify coffee beans, achieving the highest accuracy of 96 %, demonstrating an improvement over conventional CNN models (Murinto and Melany 2023). Cahya and Murinto (2021) classified batik motifs in the southern coastal region of Java using CNN, achieving 100 % training accuracy, 99 % testing accuracy, and 93.3 % validation accuracy using a dataset consisting of 630 training data, 180 validation data, and 90 test data (Cahya and Murinto 2021). Rosyda (2022) applied the Logarithm Decreasing Inertia Weight Particle Swarm Optimization (LogDIWPSO) algorithm to improve CNN accuracy on the CIFAR-10 dataset, reaching an accuracy of 69.3 % from a baseline of 28.07 % at the tenth epoch (Rosyda 2022). Lei, Pan, and Huang (2019) developed the Hybrid Dilated CNN (HDC) model for character image classification, which addresses the loss of detail in dilated CNN, reducing training time by 2.02 % and improving training and testing accuracy by 14.15 % and 15.35 %, respectively. All these studies provide valuable insights into improving CNN model accuracy and efficiency in various image classification applications (Lei, Pan, and Huang 2019).

## 2.3 Datasets

This research uses a dataset of Hanacaraka Javanese script character images sourced from Kaggle, provided by Hanna Hunnafa, as shown in Figure 2. The dataset consists of 12,000 PNG images, divided into 8,400 training data images, 2,400 validation data images, and 1,200 testing data images.

## 2.4 Preprocessing

Preprocessing is a stage to prepare the images before training the model using the dataset. This step is crucial to ensure that all images classified by the CNN method have uniform pixel sizes, as this can affect the accuracy of the results (Hasan et al. 2020). The preprocessing process includes converting image channels from RGB to grayscale and resizing the images so that all data have the same pixel resolution.

Figure 3(a) shows an image sized at  $224 \times 224$  pixels. This size makes it difficult to train the model due to the high number of pixels that need to be extracted, necessitating the

resizing technique during preprocessing. The resizing process involves reducing the image size from  $224 \times 224$  pixels to  $64 \times 64$  pixels as shown in Figure 3(b). Although this size differs significantly from the original size, it does not reduce the information in the  $224 \times 224$  pixel image.

## 2.5 Convolutional Neural Network

Convolutional Neural Network (CNN) is one of the most effective artificial neural network architectures for processing grid-structured data such as images (see Figure 4). CNN consists of several key layers that work sequentially to extract and process important features from the input data (Liu, Pu, and Sun 2021; Muis, Sunardi, and Yudhana 2023). First, the convolutional layer uses filters to capture local patterns such as edges, textures, and other visual patterns from the images. This layer allows the network to learn hierarchical representations of the input where each filter is responsible for extracting specific features (Basha et al. 2020). After that, the pooling layer is used to reduce the dimensions of the feature maps produced by the convolutional layer, as shown in

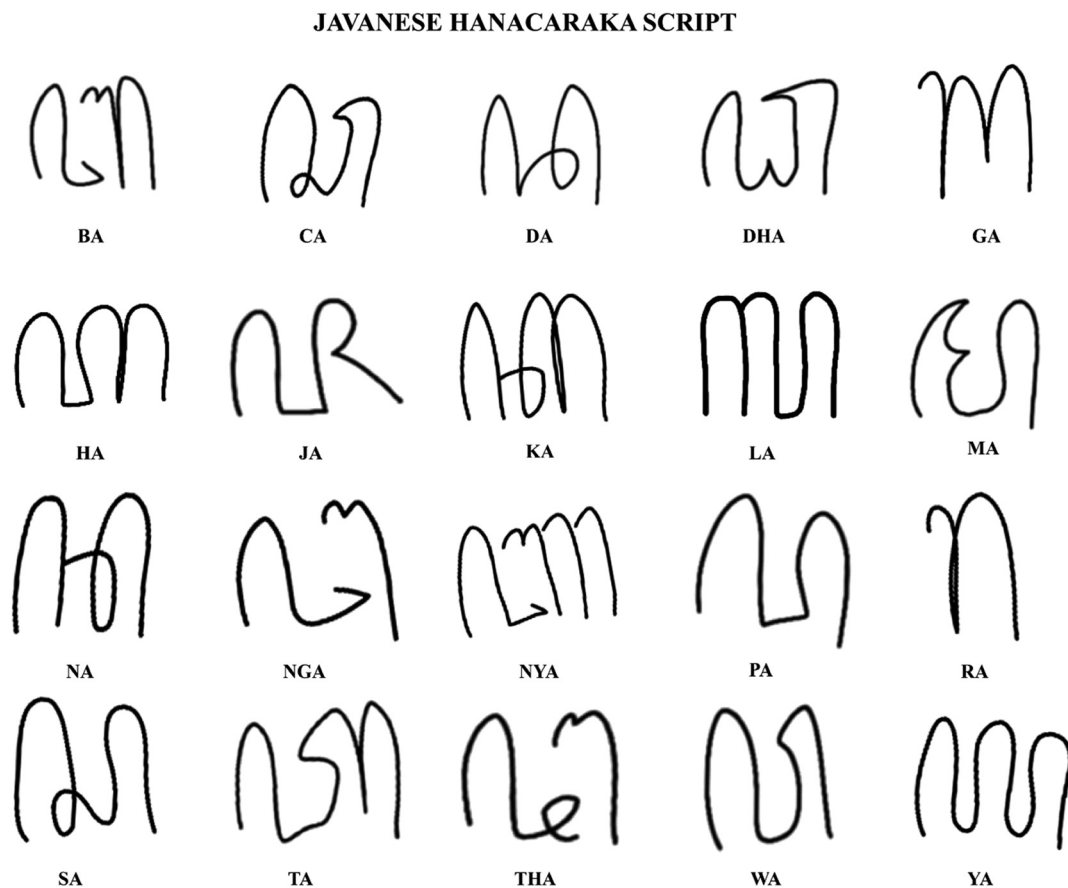
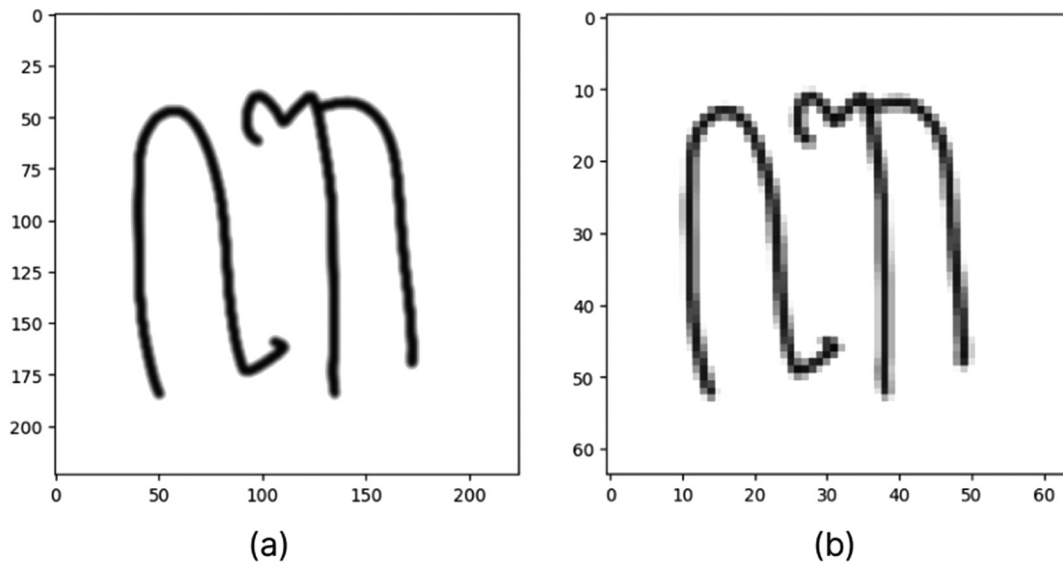


