**Accredited SINTA 2 Ranking**
Decree of the Director General of Higher Education, Research, and Technology, No. 158/E/KPT/2021
Validity period from Volume 5 Number 2 of 2021 to Volume 10 Number 1 of 2026

Published online at: **http://jurnal.iaii.or.id**

**JURNAL RESTI**
**(Rekayasa Sistem dan Teknologi Informasi)**
Vol. 8 No. 6 (2024) 806 - 818 e-ISSN: 2580-0760

# Efficient Pattern Recognition of Sundanese Script Variants Using CNN

Muhammad Husni Wahid[1*], Erik Iman Heri Ujianto[2]
[1]Informatics Study Program, Faculty of Science and Technology, Universitas Teknologi Yogyakarta, Sleman, Indonesia
[2]Master of Information Technology, Universitas Teknologi Yogyakarta, Sleman, Indonesia
[1]husniwahid887@gmail.com, erik.iman@uty.ac.id [2]

***Abstract***

*This research aims to apply pattern recognition technology, specifically through the Convolutional Neural Network (CNN) approach, in identifying and translating Sundanese script accurately. This research is focused on recognizing rarangken script patterns based on ngalagena script in Indonesian cultural heritage. This study uses the MobileNetV2 based CNN model, utilizing transfer learning and trained for 50 epochs using the Adam optimizer with a learning rate of 0.0001, to achieve a training accuracy of 98.75% and test accuracy of 96.95% in 1 hour and 23 minutes, respectively. The results of the study show that the simpler CNN architecture without augmentation achieved the highest accuracy of 99.26%, and the augmented CNN model achieved 94.42% accuracy in 2 hours and 22 minutes. These results enable practical applications in both education and cultural preservation, demonstrating how modern technology can effectively contribute to maintaining traditional cultural elements in the digital era.*

*Keywords: CNN, MobileNetV2, Pattern Recognition, Sundanese Script*

*How to Cite:* Muhammad Husni Wahid and Erik Iman Heri Ujianto, "Efficient Pattern Recognition of Sundanese Script Variants Using CNN", *J. RESTI (Rekayasa Sist. Teknol. Inf.)* , vol. 8, no. 6, pp. 808 - 818, Dec. 2024.
*DOI*: https://doi.org/10.29207/resti.v8i6.6122

## 1. Introduction

Sundanese script is one of Indonesia's cultural heritages that has a long and significant history, used as a writer of Sundanese language since the 5th century AD [1]. It is a structured script where consonants inherently carry the vowel sound /a/, modified by diacritics, which sets it apart from alphabetic scripts. Each character combines consonants and vowels into syllables, with key components including consonants (ngalagena), vowels (swara), and modifiers (rarangken) that contribute to its complexity [2]. The introduction of Sundanese script patterns is an important foundation in exploring the richness of culture in Indonesia. Swara script, ngalagena script, special script, rarangken, and pairs are the several groupings that make up Sundanese script [3]. In the context of current technological advances, it is important to apply Sundanese script pattern recognition to preserve and empower this cultural wealth. However, the preservation of this traditional script faces challenges, such as the lack of public understanding and the lack of resources. Therefore, this research aims to apply pattern recognition technology, specifically through the CNN method, in identifying and translating Sundanese Script accurately. CNN are well-suited for this task due to their proven ability to handle complex image classification problems effectively. CNN work through hierarchical feature extraction, learning to detect edges, textures, and patterns that are crucial for distinguishing characters in challenging scripts like Sundanese. Unlike traditional machine learning methods, CNN do not rely on handcrafted features but instead learn features directly from data. This makes them particularly effective for diverse handwriting recognition scenarios [4].

Similar research has previously been conducted by Alif et al., who examined Sundanese script pattern recognition using the CNN method. The study utilized image data sizes ranging from 128 x 128 to 300 x 300 pixels, divided into 2,325 files for training and validation. It reported success rates of 72.4% for e-books, 100% for computer fonts, 87.1% for mobile phone cameras, and 85.5% for scanners [3]. Another study by Aldo et al. employed the CNN method with the VGG-16 architecture, achieving the best accuracy of 97.6% using the Adam optimizer with a learning rate of 0.0001, while the SGD optimizer produced the worst results with a maximum accuracy of only 36.4% [5].

Received: 19-10-2024 | Accepted: 22-12-2024 | Published Online: 28-12-2024

Research by Rahmawati et al. on Sundanese script recognition using the CNN method achieved an accuracy of 98.03% with 500 epochs, a learning rate of 0.001, and the Adam optimizer. These findings demonstrate that CNN outperforms other methods for Sundanese script recognition [6].

Based on this description, this research is focused on recognizing rarangken script patterns based on ngalagena script in Sundanese Script. This study classifies the rarangken script using the Convolutional Neural Network (CNN) approach, which involves the letter "ka" and 13 other rarangken scripts. The performance of the *CNN* model in recognizing rarangken script patterns will be analyzed to determine how effective this method is in script recognition compared to existing conventional methods. It is anticipated that this study will aid in the preservation of Sundanese script and strengthen local cultural identity.

## 2. Research Methods

The stages of this research can be seen in Figure 1, which shows 5 stages carried out, specifically the CNN model, data collection, data pre-processing, model evaluation, and analysis outcomes.

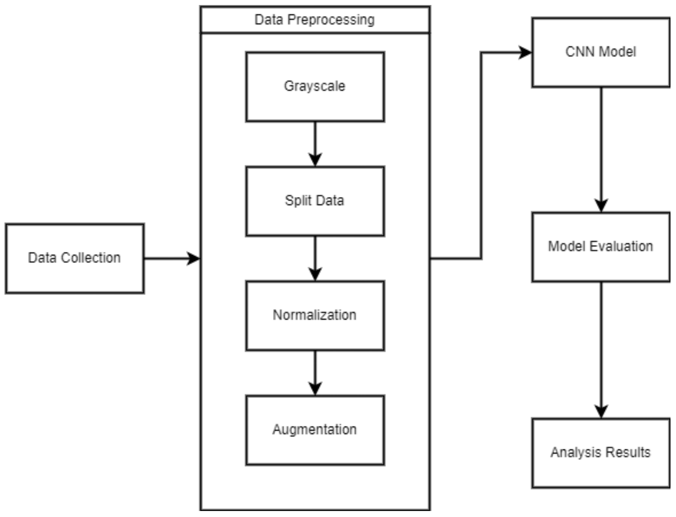

Figure 1. Research Phase

### 2.1. Data Collection

In this research, an important first step was the collection of handwritten data that would be used for analysis. Rarangken is a vocalization marker in ancient Sundanese script that consists of 13 forms: 5 forms above the base character, 2 forms below, and 6 parallel forms as shown in Figure 2. Meanwhile, Ngalagena, also an ancient Sundanese script, symbolizes consonant phonemes with the vowel /a/ and consists of 25 characters resulting from the placement of the speech organ [7]. The focus of this research is on the "Ka" script, which is used as example data.

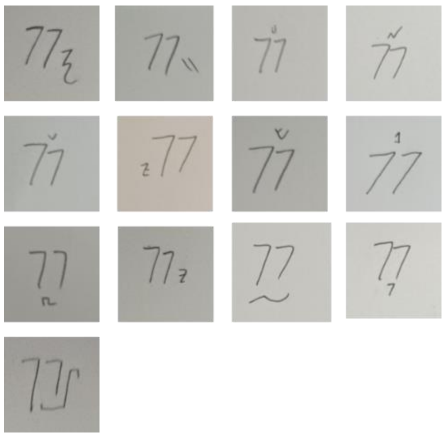

Figure 2. Example of Sundanese Script Data from 13 classes

The research began by collecting handwritten data in the form of rarangken script combined with ngalagena script "Ka" on paper. Each script was written manually to produce variations in the handwritten form as shown in Figure 3

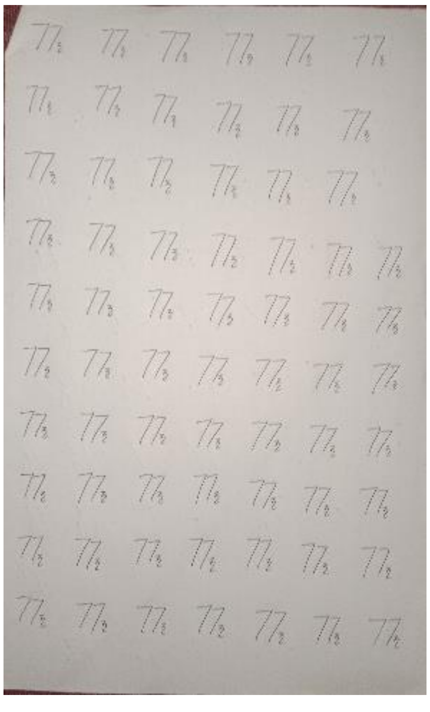

Figure 3. Sundanese Script Data

Next, the paper containing the script was photographed, and the photo was manually cropped to a size of 300x300 pixels to separate each script. After that, each script was labeled according to its class. The collected data was grouped into 13 classes, with each class containing 500 samples, so the total amount of data reached 6,500 samples can be seen in Figure 4.

