Volume XI Issue 3 Year 2026 | Page 1042-1052 | ISSN: 2527-9866
Received: ***10-06-2026*** | *Revised:* ***12-07-2026*** | *Accepted:* ***14-08-2026***

Implementation of a Convolutional Neural Network Using VGG19 for Ogan Malay Script Recognition

Steven Liem[1], Hafiz Irsyad[2]
[1,2] Multi Data Palembang University, Palembang, Indonesia
e-mail: stevenliem115@mhs.mdp.ac.id[1], hafizirsyad@mdp.ac.id[2]
**Correspondence:* stevenliem115@mhs.mdp.ac.id

Abstract: *Regional languages and scripts, including the Melayu Ogan script, face the threat of extinction due to declining usage and limited digital documentation in the modern era. While current Indonesian script research primarily focuses on popular scripts, research addressing the Ogan Malay script remains severely limited. To address this gap, this study provides one of the earliest implementations of the Convolutional Neural Network (CNN) VGG-19 architecture specifically designed for Ogan Malay script classification. This research utilizes a primary dataset provided by the Language Center of South Sumatra Province, consisting of 185 distinct character classes, with each class initially containing one original image. The VGG-19 architecture is applied and supported by data augmentation techniques to enrich spatial variability, followed by evaluation using K-Fold Cross Validation. Evaluation results demonstrate excellent classification performance. The model achieved maximum convergence without any indications of overfitting at an optimal configuration of 30 epochs with a learning rate of 0.0001. This configuration successfully resulted in an Accuracy of 99.14%, Precision of 0.9870, Recall of 0.9914, and an F1-Score of 0.9885. The success of this classification model provides a strong foundation for future real-time regional script recognition applications to support cultural preservation.*

Keywords: *Melayu Ogan Script; VGG-19, Convolutional Neural Network.*

1. Introduction

Communication is the process of conveying information between two or more individuals [1]. One of the most essential elements in this process is language. Since ancient times, language has functioned not only as a tool for human interaction but also as a medium for passing down knowledge [2]. Furthermore, language plays a vital role as the identity of a nation or country because it possesses unique characteristics that represent its region of origin. In addition to spoken language, regional scripts and dialects also reflect the cultural uniqueness that strengthens a region's identity [3].

In the context of Indonesia, the diversity of languages and scripts is a cultural treasure that must be protected and preserved [4]. According to the Agency for Language Development and Cultivation [5], Indonesia has over 700 regional languages spread across various regions, making it one of the most linguistically diverse countries in the world. However, in today's era of globalization and digitalization, the use of regional languages has experienced a significant decline. The younger generation tends to consume digital content primarily in foreign languages or standard Indonesian, which slowly displaces the role of regional languages in everyday life. This situation has led to the increasingly rare use of regional languages, raising concerns that it may cause the extinction of traditional languages and scripts in the future [6].

Regarding regions in Indonesia that predominantly use a specific regional language, typically on a per-province basis, the focus of this discussion will be on a regional language in Sumatra[7]. Historically, the literacy rate in Sumatra was relatively high and non-exclusive, meaning it was not restricted solely to individuals from high social strata. This is evidenced by the discovery of manuscripts authored by ordinary people writing based on their own knowledge [8]. Examples include the prevalence of traditional courting customs in ancient

Sumatra, which resulted in lamentation manuscripts written by bachelors in Batak and Kerinci [9], as well as manuscripts in the Ulu script that can still be read and written by elders from various social backgrounds [10], In addition to the written tradition, ancestral regional languages are still actively used as daily communication tools in various villages and continue to be passed down to the younger generation, as can be observed in the Muara Enim Regency [11]. This research focuses on one such regional language that is becoming increasingly rare: the Ogan Malay language and its script. The Ogan Malay script holds significant historical and cultural value as the identity of the Ogan community in South Sumatra [12]. However, its lack of use in everyday life and minimal documentation have made it difficult for the younger generation to recognize and learn. Without concrete preservation efforts, there is a legitimate fear that the Ogan Malay script will become increasingly marginalized.

Given the high language diversity in Indonesia and the lack of preservation efforts in digital formats, this research is crucial in providing a modern solution to help reintroduce the Ogan Malay script to the wider public [13]. One approach is by utilizing artificial intelligence technology, specifically in the field of image processing. A widely used method in pattern recognition is the CNN due to its ability to automatically extract visual features from an object. Furthermore, utilizing the VGG architecture approach can improve CNN performance, enabling it to recognize script patterns with higher accuracy [14].

In the fields of pattern, character, and sign language recognition, the shift from classical feature extraction methods to deep learning architectures has shown significant performance improvements. In the early stages, the use of non-deep learning methods with classical features still dominated. [15] implemented the K-Nearest Neighbor (K-NN) algorithm combined with Histogram of Oriented Gradient (HOG) feature extraction on the American Sign Language (ASL) alphabet, which was able to achieve up to 99% accuracy at k=3 and k=5. A similar approach was reported by [16], who used Artificial Neural Networks (ANN) for Arabic script; they found that HOG features provided more optimal and robust accuracy in representing visual shape characteristics compared to LBP features. Although yielding good results, these methods have limitations as they do not use modern CNN architectures and often struggle to handle complex datasets.