Figure 2: Javanese Hanacaraka script.



**Figure 3:** Preprocessing Hanacaraka Script (a) before (b) after.

Figure 4. This not only reduces computational complexity but also helps prevent overfitting by simplifying the generated representations.

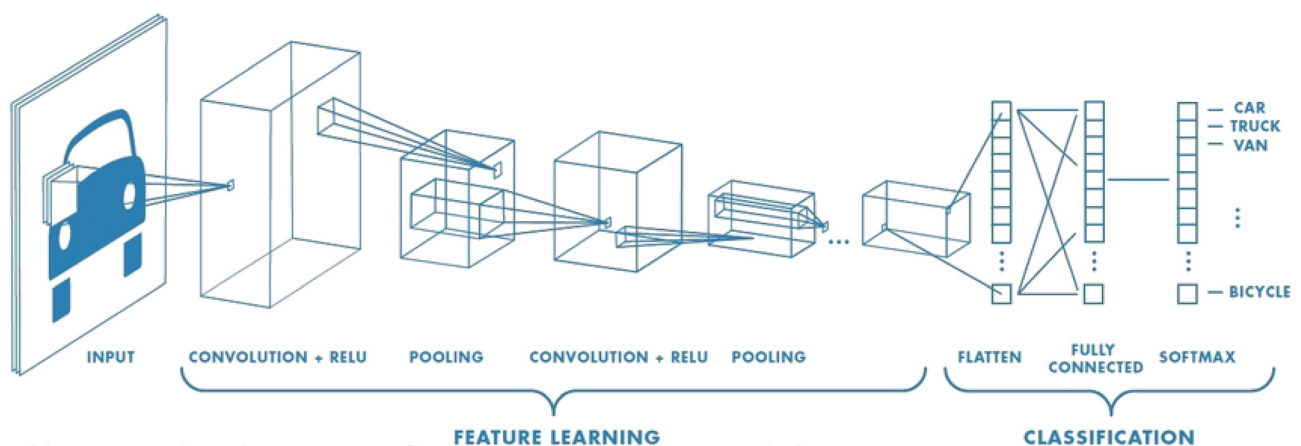
Then, the ReLU (Rectified Linear Unit) layer introduces non-linearity into the network by converting negative values to zero. This is important because non-linearity allows CNN to learn more complex relationships between input features, enhancing the network's ability to accurately classify and recognize objects. Finally, the fully connected layer at the end of the network combines the features extracted from the entire image to make the final decision such as classifying the objects in the image.

Some well-known CNN architectures, such as AlexNet (Madhulatha and Ramadevi 2020), VGGNet (Agrawal and Mittal 2020), GoogLeNet (Inception) (Shadin, Sanjana, and