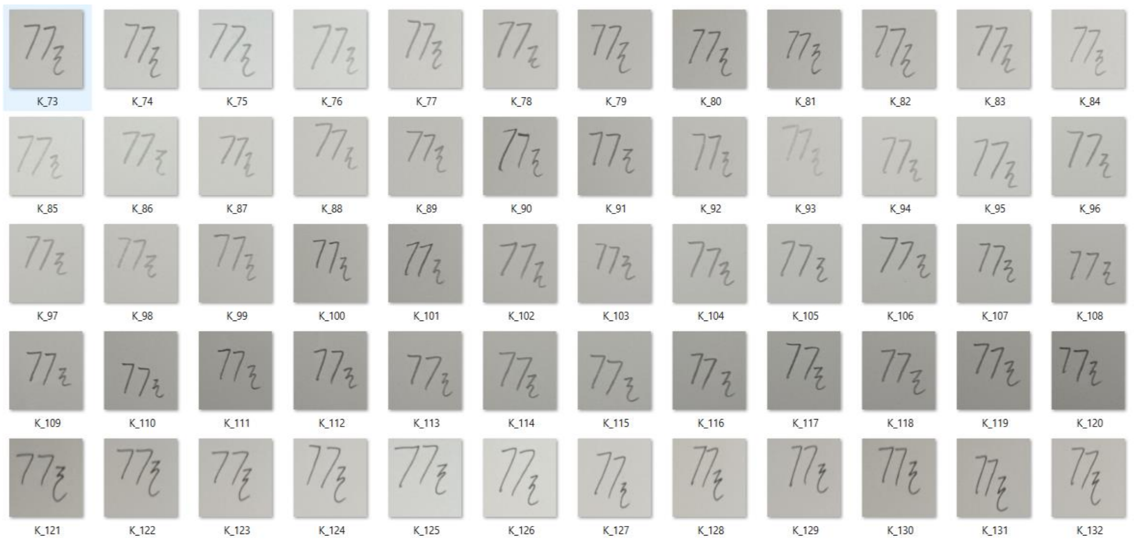

Figure 4. After Cropping

This dataset was compiled to facilitate further analysis, and has been stored on GitHub for other researchers to access via the following link: https://github.com/Hbehong/Sunda, thereby enabling repeat research.

To enhance data diversity, selected images underwent manual augmentation with random variations. The augmentation process included four types of variations: thickness adjustment using random threshold values 127, 150 or 170, size modification with random scaling factors between 0.8 and 1.5, rotation with random angles between -20 and 20 degrees, and brightness adjustment with random factors between 0.7 and 1.5. These augmented images help improve the model's robustness to different writing styles and conditions and the result can be seen in Figure 5 and Table 1.

Table 1. Data Count

| Class | Data Count |
|---|---|
| K | 580 |
| Kah | 580 |
| Kang | 580 |
| Kar | 580 |
| To | 580 |
| Ké | 580 |
| Keu | 580 |
| Ki | 580 |
| Kla | 580 |
| Ko | 580 |
| Kra | 580 |
| Ku | 580 |
| Kya | 580 |
| Total | 7540 |

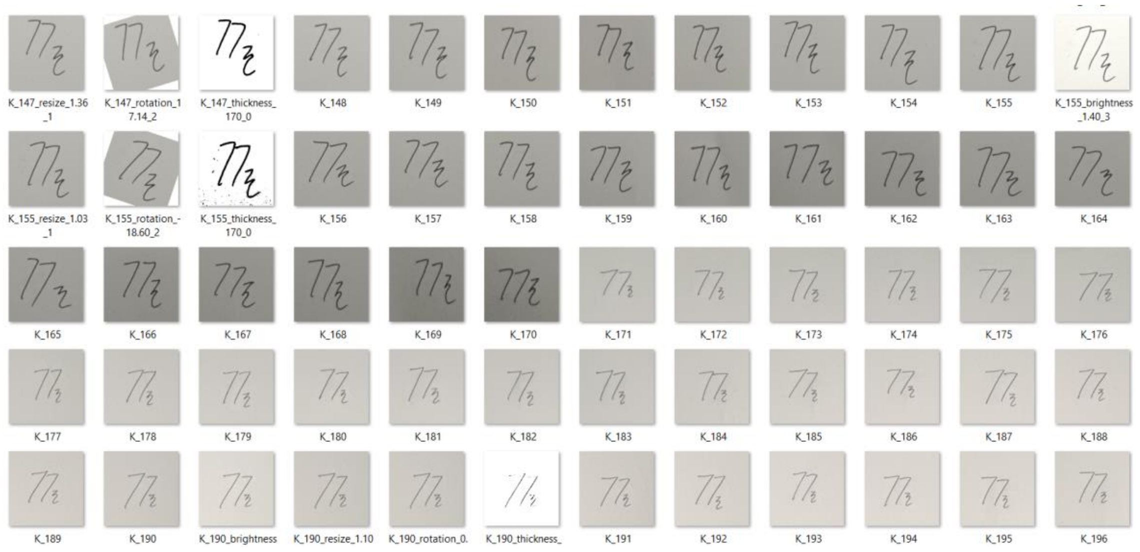

Figure 5. After Add Variation

### *2.2. Data Pre-processing*

The dataset that has been collected is then forwarded to the data preprocessing stage. Data preprocessing is carried out with the aim of making it easier for models to process data and improve data quality [8]. In the first stage, the Sunda Script image dataset was converted to grayscale format as a crucial first step in data pre-processing. The grayscale format reduces the complexity of the data because each pixel of the image has only one color channel (Figure 6), unlike a color image which has three channels (RGB) (Figure 7). With this conversion, CNN models can focus more on the shape and texture of the script, which is important for pattern recognition. In addition, the conversion to grayscale helps to simplify the image, increase the training speed, and simplify the implementation of the model [9].

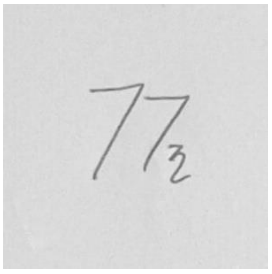
Figure 6. Grayscale

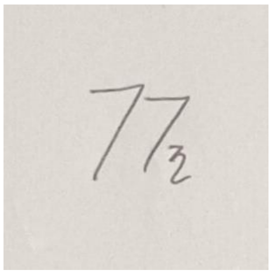
Figure 7. RGB

The Sundanese Script dataset is split into three sections: test data makes up 10%, validation data makes up 10%, and training data makes up 80% as shown in Table 2 and Figure 8. This division is important to ensure the model can be trained and evaluated accurately. A total of 80% of the dataset, which is 6.032 images, is used to train the model to recognize patterns from the data. The validation data, which consists of 10% or 754 images, is employed to evaluate the model's performance during training and help prevent overfitting. Meanwhile, the remaining 10% or 754 images are used as test data to evaluate the model's performance on new data that has never been seen before, so that the identification results can be compared with the training and validation data [10].

To increase the variety of data and employ data augmentation strategies to improve the model's capacity for generalization. Prior to augmentation, the dataset is resized to 128x128 pixels to ensure the CNN model can process the data more efficiently [11]. Augmentation aims to create variations of the original dataset by making random changes to the image without changing the original label. This method is an effective way to enrich the variety of features [12]. The augmentation techniques used include rotation by 20%, width and height shift by 20%, shear by 20%, and zoom in a range of 20% as shown in Figure 9. Empty areas that appear due to augmentation are filled using the nearest method, which ensures that they are filled with the corresponding values from the nearest area in the image. Before augmentation, normalization was performed by setting the pixel intensity scale to a range between 0 and 1 to avoid imbalance in the scale of pixel values that could affect the performance of the model. This augmentation is effective in enriching the variety of features and helps the model recognize patterns in the new data. Additionally, data augmentation provides a cost-effective way to generate additional data for CNN models without requiring the expense of labeling new data [13].