Along with the transition to deep learning methods, the application of basic CNN architectures began to be tested, although they often produced moderate accuracy levels in the early stages. For instance, the application of the LeNet CNN architecture by [17] successfully achieved an accuracy of 98.8% in the prediction stage of the Indonesian Sign Language System (SIBI). However, for more complex character recognition using simple architectures, the results were quite limited. [18] implemented a 5-layer CNN for Hiragana characters with 82% accuracy, while [19] used the AlexNet architecture for Hebrew letters and only reached a maximum accuracy of 81.5%. Interestingly, various studies consistently highlight the crucial role of optimizers in CNN architectures. Both the studies by [19] and [20] proved that the Adam optimizer is far superior compared to SGD. For the Hiragana script, [20] noted a surge in accuracy up to 97.6% when using VGG-16 with the Adam optimizer, in stark contrast to SGD which slumped at 36.4%. The reliability of the Adam optimizer was also proven by [21] in the preservation of the Sundanese script, successfully recording an overall prediction accuracy of 96.7%.

Furthermore, the adoption of deeper CNN architectures such as VGG shows very strong performance dominance, both on large-scale datasets and varied research objects. On a dataset of 72,000 ASL images, [22] utilized VGG-19 and recorded an almost perfect average accuracy of 99.995%. The consistency of the VGG architecture family was also confirmed outside the text character recognition domain; [23] compared VGG-16 and VGG-19 in the classification of rice varieties, where VGG-16 slightly led with a 98% accuracy compared to VGG-19 (97%). Despite the very high accuracy of CNN-based character recognition, full Optical Character Recognition (OCR)-based translation systems still harbor segmentation challenges. This is

clearly illustrated in the research by [24] on a translation machine for the regional Minangkabau language; although the initial character recognition accuracy was excellent at 98.97%, its performance momentarily plummeted to 50.72% during the word classification process due to segmentation failures, before eventually being restored to 75.78% through the implementation of the Levenshtein Distance method. This consistent track record affirms that the use of modern architectures (such as VGG) with the Adam optimizer is a highly reliable standard, although hierarchical segmentation challenges still require special handling.

Moreover, current research on regional scripts in Indonesia is still dominated by relatively popular scripts such as Javanese, Sundanese, and Arabic-Malay. Conversely, research on the Ogan Malay script is still very limited, whether in terms of dataset availability, models, or the implementation of automatic recognition systems. Therefore, this research focuses on implementing the VGG-19 CNN architecture for the classification of the Ogan Malay script using a limited local dataset. This study also analyzes the model's performance under data constraints to support the digitalization and preservation of regional scripts [25]. This study contributes to the field of local script recognition in three main aspects. First, it provides one of the earliest implementations of the VGG-19 architecture for Ogan Malay script classification. Second, this research evaluates the effectiveness of data augmentation and hyperparameter optimization in improving recognition performance on limited local datasets. Third, the study provides an experimental foundation for future development of real-time regional script recognition systems to support digital cultural preservation.

2.Methods

A. Data Description

This study utilized a dataset provided by the Language Center of South Sumatra Province, which consists of 185 language image classes, with each class containing 1 image. These images then underwent an augmentation process to increase the number of available images. The augmentation process is divided into two stages. The first stage uses geometric transformations with rotations of 30°, 45°, 60°, 300°, 315°, and 330°, generating 6 new images. These are then divided using an 80:10:10 ratio into 5 training images, 1 testing image, and 1 validation image. The second stage employs geometric transformations (5° and 10°), scale modifications (zoom 0.8 and 1.2), as well as variations in brightness levels (0.7 and 1.3). This procedure effectively multiplies the dataset into 17 images, which were subsequently split using an 80:10:10 ratio into 15 training images, 1 testing image, and 1 validation image.

B. Research Methodology

This chapter explains the workflow of the research conducted. The workflow begins with a literature study, dataset collection, design, implementation, and research instruments. This research integrated the VGG-19 model for Ogan Malay text classification based on an image dataset provided by the Language Center of South Sumatra Province.

1. Literature Study

At this stage, the research problem is identified through a literature review encompassing national and international journals sourced from the internet, directly related to the topic and title of this study: "Implementation of a Convolutional Neural Network Using VGG19 for Ogan Malay Script Recognition." Based on the search results, 10 closely related and highly relevant journals were selected as reference sources, which can be seen in the related research subsection.

2. Dataset Collection

At this stage, the dataset is collected independently in the form of two-dimensional image data. The steps taken during the dataset collection are as follows:

- The image data of the Ogan Malay script was obtained from official sources at the Language Agency (*Balai Bahasa*), provided by Mrs. Yulia Masithoh, S.Pd., to serve as a reference for the standard form of the script.
- The obtained image data were separated and underwent a manual rewriting process for the 24 basic characters along with 185 punctuation combinations, which can be seen on the appendix page.
- Each written character combination was converted into digital image data by photographing or scanning it.
- To enrich and strengthen the model's training capacity, the dataset underwent an extensive data augmentation process, which includes image rotation at various angles, brightness adjustments, and zooming.
- Afterward, all collected script image data were uploaded to the Kaggle platform to facilitate management and training in the subsequent stages.