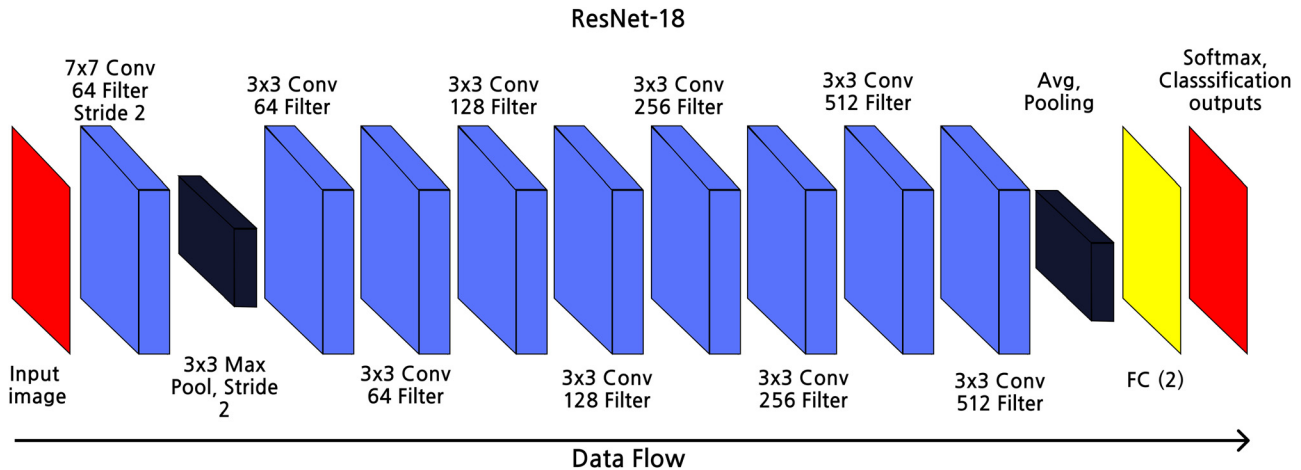
Lisa 2021), and DenseNet (Sudewo, Biddinika, and Fadlil 2024a), each offer innovations in the use of convolutional layers, managing multi-scale information, and addressing gradient issues. CNNs are not only used in image recognition and classification but are also widely applied in object detection, image segmentation, and even natural language processing. Their success in various image processing tasks makes CNN one of the most effective and popular tools in the world of pattern recognition and visual analysis.

## 2.6 ResNet-18

ResNet-18 is a variant of the Residual Network (ResNet) architecture known for balancing relatively shallow depth



**Figure 4:** Convolutional Neural Network (taken from: Researchgate.net).



**Figure 5:** Resnet-18 (source: Researchgate.net).

with its ability to address the vanishing gradient problem (Lu et al. 2023). By using residual blocks, ResNet-18 can build deeper networks without experiencing performance degradation (Chandu and Bharatha Devi 2023). Each block in ResNet-18 consists of several convolutional layers followed by skip connections, allowing direct information flow across multiple layers (Ahmed et al. 2024), as shown in Figure 5.

This capability makes ResNet-18 a popular choice for various image recognition tasks, such as image classification, object detection, and segmentation, due to its combination of good performance and relatively simple structure.

## 2.7 Hyperparameters

Hyperparameters are parameters set before the model training process begins and are not updated during training. They influence the model's behavior and performance but their values are not determined directly from the training data (Bartz et al. 2023). Examples of hyperparameters include learning rate, momentum, the number of layers and neurons in the network, and many others (Roy et al. 2023). The appropriate selection of hyperparameters can affect model convergence, final performance, and generalization ability (Alkaff and Prasetyo 2022).

## 2.8 Optimizers

An optimizer is an algorithm used to update the model's weights based on the gradients of the loss function. The optimizer is responsible for finding the optimal weights that can reduce the model's error (Cong and Zhou 2023). Common optimizers include Stochastic Gradient Descent (SGD), Adaptive Moment Estimation (Adam), Root Mean Square Propagation (RMSprop), as well as many others (Yaqub et al.

2020). Each optimizer has unique characteristics in how they adjust the learning rate and optimize the training process.

Commonly used optimizers include:

- (1) **Stochastic Gradient Descent (SGD):** This method updates the model's weights based on the gradient of the loss function against a random batch of data (Prasher, Nelson, and Sharma 2022). Although simple and efficient for large datasets, SGD tends to be slow in reaching the global minimum and requires careful tuning of the learning rate (Duda 2019).
- (2) **Adaptive Moment Estimation (Adam):** Adam combines the momentum and RMSprop to adaptively adjust the learning rate based on the first moment (mean) and second moment (uncentered variance) of the gradient (Norouzi and Ebrahimi 2019). This accelerates convergence and improves training stability although it requires careful hyperparameter tuning (Mourya and Patil 2024).
- (3) **Root Mean Square Propagation (RMSprop):** This optimizer adjusts the learning rate based on the exponentially weighted average of squared gradients of parameters (Kumar Reddy, Srinivasa Rao, and Prudvi Raju 2018). RMSprop helps stabilize parameter updates and speeds up convergence by considering gradient fluctuations across parameters.
- (4) **Adaptive Gradient Algorithm (AdaGrad):** By adjusting the learning rate based on the historical gradient of parameters, AdaGrad is effective for handling sparse features in the data, although it is prone to monotonically decreasing learning rates (Radha and Prasanna 2024).
- (5) **AdaDelta:** As a variant of AdaGrad, AdaDelta limits gradient accumulation with an exponential moving window, keeping the learning rate stable without manual tuning. This helps maintain consistency and efficiency in neural network training (Sveleba et al. 2023).



- (6) Adamax: A variant of the Adam algorithm that uses the infinity norm to update parameters, providing better numerical stability, especially with large gradients or large-scale data. With decentralized exponential momentum for the first and second moments, Adamax offers more stable parameter updates and resistance to large gradient fluctuations (Das et al. 2023).
- (7) NAdam (Nesterov-accelerated Adaptive Moment Estimation): Combines the Nesterov accelerated gradient (NAG) technique with Adam for more efficient parameter updates. NAdam gives a “peek” at the gradient at future positions to enhance convergence and uses the first and second moment adaptation from Adam, which accelerates convergence and improves training efficiency (Harish et al. 2024).

## 2.9 Model Testing

Accuracy is a model evaluation measure that describes how accurately a classification model predicts the class or label of the data. Accuracy values range from 0 to 1, with a value of 1 indicating perfect predictions or no errors in classification (Riadi, Yudhana, and Djou 2024).

$$\text{Accuracy (\%)} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} \times 100 \%$$

The figures in the matrix provide information regarding TP (true positive), TN (true negative), FP (false positive), and FN (false negative).