Table 2. Split Data Count

| Class | Training | Validation | Testing |
|---|---|---|---|
| K | 464 | 58 | 58 |
| Kah | 464 | 58 | 58 |
| Kang | 464 | 58 | 58 |
| Kar | 464 | 58 | 58 |
| To | 464 | 58 | 58 |
| Ké | 464 | 58 | 58 |
| Keu | 464 | 58 | 58 |
| Ki | 464 | 58 | 58 |
| Kla | 464 | 58 | 58 |
| Ko | 464 | 58 | 58 |
| Kra | 464 | 58 | 58 |
| Ku | 464 | 58 | 58 |
| Kya | 464 | 58 | 58 |
| Total | 6032 | 754 | 754 |

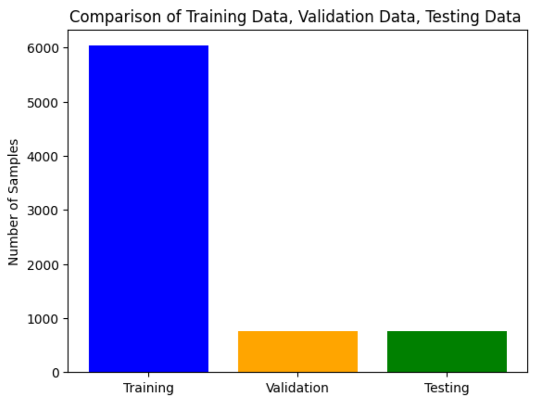

Figure 6. Split Data

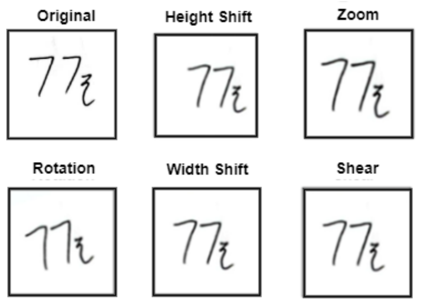

Figure 7. Augmentation Data

### *2.3. CNN Model*

At this stage, the architecture of the CNN model in Figure 10 is created to handle image data sized 128x128 pixels with a single grayscale channel. In the initial stage, the input image passes through the first convolutional layer, which is equipped with 32 filters and a 3x3 kernel. The main function of this layer is to extract key features from the image, such as lines, textures, or edges that define the object in the image. Each filter plays a role in detecting specific patterns in different parts of the image. Afterward, this convolutional layer's output is passed to the pooling layer [14]. The pooling layer works by reducing the size or dimensionality of the image while retaining essential features. By performing downsampling with a pool size of 2x2, the image size is significantly reduced, but crucial information needed for classification is preserved.

After several passes through convolutional and pooling layers, the processed image is then forwarded to the Flatten layer. The multidimensional output is transformed using the Flatten layer from the previous layer into a one-dimensional format. This transformation is important because the data produced by the convolution process must be flattened before it can be further processed by the Dense layers [15]. Dense layers are completely interconnected layers in which every neuron is linked to every other neuron in the layer above it. In this architecture, there are two Dense layers with 256 and 512 units, in that order. The activation function for the Rectified Linear Unit (ReLU) is used in both layers to handle non-linearity by transforming negative values into zeros, helping the model to learn more complex patterns from the data [16].

Additionally, to prevent the issue of overfitting, which often occurs when the model becomes too "memorized" with the training data, a Dropout layer with a rate of 0.5 is applied. This Dropout works by randomly ignoring half of the neurons during the training process, aiming to reduce excessive dependence on certain neurons and improve the model's generalization on unseen data [17]. The implementation of dropout is an important technique to enhance the model's performance on validation or test data.

After passing through all the Dense layers, the output is finally forwarded to the last layer, the output layer. In this layer, the model uses the softmax activation function. Softmax plays a role in calculating the probability of each target class, where in this case, there are 13 different classes. Softmax produces a probability value between 0 and 1 for each class, and the total sum of these values will be 1. Thus, the model can predict which class is most likely based on the given input image [18].

In addition to the CNN Sequential architecture, another model used is MobileNetV2, a Convolutional Neural Network (CNN) architecture known for its lightweight and efficient design, making it suitable for image classification tasks in resource-constrained environments. Like other CNNs, MobileNetV2 extracts hierarchical features to detect patterns such as edges and textures, which are crucial for recognizing Sundanese script. By adopting the concepts of a linear bottleneck and inverted residual structure, MobileNetV2 efficiently extracts features with low computational complexity, enabling high accuracy with reduced resource usage [19]. In this study, MobileNetV2 was utilized with two different learning rates: 0.001 and 0.0001. This approach allowed comparison of the model's performance with different learning rates, evaluating its efficiency and accuracy in the context of Sundanese script recognition.

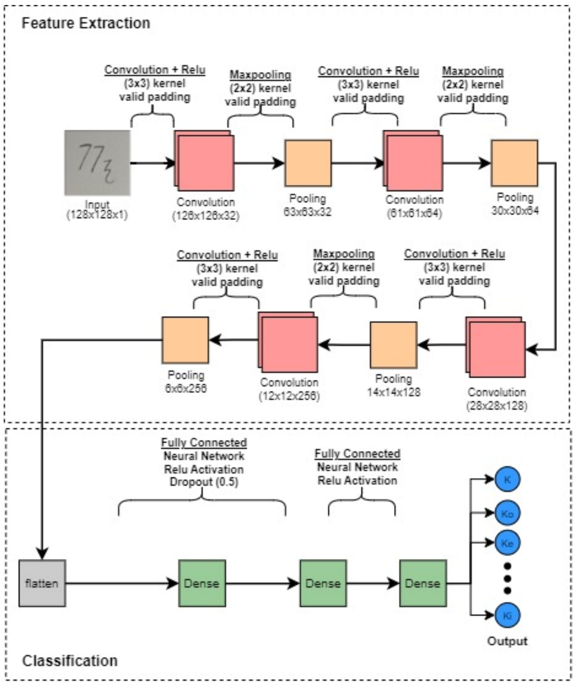

Figure 8. Sequential CNN Model

Table 3. Sequential CNN Model

| Layer | Parameters |
|---|---|
| Image input | 128x128x1 |
| Conv2D + Relu | Filters: 32, Size: (3x3), Padding: valid |
| MaxPooling2D | Size: (2x2), Strides: (2,2), Padding: valid |
| Conv2D + Relu | Filter: 64, Size: (3x3), Padding: valid |
| MaxPooling2D | Size: (2x2), Strides: (2,2), Padding: valid |
| Conv2D + Relu | Filter: 128, Size: (3x3), Padding: valid |
| MaxPooling2D | Size: (2x2), Strides: (2,2), Padding: valid |
| Conv2D + Relu | Filter: 256, Size: (3x3), Padding: valid |
| MaxPooling2D | Size: (2x2), Strides: (2,2), Padding: valid |
| Flatten | - |
| Dense + Relu | Units: 512 |
| Dropout | Rate: 0.5 |
| Dense + Relu | Units: 256 |
| Dense + Softmax | Units: 13 |

Table 3 shows the layers that make up the CNN model's design in this investigation. Each layer has an important role in the process of feature extraction and image classification, beginning with the input layer and moving on to the output layer that produces classification probabilities for each class. The design of each stage in this architecture aims to optimize the model's ability to recognize patterns in images, so that the model can function effectively and efficiently in pattern recognition applications.

The CNN architecture in this study was designed based on previous approaches, with adaptations for Sundanese script recognition. The sequential CNN model with convolutional and pooling layers was chosen based on the research by Rahmawati et al., which achieved an accuracy of 98.03% [6]. This architecture was modified with additional convolutional layers to handle variations of *rarangken*. The input size of 128x128 was chosen for a balance between detail and computational efficiency, reducing the computational load by approximately 75% compared to the 300x300 size. The convolutional layer configuration uses 32 filters with 3x3 kernels, progressively increasing in number, following the VGG-style architecture, with consistent 3x3 kernels for optimal local feature detection. MaxPooling 2x2 is used to reduce spatial dimensions and maintain a consistent downsampling ratio, which is crucial for handling variations in handwritten characters.

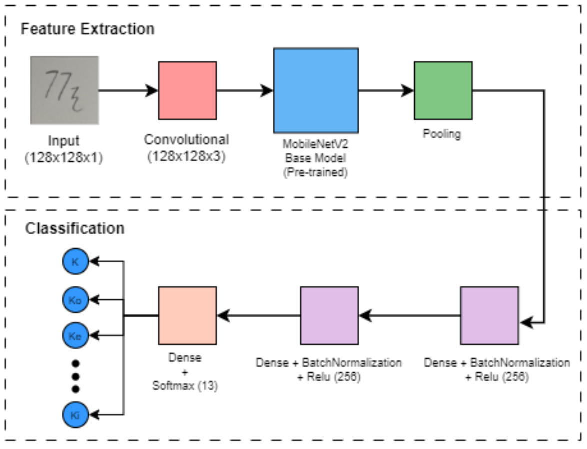
Feature Extraction
Input (128x128x1)
Convolutional (128x128x3)
MobileNetV2 Base Model (Pre-trained)
Pooling
Classification
K
Ka
Ke
Ki
Dense + Softmax (13)
Dense + BatchNormalization + Relu (256)
Dense + BatchNormalization + Relu (256)

Figure 9. CNN Model with Transfer Learning Using MobileNetV2 Architecture

Table 4 and Figure 11 shows the architecture of the MobileNetV2-based CNN model designed for this study. The model begins with a grayscale input layer 128x128x1 followed by a Conv2D layer that converts the input to RGB format 3 channels required by MobileNetV2. The pre-trained MobileNetV2 base model, initialized with ImageNet weights and excluding the top classification layers (include_top=False), serves as an efficient feature extractor. The GlobalAveragePooling2D layer reduces spatial dimensions while preserving essential features. The model then employs two Dense layers with ReLU activation 256 and 128 units for feature learning, with a Dropout layer rate 0.4 between them to prevent overfitting. Finally, a Dense layer with 13 units and Softmax activation produces classification probabilities for each character class. This architecture combines the efficiency of MobileNetV2 with custom classification layers to effectively recognize Sundanese script patterns.