## 3 Results and Discussion

The results of comparing hyperparameter optimizers on the CNN model based on ResNet-18 were obtained by training

the model using a dataset of Javanese Hanacaraka script images, divided into three sets: training, validation, and test. The images were resized to  $64 \times 64$  pixels and converted into tensors. The learning rate used was 0.001, and the batch size was 64. The model was implemented using optimizers such as SGD, Adam, RMSprop, Adagrad, Adadelta, Nadam, and Adamax on the validation set and trained for 10 epochs.

In Figure 6, NAdam shows the best performance among all the optimizers tested, with accuracy, precision, recall, and F1-Score reaching 100 %. This indicates that NAdam can optimize the model maximally for the task of recognizing Javanese characters, providing perfect results without errors. The use of the Nesterov accelerated gradient (NAG) technique combined with moment adaptation from Adam appears to offer an advantage in faster and more efficient convergence.

Adamax also provides excellent results, with accuracy, precision, recall, and F1-Score of 97.92 % each. Adamax, as a variant of Adam that uses infinity norm for parameter updates, seems to offer very good numerical stability, allowing the model to handle large fluctuations in gradients.

Adam and Adagrad show high performance with metric values above 97 %, making both strong choices for the task of Hanacaraka character recognition. Adam, known for its combination of momentum and learning rate adaptation, and Adagrad, which adjusts the learning rate based on gradient frequency, both provide consistent and accurate results.

SGD (Stochastic Gradient Descent) performs quite well with 93.72 % accuracy and 93.57 % F1-Score. Although not as good as more advanced optimizers like Adam or NAdam, SGD remains a reliable and simple choice, especially for tasks that are not overly complex.

Adadelta shows adequate performance with 86.58 % accuracy and 86.15 % F1-Score. While not as effective as Adam or Adamax, Adadelta remains a solid choice, particularly when stability in parameter updates is a primary consideration.

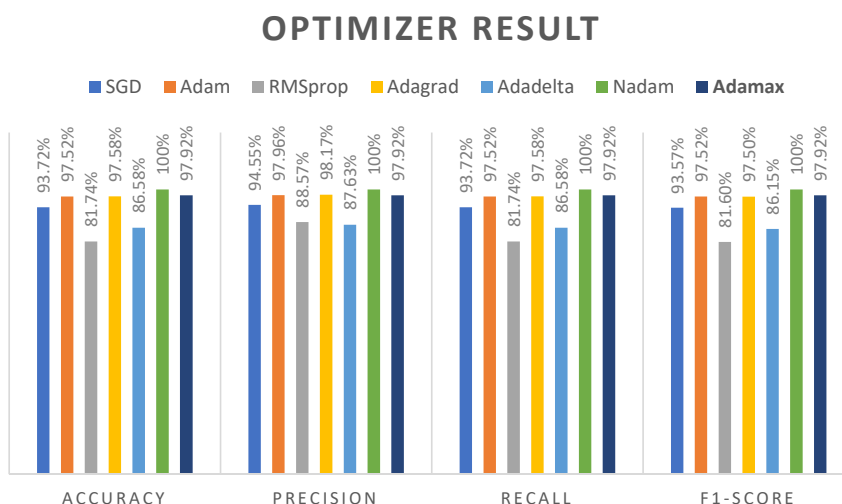


Figure 6: Training results.

Conversely, RMSprop yields the lowest performance with 81.74 % accuracy and 81.60 % F1-Score. RMSprop, designed to address the declining learning rate issue in Adagrad, appears less effective for the task of Hanacaraka character recognition in this study. This lower performance may be due to its inability to handle large variations in gradients as effectively as other optimizers.

The evaluation results show that the choice of optimizer significantly impacts the model's performance in the task of recognizing Javanese Hanacaraka characters. NAdam and Adamax proved to be the best choices, providing highly accurate and consistent results in detecting Javanese characters. NAdam's advantage is particularly evident from the combination of the NAG technique and moment adaptation, offering faster and more stable convergence.

Adam and Adagrad also demonstrate very good performance, reflecting the effectiveness of algorithms capable of adjusting the learning rate during training. Although not as effective as more advanced optimizers, SGD still provides solid and reliable results in many situations.

In contrast, RMSprop's performance shows that although some optimizers are designed to address specific weaknesses in other algorithms, they may not always be suitable for all types of tasks. In this case, RMSprop was less effective in handling large gradient variations for Hanacaraka character recognition.

Overall, this study emphasizes the importance of selecting the appropriate optimizer to achieve optimal performance in Hanacaraka character classification. Optimizers such as NAdam and Adamax show great potential for practical applications in character recognition while results obtained from other optimizers provide valuable insights into their effectiveness under various training conditions.

Table 1 presents the evaluation results of various optimizers in recognizing Javanese Hanacaraka characters, covering precision, recall, and F1-Score metrics for each character. From the table, it is evident that the NAdam and Adamax optimizers consistently deliver high performance across almost all characters, with precision, recall, and F1-Score values close to or reaching 1.00. For instance, for the character "ba," these two optimizers achieve perfect scores in all three metrics. This indicates that these optimizers are highly effective in accurately and consistently identifying Javanese script characters. Conversely, optimizers such as RMSprop, Adagrad, and Adadelata exhibit greater performance variation. For example, RMSprop has low precision values for some characters like "da" (0.85) and "ba" (0.73) while Adagrad and Adadelata show good results for some characters but lack overall consistency. SGD and Adam consistently deliver good results, though still falling short of NAdam and Adamax in some cases. From these results, it can

be concluded that NAdam and Adamax are the best choices for the task of recognizing Javanese Hanacaraka characters, providing higher accuracy and consistency compared to other optimizers.