Table 4. CNN Model with Transfer Learning Using MobileNetV2 Architecture

| Layer | Parameters |
|---|---|
| Image input | 128x128x1 |
| Conv2D | Filters: 3, Size: (1x1), Padding: valid |
| MobileNetV2 Base Model | Weights: imagenet, include_top=False |
| GlobalAveragePooling2D | - |
| Dense + BatchNormalization + Relu | Units: 512 |
| Dropout | Rate: 0.5 |
| Dense + BatchNormalization + Relu | Units: 256 |
| Dense + Softmax | Unit 13 |

### 2.4. Model Evaluation

Model Evaluation needs to be done which aims to determine the best model after model training. [20]. A confusion matrix is one tool used to assess a classification model's performance. A confusion matrix is a table that displays how many test results were correctly and incorrectly labeled [21]. False Positive (FP), False Negative (FN), True Positive (TP), and True Negative (TN) are some of the key concepts used in the confusion matrix [22].

Each model scenario's evaluation outcomes will yield accuracy, precision, recall, and F1-score values, which together give a complete picture of how well the model performed during the classification process [23].

### 2.5. Results and Analysis

The final stage of this research is to analyze the performance of the Sundanese script pattern recognition classification model using the *CNN* model. Model performance is assessed based on *confusion* matrix [24].

The model's classification accuracy of the test data will be displayed in the confusion matrix, as well as help identify the number of correct and incorrect predictions for each Sundanese script category.

## 3. Results and Discussions

### 3.1. Training Loss and Accuracy

An examination of the training data reveals a notable improvement in the model's accuracy, starting from 0.5265 at epoch 10 and reaching 0.9442 at epoch 100 as shown in Table 5. This is accompanied by a significant decrease in the loss value, which initially stood at 1.3713 and dropped to 0.1539 at epoch 100. In the validation data, the validation accuracy increased from 0.7269 at epoch 10 to 0.9946 at epoch 100, while the validation loss decreased from 0.7820 to 0.0212. This development shows that the model achieves not only high accuracy but also a strong generalization ability, as indicated by the performance on unseen validation data. Figures 12 and 13 present visualizations of these training results. Figure 12 shows a consistent increase in accuracy over the epochs, while Figure 13 illustrates a significant decrease in loss values. These results reflect the efficiency and robustness of the Sequential model with data augmentation in accurately recognizing patterns within the dataset.

Table 5. Training Accuracy of Sequential Model with Augmentation

| Epoch | Accuracy | Loss | Val Accuracy | Val Loss |
|---|---|---|---|---|
| 10 | 0.5265 | 1.3713 | 0.7269 | 0.7820 |
| 20 | 0.7212 | 0.7802 | 0.8505 | 0.4354 |
| 30 | 0.7842 | 0.5869 | 0.8546 | 0.3700 |
| 40 | 0.8335 | 0.4660 | 0.9361 | 0.2119 |
| 50 | 0.8667 | 0.3743 | 0.9470 | 0.1376 |
| 60 | 0.8975 | 0.2835 | 0.9715 | 0.0808 |
| 70 | 0.9162 | 0.2425 | 0.9905 | 0.0444 |
| 80 | 0.9315 | 0.2046 | 0.9878 | 0.0321 |
| 90 | 0.9335 | 0.1837 | 0.9918 | 0.0215 |
| 100 | 0.9442 | 0.1539 | 0.9946 | 0.0212 |

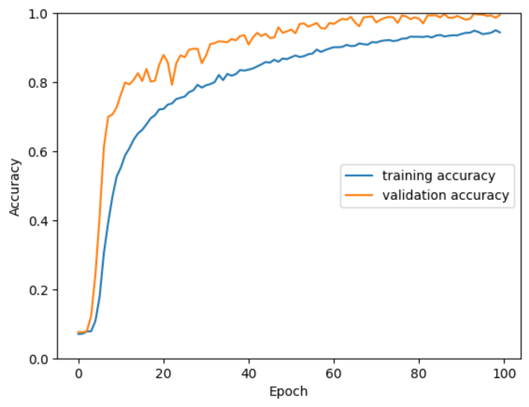

Figure 10. Accuracy Graph of Sequential Model With Augmentation

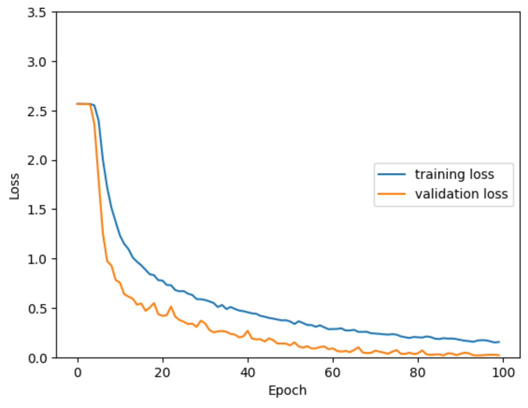

Figure 11. Loss Graph of Sequential Model With Augmentation

An analysis of the training data reveals a steady improvement in the model's performance over time. Starting with an accuracy of 0.8237 at epoch 10, the accuracy increased to 0.9966 by epoch 100. This was accompanied by a significant reduction in the loss value, which decreased from 0.4920 at epoch 10 to 0.0166 at epoch 100. For the validation data, the validation accuracy improved from 0.8344 at epoch 10 to 0.9406 at epoch 100, while the validation loss dropped from 0.4818 to 0.2386. The training data results show significant improvement in the model's performance. At epoch 10, the accuracy was 0.8237 with a loss of 0.4920, and by epoch 100, accuracy increased to 0.9966, with a reduced loss of 0.0166. Validation data also improved, with accuracy rising from 0.8344 to 0.9406, and loss decreasing from 0.4818 to 0.2386. These improvements indicate the model's ability to effectively learn data patterns without overfitting and to generalize well to unseen data. Figures 14 and 15 show the accuracy and loss trends throughout the training process. Figure 14 illustrates the consistent increase in training and validation accuracy, while Figure 15 highlights the steady decrease in loss values. These results underscore the efficiency and robustness of the Sequential model in recognizing data patterns without augmentation as shown in Table 6.

Table 6. Training Accuracy of Sequential Model Without Augmentation

| Epoch | Accuracy | Loss | Val Accuracy | Val Loss |
|---|---|---|---|---|
| 10 | 0.8237 | 0.4920 | 0.8344 | 0.4818 |
| 20 | 0.9017 | 0.2682 | 0.8844 | 0.3165 |
| 30 | 0.9443 | 0.1535 | 0.8938 | 0.3001 |
| 40 | 0.9675 | 0.0955 | 0.9094 | 0.2647 |
| 50 | 0.9805 | 0.0584 | 0.9156 | 0.2907 |
| 60 | 0.9845 | 0.0477 | 0.9187 | 0.2477 |
| 70 | 0.9874 | 0.0360 | 0.9281 | 0.2510 |
| 80 | 0.9926 | 0.0216 | 0.9375 | 0.2829 |
| 90 | 0.9950 | 0.0167 | 0.9344 | 0.2831 |
| 100 | 0.9966 | 0.0166 | 0.9406 | 0.2386 |

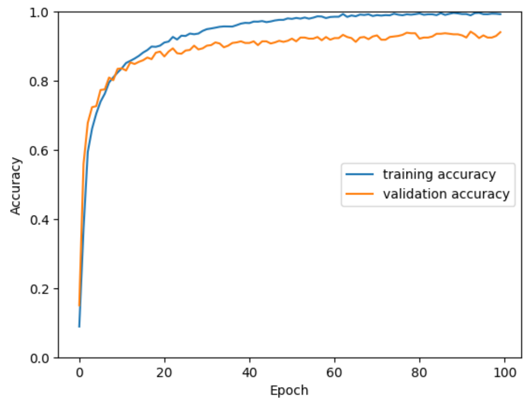

Figure 12. Accuracy Graph of Sequential Model Without Augmentation

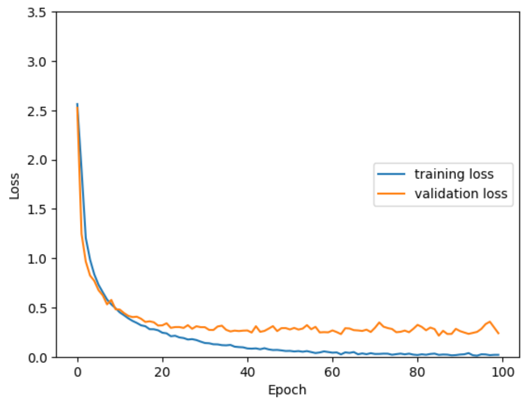

Figure 13. Loss Graph of Sequential Model Without Augmentation

From the training results using the MobileNetV2 architecture with a learning rate of 0.0001, a significant improvement in the model's accuracy is observed as the epochs progress as shown in Table 7.