Figure 7 displays a series of graphs comparing the performance of various optimizers used in training the neural network. Each graph shows loss and accuracy for training and validation data over several epochs. The optimizers compared include Adam, SGD, RMSprop, AdaGrad, Adadelata, and NAdam. The four lines on each graph represent Training Loss (blue), Validation Loss (orange), Training Accuracy (green), and Validation Accuracy (red).

The Adam optimizer demonstrates stable performance with a consistent decrease in Training Loss and a gradual increase in Training Accuracy. However, slight overfitting is visible from the difference between Training Accuracy and Validation Accuracy. The Adadelata optimizer shows a similar pattern, with a consistently decreasing training loss and increasing accuracy for both training and validation data, indicating better generalization compared to other optimizers.

On the other hand, RMSprop and NAdam optimizers show significant instability, evidenced by large fluctuations in Validation Loss and Validation Accuracy. Although Training Loss for these optimizers decreases well, the instability in validation data suggests that the model may have generalization issues. This makes these optimizers less reliable than Adam and Adadelata in terms of overall performance.

The SGD optimizer shows a rapid decrease in Training Loss but reaches a plateau, indicating that adjustments to the learning rate or more epochs might be needed for better results. Meanwhile, the AdaGrad optimizer seems to experience overfitting, with Training Accuracy increasing but Validation Accuracy remaining flat. This indicates that while the model learns well on training data its performance does not improve on validation data. Overall, Adam and Adadelata appear to provide the best balance between training and validation performance while other optimizers exhibit issues with overfitting or instability. There are some differences with the literature review results, as shown in Table 2.

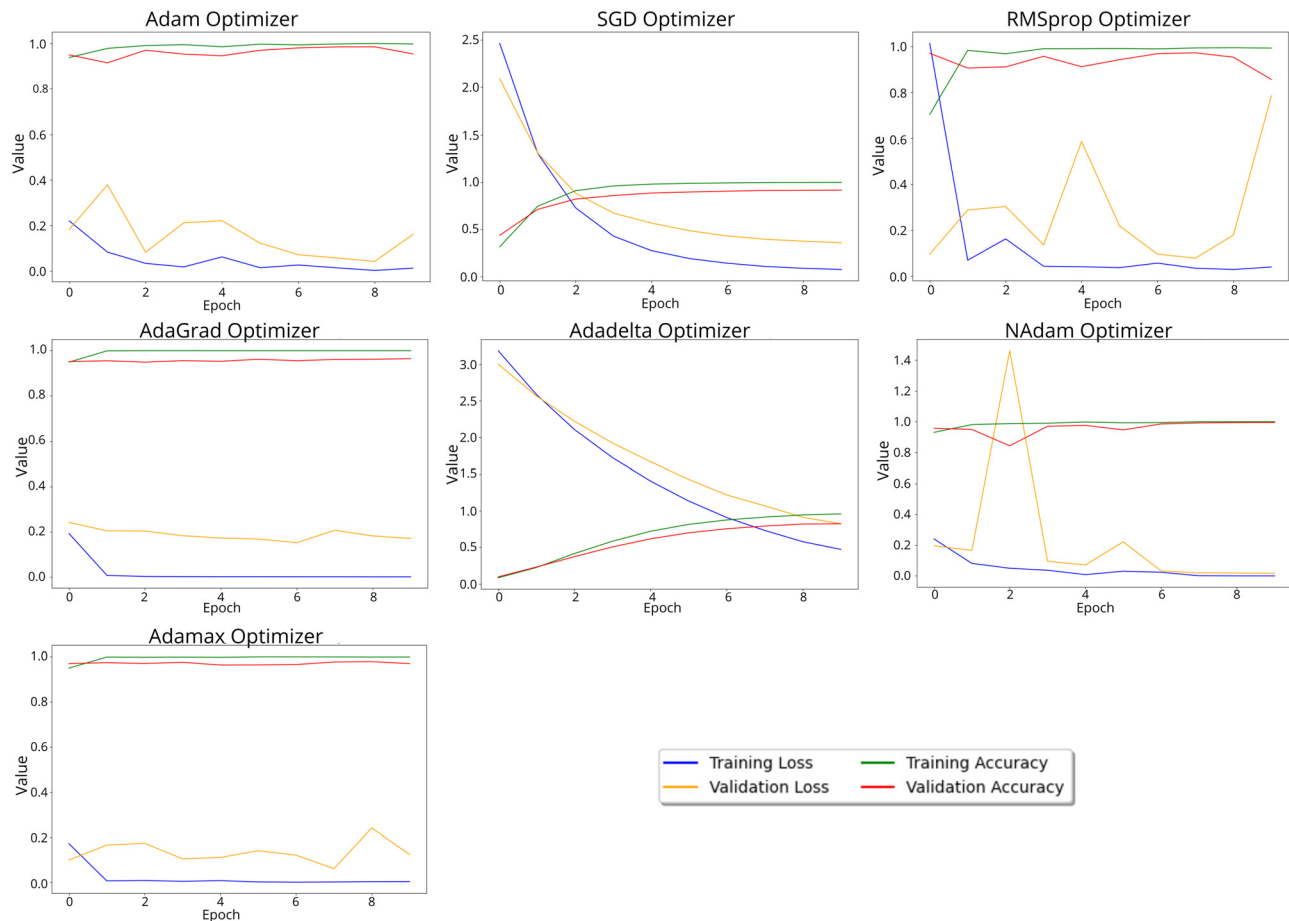
In the comparison of hyperparameter optimization results for the ResNet-18 based CNN model in recognizing Javanese Hanacaraka characters, NAdam proved to be the best optimizer with accuracy, precision, recall, and F1-Score reaching 100 %. This demonstrates NAdam's effectiveness in optimizing the model for this task, outperforming other optimizers such as Adamax, Adam, and SGD which also showed good performance, albeit with slightly lower values.

Comparisons with other studies show variations in CNN application and results depending on the object. For

Table 1: Results of precision, recall, and F1-Score of Javanese script letters.

Characters	Optimizer																				
	SGD			Adam			RMSprop			Adagrad			Adadelta			NAdam			Adamax		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score
ba	0.88	0.98	0.93	1.00	1.00	1.00	0.73	0.93	0.82	1.00	1.00	1.00	0.84	0.95	0.89	1.00	1.00	1.00	1.00	1.00	1.00
ca	0.98	0.97	0.97	1.00	0.97	0.98	1.00	0.82	0.90	1.00	0.98	0.99	0.94	0.55	0.69	1.00	1.00	1.00	1.00	0.98	0.99
da	1.00	0.93	0.97	1.00	1.00	1.00	0.85	0.48	0.62	0.97	1.00	0.98	0.73	0.85	0.78	1.00	1.00	1.00	0.97	1.00	0.98
dha	0.97	1.00	0.98	1.00	1.00	1.00	0.95	0.98	0.97	1.00	1.00	1.00	0.92	1.00	0.96	1.00	1.00	1.00	0.98	1.00	0.99
ga	0.87	1.00	0.93	0.78	1.00	0.88	0.98	0.83	0.90	1.00	1.00	1.00	0.82	0.97	0.89	1.00	1.00	1.00	1.00	1.00	1.00
ha	0.69	0.99	0.81	1.00	0.86	0.92	1.00	0.57	0.73	0.71	1.00	0.83	0.63	0.95	0.75	1.00	1.00	1.00	0.81	1.00	0.90
ja	1.00	0.97	0.98	1.00	1.00	1.00	0.95	1.00	0.98	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
ka	0.91	1.00	0.95	0.97	1.00	0.98	1.00	0.53	0.70	0.97	1.00	0.98	0.95	0.90	0.92	1.00	1.00	1.00	0.91	0.97	0.94
la	0.95	0.67	0.78	1.00	0.97	0.98	1.00	0.42	0.59	1.00	0.98	0.99	0.76	0.63	0.69	1.00	1.00	1.00	1.00	1.00	1.00
ma	1.00	0.98	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.98	0.98	0.98	1.00	1.00	1.00	1.00	1.00	1.00
na	0.98	0.95	0.97	1.00	1.00	1.00	0.53	1.00	0.69	1.00	0.95	0.97	0.70	0.67	0.68	1.00	1.00	1.00	0.97	0.97	0.97
nga	0.98	0.98	0.98	1.00	1.00	1.00	0.77	1.00	0.87	1.00	1.00	1.00	0.93	0.93	0.93	1.00	1.00	1.00	1.00	1.00	1.00
nya	0.93	0.95	0.94	0.98	1.00	0.99	1.00	0.65	0.79	1.00	1.00	1.00	0.94	0.98	0.96	1.00	1.00	1.00	1.00	1.00	1.00
pa	0.98	0.98	0.98	0.86	1.00	0.92	0.42	1.00	0.59	1.00	1.00	1.00	0.94	0.82	0.87	1.00	1.00	1.00	1.00	1.00	1.00
ra	0.98	0.85	0.91	1.00	0.75	0.86	0.86	1.00	0.92	1.00	1.00	1.00	0.98	0.87	0.92	1.00	1.00	1.00	1.00	1.00	1.00
sa	0.94	1.00	0.97	1.00	1.00	1.00	0.85	1.00	0.92	0.98	1.00	0.99	0.83	0.97	0.89	1.00	1.00	1.00	1.00	1.00	1.00
ta	0.95	0.60	0.73	1.00	0.98	0.99	1.00	1.00	1.00	1.00	0.60	0.75	0.88	0.47	0.61	1.00	1.00	1.00	1.00	0.67	0.80
tha	1.00	0.93	0.97	1.00	1.00	1.00	1.00	0.70	0.82	1.00	1.00	1.00	0.98	0.90	0.94	1.00	1.00	1.00	1.00	1.00	1.00
wa	0.98	1.00	0.99	1.00	1.00	1.00	0.85	1.00	0.92	1.00	1.00	1.00	0.83	0.97	0.89	1.00	1.00	1.00	1.00	1.00	1.00
ya	0.97	1.00	0.98	1.00	1.00	1.00	0.97	0.47	0.63	1.00	1.00	1.00	0.95	0.97	0.96	1.00	1.00	1.00	1.00	1.00	1.00





**Figure 7:** Accuracy and loss graph.

**Table 2:** Summary of literature review.

Research	Title	Method used	Result
This current research	Evaluating the impact of optimizer hyper-parameters on ResNet in Hanacaraka character recognition	CNN with Resnet-18	High performance with accuracy reaching 97.92 %–100 %, showing NAdam as the best optimizer.
Murinto and Melany (2023)	MobileNetV2 classification of coffee bean types	CNN + transfer learning (MobileNetV2)	Accuracy reaching 96 %, showing improvement over regular CNN models.
Prayinta and Murinto (2021)	Classification of Batik in southern coast area of Java using convolutional neural network method	CNN	Training accuracy of 100 % and testing accuracy of 93.3 %, showing a good model for batik motif classification
Rosyda (2022)	Logarithm decreasing inertia weight particle Swarm optimization algorithms for CNN	CNN + LogDIWPSO	Achieved accuracy of 69.3 %, showing a significant improvement from the baseline.
Lei, Pan, and Huang (2019)	A dilated CNN model for image classification	Backpropagation	Reduced training time by 2.02 % and increased accuracy by 14.15 % in training and 15.35 % in testing on average.

example, a study on coffee bean classification using MobileNetV2 achieved 96 % accuracy while batik motif classification reached 100 % accuracy on training data and 93.3 %

on test data. This indicates that each model and dataset have specific characteristics and results, although the focus on using CNN for classification tasks varies.