Table 7. Training Accuracy of MobileNetV2 with Transfer Learning (Learning Rates: 0.0001)

| Epoch | Accuracy | Loss | Val Accuracy | Val Loss |
|---|---|---|---|---|
| 10 | 0.9603 | 0.1520 | 0.9701 | 0.1337 |
| 20 | 0.9628 | 0.0607 | 0.9715 | 0.0969 |
| 30 | 0.9738 | 0.0298 | 0.9755 | 0.0871 |
| 40 | 0.9868 | 0.0179 | 0.9764 | 0.0773 |
| 50 | 0.9875 | 0.0129 | 0.9701 | 0.0871 |

At epoch 10, the training accuracy was 0.9603, with a loss value of 0.1520. This trend continued, with epoch 20 showing a training accuracy of 0.9628, and the loss decreasing to 0.0607. By epoch 40, the training accuracy reached 0.9868, and the loss was further reduced to 0.0179. The validation data also shows consistent improvements, indicating strong generalization capabilities. Validation accuracy started at 0.9701 with a loss of 0.1337 at epoch 10 and increased to 0.9764 with a loss of 0.0773 by epoch 40. These improvements demonstrate the model's robustness in handling unseen data.

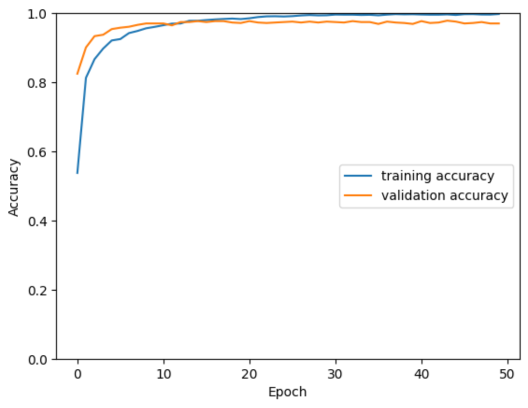

Figure 14. Accuracy Graph of MobileNetV2 with Transfer Learning (Learning Rates: 0.0001)

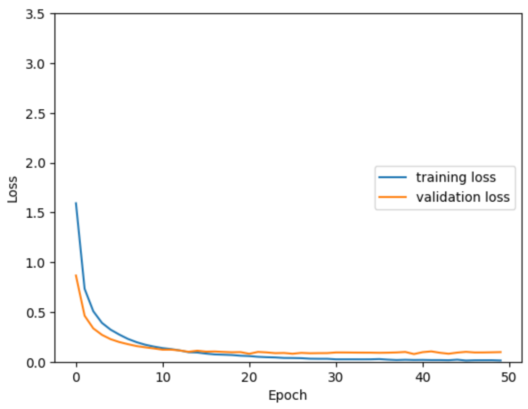

Figure 15. Loss Graph of MobileNetV2 with Transfer Learning (Learning Rates: 0.0001)

Figure 16 illustrates the accuracy graph for both training and validation data. This graph shows consistent improvement in accuracy for both datasets, indicating that the model is learning effectively from the training data while maintaining performance on the validation data. Figure 17 shows the loss graph for both training and validation. This graph illustrates the steady decrease in loss for both datasets, indicating the reduction of prediction errors as the epochs progress.

From the training results using the MobileNetV2 architecture with a learning rate of 0.001, a clear improvement in the model's accuracy is seen as training progresses as shown in Table 8. At epoch 10, the training accuracy was 0.9712, with a loss value of 0.0900. As the epochs continued, epoch 20 showed a training accuracy of 0.9852, with the loss decreasing to 0.0592. By epoch 50, the training accuracy had reached 0.9920, with a loss value of 0.0248.

Table 8. Training Accuracy of MobileNetV2 with Transfer Learning (Learning Rates: 0.001)

| Epoch | Accuracy | Loss | Val Accuracy | Val Loss |
|---|---|---|---|---|
| 10 | 0.9712 | 0.0900 | 0.9674 | 0.0893 |
| 20 | 0.9852 | 0.0592 | 0.9742 | 0.0867 |
| 30 | 0.9863 | 0.0303 | 0.9715 | 0.0909 |
| 40 | 0.9893 | 0.0293 | 0.9674 | 0.0933 |
| 50 | 0.9920 | 0.0248 | 0.9701 | 0.0909 |

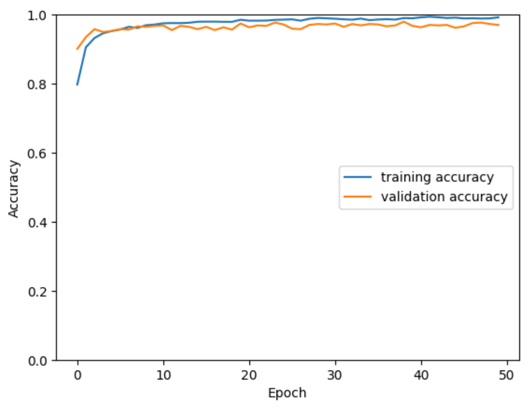

Figure 16. Accuracy Graph of MobileNetV2 with Transfer Learning (Learning Rates: 0.001)

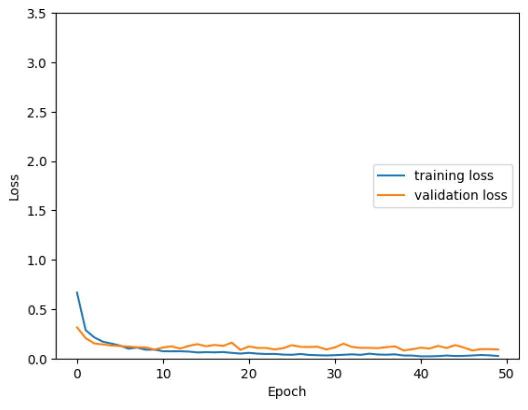

Figure 17. Loss Graph of MobileNetV2 with Transfer Learning (Learning Rates: 0.001)

The validation data also shows consistent improvements, reflecting the model's strong ability to generalize. At epoch 10, the validation accuracy was 0.9674 with a loss of 0.0893, and by epoch 50, the validation accuracy was 0.9701 with a loss of 0.0909.

Figure 18 presents the accuracy graph for both training and validation data. The graph clearly shows the increase in accuracy for both datasets, suggesting that the model is learning well and generalizing effectively to unseen data. Figure 19 shows the loss graph for both training and validation. The loss decreases steadily, indicating that the model is reducing errors over time, further supporting the effectiveness of the model and the chosen learning rate.

### *3.2. Model Evaluation*

The accuracy, precision, recall, and f1-score values for the model can be observed in the classification report shown in Table 9. The Sequential Model with Augmentation demonstrates excellent performance across most classes, with many achieving a perfect f1-score, precision, and recall of 1.00. For instance, the "Kya" and "Kra" classes achieved flawless scores across all metrics, indicating ideal classification. The "Keu" class recorded a precision of 0.95, recall of 1.00, and an f1-score of 0.97, showcasing strong yet slightly varied performance. Similarly, the "Ké" class achieved a precision of 1.00, recall of 0.98, and an f1-score of 0.99. Meanwhile, the "Ke" class also performed exceptionally well, with both precision and f1-score at 1.00, and a recall of 0.98 as shown in Figure 20.