## 4 Conclusion and Recommendation

The conclusion of this study is that the choice of optimizer significantly impacts the model's performance in recognizing Javanese Hanacaraka characters. Among the tested optimizers, NAdam proved to be the best, achieving 100 % accuracy, precision, recall, and F1-Score. This highlights NAdam's effectiveness in optimizing the model through the combination of Nesterov Accelerated Gradient (NAG) and momentum adaptation, leading to faster and more stable convergence. Adamax also demonstrated excellent performance, followed by Adam and Adagrad, with all metrics scoring above 97 %.

SGD, although simple, performed reasonably well with an accuracy of 93.72 % but was outperformed by more advanced optimizers like NAdam and Adamax. Adadelat provided adequate performance with stable parameter updates, while RMSprop showed the lowest performance, likely due to its inability to handle large gradient variations effectively for Hanacaraka character recognition.

Overall, this study emphasizes the importance of selecting the appropriate optimizer to achieve optimal performance in Hanacaraka character classification. NAdam and Adamax proved to be the most effective choices while the results from other optimizers provide additional insights into their effectiveness under different training conditions.

**Acknowledgments:** The authors are grateful for the support in facilitating this study.

**Research funding:** This work was supported by Directorate of Research, Technology, and Community Service (DRTPM), Ministry of Education, Culture, Research and Technology, Republic of Indonesia (0609.12/LL5-INT/AL.04/2024, 085/PTM/LPPM-UAD/VI/20).

## References

- Agrawal, A., and N. Mittal. 2020. "Using CNN for Facial Expression Recognition: A Study of the Effects of Kernel Size and Number of Filters on Accuracy." *The Visual Computer* 36 (2): 405–12.
- Ahmed, F., A. Fatima, M. Mamoon, and S. Khan. 2024. "Identification of the Diabetic Retinopathy Using ResNet-18." In *2nd International Conference on Cyber Resilience, ICCR 2024*, 1–6. Dubai, UAE: IEEE.
- Alkaff, A. K., and B. Prasetyo. 2022. "Hyperparameter Optimization on CNN Using Hyperband on Tomato Leaf Disease Classification." In *Proceedings – 2022 IEEE International Conference on Cybernetics and Computational Intelligence, CyberneticsCom 2022*, 479–83. Malang, Indonesia: IEEE.
- Bartz, E., T. Bartz-Beielstein, M. Zaefferer, and O. Mersmann. 2023. *Hyperparameter Tuning for Machine and Deep Learning with R: A Practical Guide*. Singapore: Springer.
- Basha, S. H. S., S. R. Dubey, V. Pulabaigari, and S. Mukherjee. 2020. "Impact of Fully Connected Layers on Performance of Convolutional Neural Networks for Image Classification." *Neurocomputing* 378: 112–9.
- Cahya, T., and M. Murinto. 2021. "Classification of Batik in Southern Coast Area of Java Using Convolutional Neural Network Method." *Jurnal Informatika* 15 (2): 123–30.
- Chandu, N., and N. Bharatha Devi. 2023. "Detection of Plant Disease Using ResNet Framework in Comparison with Neural Network Classifier to Improve Classification Accuracy." In *Proceedings of 8th IEEE International Conference on Science, Technology, Engineering and Mathematics, ICONSTEM 2023*, 1–6. Chennai, India: IEEE.
- Cong, S., and Y. Zhou. 2023. "A Review of Convolutional Neural Network Architectures and Their Optimizations." *Artificial Intelligence Review* 56: 1905–69.
- Das, P., S. Gupta, J. Patra, and B. Mondal. 2023. "ADAMAX Optimizer and CATEGORICAL CROSSENTROPY Loss Function-Based CNN Method for Diagnosing Lung Cancer." In *7th International Conference on Trends in Electronics and Informatics, ICOEI 2023 – Proceedings*, 806–10. Tirunelveli, India: IEEE.
- Duda, J. 2019. "SGD Momentum Optimizer with Step Estimation by Online Parabola Model," *arXiv*, 1–7. <https://doi.org/10.48550/arXiv.1907.07063>
- Han-wen, Z., H. Ying, Z. Yong-jia, and W. Cheng-yu. 2021. "Fingerspelling Identification for American Sign Language Based on Resnet-18." *International Journal of Advanced Networking and Applications* 13 (1): 4816–20.
- Harish, V., T. Vijaya Kumar, P. Rajasekaran, P. Poovizhi, P. Jason Joshua, and R. Sridhar. 2024. "Classification of Early Skin Cancer Prediction Using Nesterov- Accelerated Adaptive Moment Estimation (NADAM) Optimizer Algorithm." In *2024 International Conference on Cognitive Robotics and Intelligent Systems, ICC – ROBINS 2024*, 257–62. Coimbatore, India: IEEE.
- Hasan, M. M., H. Ali, M. F. Hossain, and S. Abujar. 2020. "Preprocessing of Continuous Bengali Speech for Feature Extraction." In *2020 11th International Conference on Computing, Communication and Networking Technologies, ICCCNT 2020*, 1–4. Kharagpur, India: IEEE.
- Kumar Reddy, R., B. Srinivasa Rao, and K. Prudvi Raju. 2018. "Handwritten Hindi Digits Recognition Using Convolutional Neural Network with RMSprop Optimization." In *Proceedings of the 2nd International Conference on Intelligent Computing and Control Systems, ICCCIS 2018*, 45–51. Madurai, India: IEEE.
- Lei, X., H. Pan, and X. Huang. 2019. "A Dilated Cnn Model for Image Classification." *IEEE Access* 7: 124087–95.
- Liu, Y., H. Pu, and D. Wen Sun. 2021. "Efficient Extraction of Deep Image Features Using Convolutional Neural Network (CNN) for Applications in Detecting and Analysing Complex Food Matrices." *Trends in Food Science and Technology* 113 (May): 193–204.
- Lu, Y., W. Ma, X. Dong, M. Brown, T. Lu, and W. Gan. 2023. "Differentiate Xp11.2 Translocation Renal Cell Carcinoma from Computed Tomography Images and Clinical Data with ResNet-18 CNN and XGBoost." *Computer Modeling in Engineering and Sciences* 136 (1): 348–61.
- Madhulatha, G., and O. Ramadevi. 2020. "Recognition of Plant Diseases Using Convolutional Neural Network." In *Proceedings of the 4th International Conference on IoT in Social, Mobile, Analytics and Cloud, ISMAC 2020*, 738–43. Palladam, India: IEEE.
- Mourya, R., and G. Patil. 2024. "Dental Caries Detection through Resnet 50 Using Adam Optimizer." *Iconic Research And Engineering Journals* 7 (8): 203–7.
- Muis, A., S. Sunardi, and A. Yudhana. 2023. "Medical Image Classification of Brain Tumor Using Convolutional Neural Network Algorithm." *Jurnal Infotel* 15 (3): 227–32.