Table 9. Classification Report of Sequential Model With Augmentation

| Class | Precission | Recall | F1-Score |
|---|---|---|---|
| K | 0.98 | 1.00 | 0.99 |
| Kah | 0.98 | 1.00 | 0.99 |
| Kang | 0.98 | 0.93 | 0.96 |
| Kar | 1.00 | 0.90 | 0.95 |
| Ke | 0.90 | 0.98 | 0.94 |
| Ké | 1.00 | 0.98 | 0.99 |
| Keu | 0.95 | 1.00 | 0.97 |
| Ki | 0.95 | 1.00 | 0.97 |
| Kla | 1.00 | 0.98 | 0.99 |
| Ko | 1.00 | 0.98 | 0.99 |
| Kra | 1.00 | 1.00 | 1.00 |
| Ku | 1.00 | 0.98 | 0.99 |
| Kya | 1.00 | 1.00 | 1.00 |

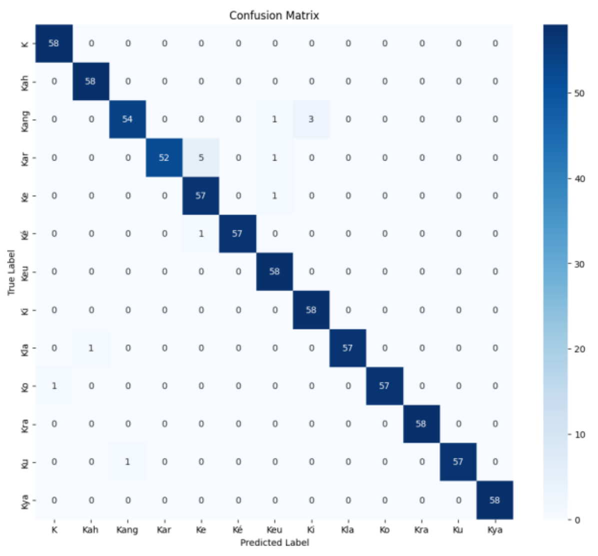

Figure 18. Confusion Matrix of Sequential Model With Augmentation

These results highlight the model's ability to classify effectively across a wide range of classes, showcasing robustness and reliability. This performance reflects the effectiveness of the CNN architecture combined with data augmentation in accurately recognizing and classifying complex patterns.

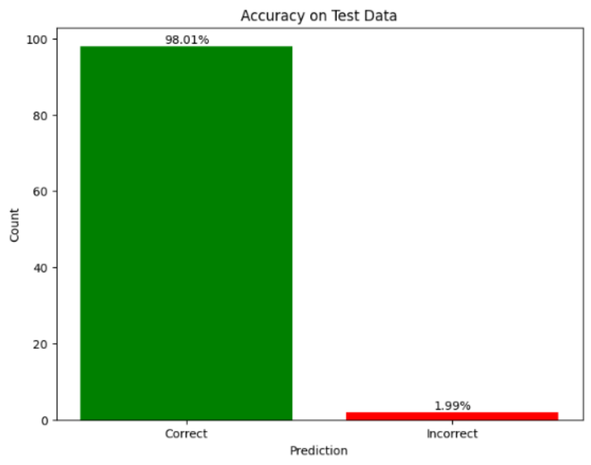

Figure 19. Comparison of Correct and Incorrect Test Data Using a Sequential Model With Augmentation

The Sequential model with augmentation, trained for 100 epochs using the Adam optimizer, achieved impressive results. The training accuracy reached 94.42%, with a significant reduction in the loss value to 1.539. The training process lasted approximately 2 hours and 22 minutes. During evaluation using the confusion matrix, the model demonstrated a test accuracy of 98.01%, along with excellent recall, precision, and f1-scores across all classes as shown in Figure 21.

The accuracy, precision, recall, and f1-score values for the model can be derived from the classification report shown in Table 10.

Table 10. Classification Report of Sequential Model Without Augmentation

| Class | Precission | Recall | F1-Score |
|---|---|---|---|
| K | 0.97 | 1.00 | 0.98 |
| Kah | 1.00 | 0.98 | 0.99 |
| Kang | 0.86 | 0.93 | 0.89 |
| Kar | 0.98 | 0.97 | 0.97 |
| Ke | 0.92 | 0.97 | 0.94 |
| Ké | 1.00 | 0.98 | 0.99 |
| Keu | 0.98 | 0.97 | 0.97 |
| Ki | 0.94 | 0.88 | 0.91 |
| Kla | 1.00 | 0.98 | 0.99 |
| Ko | 1.00 | 1.00 | 1.00 |
| Kra | 1.00 | 1.00 | 1.00 |
| Ku | 1.00 | 0.98 | 0.99 |
| Kya | 1.00 | 1.00 | 1.00 |

The CNN model's evaluation results demonstrate that the majority of classes performed exceptionally well, with most achieving a 1.00 f1-score, precision, and recall. Classes like "Ko" and "Kya" achieved perfect scores across all metrics. The "Keu" class recorded a precision of 0.98, recall of 0.97, and f1-score of 0.97, while the "Ké" class had a precision of 1.00, recall of 0.98, and f1-score of 0.99. The "Ke" class recorded a precision and f1-score of 0.92, with a recall of 0.97. These findings demonstrate the CNN model's excellent ability to classify each class effectively without the need for data augmentation as shown in Figure 22.

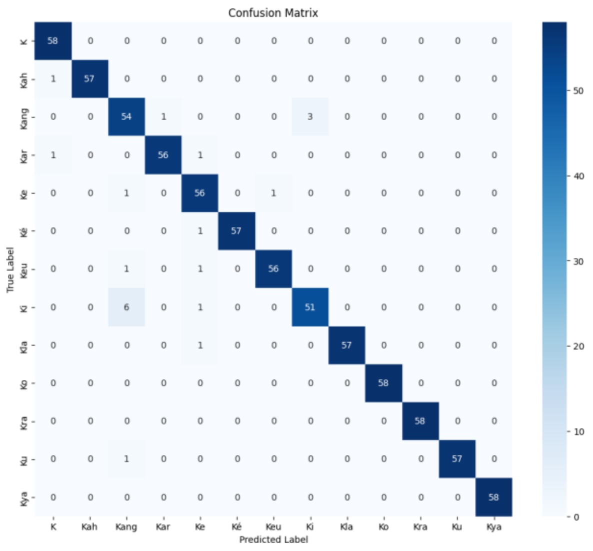

Figure 20. Confussion Matrix of Sequential Model Without Augmentation

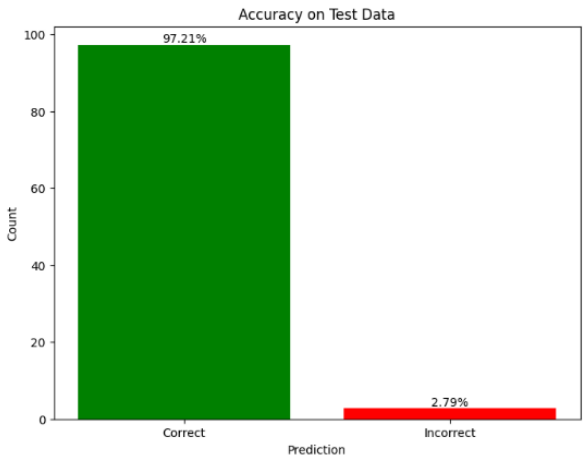

Figure 21. Comparison of Correct and Incorrect Test Data using Sequential Model without Augmentation

For the CNN model trained for 100 epochs using the Adam optimizer without data augmentation, the training accuracy reached 99.26%, and the loss value decreased significantly to 0.0196. The training duration was 2 hours and 9 minutes. When evaluated on the test set using the confusion matrix, the model achieved a test accuracy of 97.21%, with good recall, precision, and f1-score values for each class. These results demonstrate the model's strong ability to fit the training data while maintaining excellent performance on unseen data as shown in Figure 23.

Table 11 show the classification report for the MobileNetV2 model, showing strong precision, recall, and F1-scores across most classes. The "K" and "Kya" classes achieved perfect scores, while the "Ko" class had a precision of 0.98, recall of 1.00, and F1-score of 0.99. Other classes like "Kang" and "Kar" also performed well with precision above 0.90 and recall close to 1. Overall, the model demonstrates consistent and effective performance in classification, making it highly suitable for the task.

Table 11. Classification Report of MobileNetV2 Transfer Learning (Learning Rate 0.0001)

| Class | Precission | Recall | F1-Score |
|---|---|---|---|
| K | 1.00 | 1.00 | 1.00 |
| Kah | 0.95 | 1.00 | 0.97 |
| Kang | 0.98 | 0.93 | 0.96 |
| Kar | 0.92 | 0.93 | 0.92 |
| Ke | 0.90 | 0.97 | 0.93 |
| Ké | 0.98 | 0.95 | 0.96 |
| Keu | 0.98 | 0.90 | 0.94 |
| Ki | 0.95 | 0.98 | 0.97 |
| Kla | 0.97 | 0.98 | 0.97 |
| Ko | 0.98 | 1.00 | 0.99 |
| Kra | 1.00 | 0.98 | 0.99 |
| Ku | 1.00 | 0.98 | 0.99 |
| Kya | 1.00 | 1.00 | 1.00 |

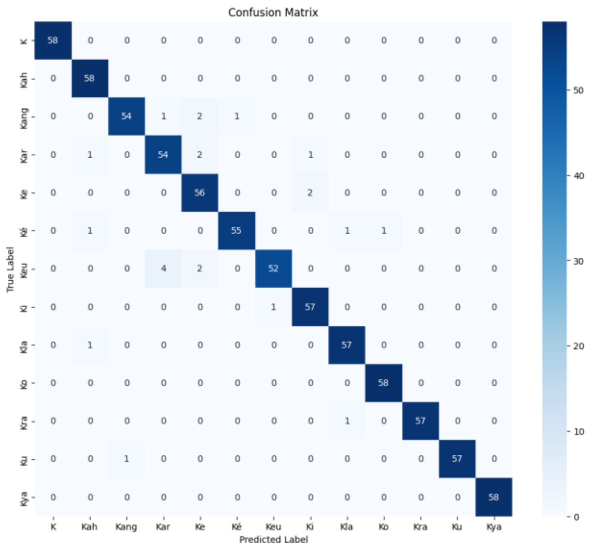

Figure 22. Confusion Matrix of MobileNetV2 Transfer Learning (Learning Rate 0.0001)

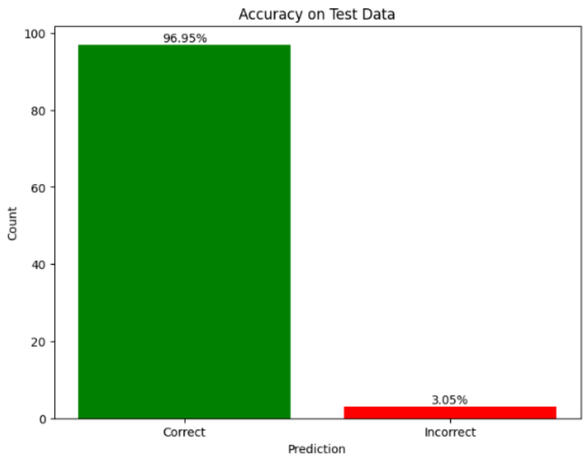

Figure 23. Performance Comparison of Test Data Using MobileNetV2 Transfer Learning (Learning Rate 0.0001)

The results demonstrate that the MobileNetV2-based Convolutional Neural Network (CNN) model, utilizing transfer learning and trained for 50 epochs using the

Adam optimizer with a learning rate of 0.0001, achieved a training accuracy of 98.75% with a loss value of 0.0129. The training process took 1 hour and 23 minutes. When evaluated on the test data using a confusion matrix, the model achieved a test accuracy of 96.95%, demonstrating robust performance with high precision, recall, and F1-score values across all classes shown in Figures 24 and 25.

Table 12 shows the classification report for the MobileNetV2 model, demonstrating strong precision, recall, and F1-score values across most classes. The "Kya" class showed perfect results with a precision, recall, and F1-score of 1.00, highlighting the model's excellent performance for this category. Classes such as "Kah," "Ko," and "Ku" also showed high scores, reflecting consistent accuracy. However, "Kang" and "Kar" showed slightly lower recall and F1-scores, indicating room for improvement in these classes. Overall, the model showed reliable and effective classification results across all categories.

Table 12. Classification Report of MobileNetV2 Transfer Learning (Learning Rate 0.001)

| Class | Precission | Recall | F1-Score |
|---|---|---|---|
| K | 1.00 | 0.97 | 0.98 |
| Kah | 0.97 | 1.00 | 0.98 |
| Kang | 0.90 | 0.97 | 0.93 |
| Kar | 0.91 | 0.90 | 0.90 |
| Ke | 0.93 | 0.93 | 0.93 |
| Ké | 1.00 | 0.95 | 0.97 |
| Keu | 0.97 | 0.98 | 0.97 |
| Ki | 0.98 | 0.95 | 0.96 |
| Kla | 0.95 | 0.98 | 0.97 |
| Ko | 0.97 | 1.00 | 0.98 |
| Kra | 1.00 | 0.98 | 0.99 |
| Ku | 1.00 | 0.97 | 0.98 |
| Kya | 1.00 | 1.00 | 1.00 |

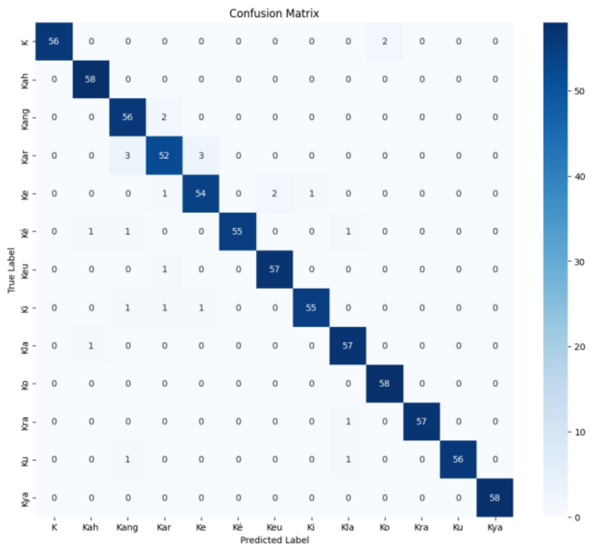

Figure 24. Confusion Matrix of MobileNetV2 Transfer Learning (Learning Rate 0.001)

The results of this study show that the MobileNetV2 model with a learning rate of 0.001 demonstrated strong performance in training. Using the Adam optimizer and trained for 50 epochs, the model achieved a training accuracy of 99.20% with a loss value of 0.0248. The training process was completed in 1 hour and 40 minutes. When evaluated on the test data, the model achieved a test accuracy of 96.68%, demonstrating strong performance with high precision, recall, and F1-score values across all classes. This result emphasizes the effectiveness of the MobileNetV2 architecture, particularly when utilizing transfer learning, to achieve high performance in the task of recognizing Sundanese script as shown in Figures 26 and 27.

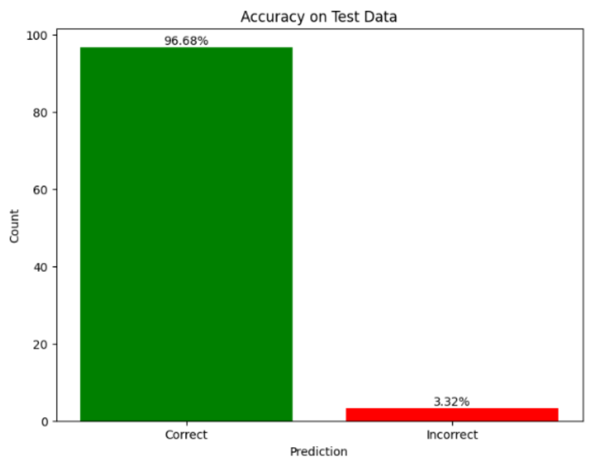

Figure 25. Performance Comparison of Test Data Using MobileNetV2 Transfer Learning (Learning Rate 0.001)

When analyzing the performance of different models in this study, the results show varying degrees of accuracy and training efficiency. The simpler CNN architecture without augmentation achieved the highest training accuracy of 99.26% with a training time of 2 hours and 9 minutes. This model, trained for 100 epochs using the Adam optimizer without data augmentation, reached a loss value of 0.0196. In comparison, the CNN model with augmentation, also trained for 100 epochs with the Adam optimizer, achieved a training accuracy of 94.42% and a loss value of 1.539, completing the training in 2 hours and 22 minutes.

The MobileNetV2 based CNN model, which uses transfer learning,also showed promising results. The MobileNetV2 model trained for 50timeswith Adam optimizer and learning rates of 0.001, achieved training accuracy of 99.20% and loss value of 0.0248,and completed training in 1 hour and 40 minutes.In the evaluation of test data, it achieved a test accuracy of 96.68%,showingits strong performance with balanced accuracy, recall, and F1 score values in all classes.In addition,the MobileNetV2 model achieved training accuracy of 98.75% and loss value of 0.0129 in 1 hour and 23 minutes using a learning rate of 0.0001, demonstrating robust performance in all classes and even more efficient training times. The success of these models has significant implications for cultural protection and education. By facilitating the digitization of historical Sundanese manuscripts and allowing the creation of accessible digital archives, this approach helps to preserve cultural heritage. In education, the model supports interactive learning and automated evaluation in the Sundanese script education by connecting traditional learning methods with modern digital tools.

Based on these findings, several areas for improvement and future research are identified. Technical enhancements could include the implementation of attention mechanisms and the development of lightweight versions for mobile deployment. Dataset expansion should focus on including historical manuscript samples and more regional writing variations. These improvements would further enhance the model's utility in both cultural preservation and educational applications, while addressing current limitations in recognition accuracy and processing efficiency.