- Murinto, M. R., and M. Melany. 2023. "Classification of Coffee Bean Types Using Convolutional Neural Networks and Transfer Learning on the VGG16 and MobileNetV2 Models." *Jurnal Riset Sains Dan Teknologi* 7 (2): 183–9.
- Nicholas, P. J., A. To, O. Tanglay, I. M. Young, and M. E. Sughrue. 2022. "Using a ResNet-18 Network to Detect Features of Alzheimer's Disease on Functional Magnetic Resonance Imaging: A Failed Replication. Comment on Odusami et Al. Analysis of Features of Alzheimer's Disease." *Diagnostics* 12 (5): 1094.
- Norouzi, S., and M. Ebrahimi. 2019. "A Survey on Proposed Methods to Address Adam Optimizer Deficiencies." Department of Electrical and Computer Engineering, University of Toronto. Available from: [http://www.cs.toronto.edu/~sajadn/sajad\\_norouzi/ECE1505.pdf](http://www.cs.toronto.edu/~sajadn/sajad_norouzi/ECE1505.pdf)
- Prasher, S., L. Nelson, and A. Sharma. 2022. "Analysis of DenseNet201 with SGD Optimizer for Diagnosis of Multiple Rice Leaf Diseases." In *Proceedings – 2022 International Conference on Computational Modelling, Simulation and Optimization, ICCMSO 2022*, 182–6. Pathum Thani, Thailand: IEEE.
- Prayitna, T. C., and M. Murinto. 2021. "Classification of Batik in Southern Coast Area of Java Using Convolutional Neural Network Method." *Jurnal Informatika* 15 (3): 39–46.
- Radha, D., and S. Prasanna. 2024. "A Unique ADAGRAD Optimized DCNN with RESNET-18 Architecture for Indoor Agriculture-Based Crop Yield." In *Proceedings – International Conference on Computing, Power, and Communication Technologies, IC2PCT 2024*, 767–71. Greater Noida, India: IEEE.
- Riadi, I., A. Yudhana, and M. Rosyidi Djou. 2024. "Optimization of Population Document Services in Villages Using Naive Bayes and K-NN Method." *International Journal of Computing and Digital Systems* 15 (1): 127–38.
- Rosyda, M. 2022. "Logarithm Decreasing Inertia Weight Particle Swarm Optimization Algorithms for Convolutional Neural Network." *Juita* 10 (1): 99–105.
- Roy, S., R. Mehera, R. K. Pal, and S. K. Bandyopadhyay. 2023. "Hyperparameter Optimization for Deep Neural Network Models: A Comprehensive Study on Methods and Techniques." *Innovations in Systems and Software Engineering*. <https://doi.org/10.1007/s11334-023-00540-3>.
- Shadin, N. S., S. Sanjana, and N. J. Lisa. 2021. "COVID-19 Diagnosis from Chest X-Ray Images Using Convolutional Neural Network(CNN) and InceptionV3." In *2021 International Conference on Information Technology, ICIT 2021 – Proceedings*, Vol. 3, 799–804.
- Singh, S., and D. Schicker. 2021. "Seven Basic Expression Recognition Using ResNet-18," 1–3. Available from: <http://arxiv.org/abs/2107.04569>.
- Sudewo, E. D. B., M. K. Biddinika, and A. Fadlil. 2024a. "DenseNet Architecture for Efficient and Accurate Recognition of Javanese Script Hanacaraka Character." *MATRIK: Jurnal Manajemen, Teknik Informatika Dan Rekayasa Komputer* 23 (2): 453–64.
- Sudewo, E. D. B., M. K. Biddinika, and A. Fadlil. 2024b. "Javanese Script Hanacaraka Character Prediction with ResNet-18 Architecture." *Jurteks* 10 (2): 401–8.
- Sveleba, S., I. Katerynychuk, I. Kunyo, V. Brygilevych, V. Kotsun, and Yu Sidelnyk. 2023. "Dynamics of the Learning Process of a Multilayer Neural Network when Using the AdaDelta Optimization Method." In *2023 24th International Conference on Computational Problems of Electrical Engineering, CPEE 2023*, 1–4. Grybów, Poland: IEEE.
- Yaqub, M., F. Jinchao, M. Sultan Zia, K. Arshid, K. Jia, Z. Ur Rehman, and A. Mehmood. 2020. "State-of-the-Art CNN Optimizer for Brain Tumor Segmentation in Magnetic Resonance Images." *Brain Sciences* 10 (7): 1–19.