## 4. Conclusions

The results of the study show that the simple CNN architecture without increase reached the highest accuracy of 99.26% with a training time of 2 hours and 9 minutes. In comparison, the enhanced CNN model achieved 94.42% accuracy in 2 hours and 22 minutes, and the MobileNetV2 model achieved a learning rate of 0.001 and a training accuracy of 99.20% and a test accuracy of 96.68% in 1 hour and 40 minutes. Further more, the MobileNetV2 model with a learning rate of 0.0001 achieved 98.75% training accuracy and 96.95% test accuracy in 1 hour and 23 minutes. These results enable practical applications in both education and cultural preservation. The model can be integrated into interactive learning applications for teaching Sundanese script and support the digitization of historical manuscripts, making cultural resources more accessible to researchers and the public. The suggestions for future research include collecting more diverse data with historical manuscripts and regional variations, implementing technical improvements such as attention mechanisms for mobile applications, and utilizing higher-capacity GPU hardware to accelerate the model training process. This study advances both the technological aspects of pattern recognition and the preservation of Indonesian cultural heritage, demonstrating how modern technology can effectively contribute to maintaining traditional cultural elements in the digital era.

## References

[1] F. Febriansyah, N. R, A. I. Purnamasari, O. Nurdiawan, and S. Anwar, “Pengenalan Teknologi Android Game Edukasi Belajar Aksara Sunda untuk Meningkatkan Pengetahuan,” *JURIKOM (Jurnal Riset Komputer)*, vol. 8, no. 6, p. 336, Dec. 2021, doi: 10.30865/jurikom.v8i6.3676.

[2] Rosalina, N. Afriliana, W. H. Utomo, and G. Sahuri, “Deep learning utilization in Sundanese script recognition for cultural preservation,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 36, no. 3, pp. 1759–1768, Dec. 2024, doi: 10.11591/ijeecs.v36.i3.pp1759-1768.

[3] A. Kirana and H. Hikmayanti, “Pengenalan Pola Aksara Sunda dengan Metode Convolutional Neural Network,” *Scientific Student Journal for Information, Technology and Science*, vol. 1, no. 2, pp. 95–100, 2020.

[4] A. Biswas and Md. S. Islam, “An Efficient CNN Model for Automated Digital Handwritten Digit Classification,” *Journal of Information Systems Engineering and Business Intelligence*, vol. 7, no. 1, p. 42, Apr. 2021, doi: 10.20473/jisebi.7.1.42-55.

[5] A. Willyanto, D. Alamsyah, and H. Irsyad, “Identifikasi Tulisan Tangan Aksara Jepang Hiragana Menggunakan Metode CNN Arsitektur VGG-16,” *Jurnal Algoritme*, vol. 2, no. 1, pp. 1–11, 2021.

[6] S. N. Rahmawati, E. W. Hidayat, and H. Mubarok, “Implementasi Deep Learning Pada Pengenalan Aksara Sunda Menggunakan Metode Convolutional Neural Network,” *INSERT: Information System and Emerging Technology Journal*, vol. 2, no. 1, pp. 46–58, 2021.

[7] L. S. Wulandari, E. Rosalina, A. Shomami, and P. N. Jakarta, “Adaptive Learning of Sundanese Script Based on Android Games in The Digital Era,” *Jurnal Ilmiah UPT P2M STKIP Siliwangi*, vol. 10, no. 1, pp. 25-37, 2023

[8] C. Nisa and F. Candra, “Klasifikasi Jenis Rempah-Rempah Menggunakan Algoritma Convolutional Neural Network,” *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 4, no. 1, pp. 78–84, Dec. 2023, doi: 10.57152/malcom.v4i1.1018.

[9] A. Y. Nadhiroh, “Sistem Klasifikasi Jenis Kain Berdasarkan Tekstur Menggunakan Metode Support Vector Machine Berbasis Web Flask Fabric Type Classification System Based On Texture Using Vector Machine Support Method Based On Web Flask,” *Jurnal Ilmiah Informatika dan Komputer*, vol. 1, no. 1, pp. 56–60, 2024.

[10] M. Arsal, B. Agus Wardijono, and D. Anggraini, “Face Recognition Untuk Akses Pegawai Bank Menggunakan Deep Learning Dengan Metode CNN,” *Jurnal Nasional Teknologi dan Sistem Informasi*, vol. 6, no. 1, pp. 55–63, Jun. 2020, doi: 10.25077/teknosi.v6i1.2020.55-63.

[11] M. F. Setyawan, J. Devgan Oktawijaya, and S. Agustin, “Implementation of SVM in Soil Type Classification Using RGB Features,” vol. 14, no. 2, 2024, doi: 10.30700/sisfotenika.v14i2.452.

[12] M. Toyib, T. Decky, and K. Pratama, “Ilmu pengetahuan Alam,” *Kebumian dan Angkasa*, vol. 2, no. 3, pp. 108–120, 2024, doi: 10.62383/algoritma.v2i3.69.

[13] M. Momeny, A. M. Latif, M. Agha Sarram, R. Sheikhpour, and Y. D. Zhang, “A noise robust convolutional neural network for image classification,” *Results in Engineering*, vol. 10, Jun. 2021, doi: 10.1016/j.rineng.2021.100225.

[14] P. Nyoman and Putu Kusuma Negara, “Deteksi Masker Pencegahan Covid19 Menggunakan Convolutional Neural Network Berbasis Android,” *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 5, no. 3, pp. 576–583, Jun. 2021, doi: 10.29207/resti.v5i3.3103.

[15] A. Satriawan, B. Imran, and S. Erniwati, “Identifikasi Kemiripan Foto Asli Dan Sketsa Menggunakan Model Generative Adversarial Networks (GANs),” *Jurnal Kecerdasan Buatan dan Teknologi Informasi*, vol. 2, no. 3, pp. 122–127, 2023.

[16] G. Henry, A. Panjaitan, and F. Simatupang, “Pemodelan Klasifikasi Penyakit Daun Tanaman Tomat dengan Convolutional Neural Network Algorithm,” *KLIK: Kajian Ilmiah Informatika dan Komputer*, vol. 4, no. 5, pp. 2667–2667, 2024, doi: 10.30865/klik.v4i5.1646.

[17] Y. Jumaryadi, A. Muhammad Ihsan, and B. Priambodo, “Klasifikasi Jenis Buah-Buahan Menggunakan Citra Digital Dengan Metode Convolutional Neural Networks,” *KLIK: Kajian Ilmiah Informatika dan Komputer*, vol. 4, no. 3, pp. 1737–1746, 2023, doi: 10.30865/klik.v4i3.1421.

[18] J. Homepage, M. Faizal Nazili, A. B. Firmansyah, and R. Purbaningtyas, “Klasifikasi Keparahan Demensia Alzheimer Menggunakan Metode Convolutional Neural Network pada Citra MRI Otak,” *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 3, no. 1, pp. 1–7, 2023.

[19] S. Liang, Tony Tan, and J. Jonathan, “MobileNetV3-based Handwritten Chinese Recognition Towards the Effectiveness of Learning Hanzi,” *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 7, no. 6, pp. 1394–1402, Dec. 2023, doi: 10.29207/resti.v7i6.5505.

[20] A. N. Hermana, M. Gustiana Husada, and O. Kurniawan, “Penerapan SMOTE Untuk Mengatasi Data Imbalance pada Identifikasi Originalitas Sepatu Converse Menggunakan CNN Arsitektur VGG-16,” 2024.

[21] D. Normawati and S. A. Prayogi, “Implementasi Naïve Bayes Classifier Dan Confusion Matrix Pada Analisis Sentimen Berbasis Teks Pada Twitter,” *Jurnal Sains Komputer & Informatika (J-SAKTI)*, vol. 5, no. 2, pp. 697-711, 2021.

[22] I. Wulandari, H. Yasin, and T. Widiharih, “Klasifikasi Citra Digital Bumbu Dan Rempah Dengan Algoritma Convolutional Neural Network (CNN),” *JURNAL GAUSSIAN*, vol. 9, no. 3, pp. 273–282, 2020.

[23] H. I. Islam, M. Khandava Mulyadien, U. Enri, U. Singaperbangsa, and K. Abstract, “Penerapan Algoritma C4.5 dalam Klasifikasi Status Gizi Balita,” *Jurnal Ilmiah Wahana Pendidikan*, vol. 8, no. 10, pp. 116–125, 2022, doi: 10.5281/zenodo.6791722.

[24] I. Wulansari and R. Arief, “Analisis Performa Metode Convolutional Neural Network (CNN) Dengan Word Embedding Glove Pada Klasifikasi Sentimen Dari Twitter,” *Jurnal Ilmiah Teknologi dan Rekayasa*, vol. 28, no. 3, pp. 252–264, 2023, doi: 10.35760/tr.2023.v28i3.6090.