An example of the Ogan Malay script dataset used can be seen in Figure 2.

Figure 2. Example of the Ogan Malay script dataset

3. Design

At this stage, the system design process required for classifying the Ogan Malay script using the VGG-19 model is conducted. The design of the developed system can be seen in Figure 3.

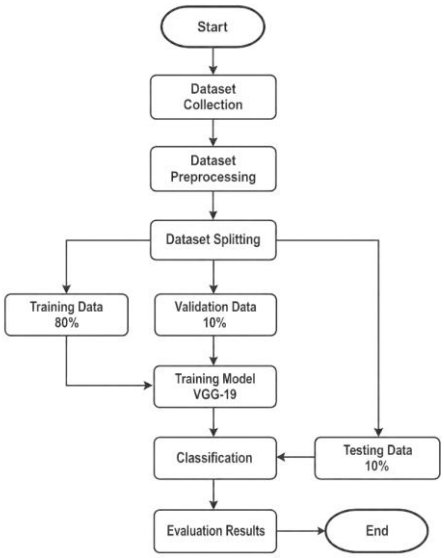

Figure 3. Design diagram

1. The first stage of this research begins with collecting a relevant dataset for Ogan Malay script classification, which was obtained from official sources at the Language Agency (*Balai Bahasa*).
2. The second stage involves preprocessing the collected dataset so the model can learn better. This process includes cropping the images, resizing them to 256×256 pixels, and applying image data augmentation, such as rotation (5 and 10 degrees), zooming (0.8 and 1.2), and adjusting brightness levels to 0.7 and 1.3.
3. The third stage is dataset splitting, where all the collected image data were divided into three parts: 80% training data, 10% validation data, and 10% test data.

Table 2. Arsitecture VGG-19

Layer (type)	Output Shape
Conv2d	256 x 256 x 64
Conv2d	256 x 256 x 64
MaxPooling2D	128 x 128 x 64
Conv2d	128 x 128 x 128
Conv2d	128 x 128 x 128
MaxPooling2D	64 x 64 x 128
Conv2d	64 x 64 x 256
Conv2d	64 x 64 x 256
Conv2d	64 x 64 x 256
Conv2d	64 x 64 x 256
MaxPooling2D	32 x 32 x 256
Conv2d	32 x 32 x 512
Conv2d	32 x 32 x 512
Conv2d	32 x 32 x 512
Conv2d	32 x 32 x 512
MaxPooling2D	16 x 16 x 512
Conv2d	16 x 16 x 512
Conv2d	16 x 16 x 512
Conv2d	16 x 16 x 512
Conv2d	16 x 16 x 512
MaxPooling2D	8 x 8 x 512
Flatten	32768
Dense	256
Dropout	256
Dense	185

4. The fourth stage is training the CNN model using the VGG-19 architecture to learn the visual features from the Ogan Malay script dataset. This architecture extracts features through convolutional and pooling layers, which are then flattened into a Dense layer for the final classification stage. To achieve optimal accuracy, the model is trained utilizing the Adam optimization algorithm with a batch size of 16. This training process evaluated and compared two primary hyperparameter scenarios: 30 epochs with a learning rate of 0.0001, and 60 epochs with a learning rate of 0.00001 using the Adam optimizer.
5. The fifth stage involves testing the trained VGG-19 model using the previously split 10% test data. If the model achieves a high accuracy rate and correctly classifies the Ogan Malay script images, it proceeds to the next stage. However, if the model exhibits poor accuracy and a high classification error rate, it reverts to the VGG-19 model retraining process.
6. The sixth stage is the final stage, wherein if the model demonstrates high accuracy and correct classification, it is saved as the best model in this study for future development in advanced stages.

To ensure that the implemented model performs well, this study utilizes a simple baseline model consisting of a basic CNN architecture. This comparison aims to evaluate the performance improvement achieved by utilizing the VGG-19 architecture.

4. Implementation

The implementation stage is the actual execution of the established design, where the VGG-19 model begins training using the Ogan Malay script dataset. This process was conducted in a Kaggle Notebook to utilize the GPU to accelerate training. The implementation begins by loading all script image data from Kaggle and splitting the data into training (train), validation (validation), and testing (test) sets. Then, the model was trained iteratively over several epochs, where each epoch adjusted the model's weights to minimize loss (error) and maximize recognition accuracy. During training, data augmentation was used to introduce the model to various script variations. This implementation produced a final model ready to automatically recognize the 24 basic characters and 185 combinations of the Ogan Malay script.

5. Research Instrument

At this stage, the previously trained model was tested on a test set comprising 130 image data points (10%). Following the testing phase, the results are processed to evaluate the success rate of the applied VGG-19 method using K-Fold Cross Validation. This approach ensures more objective and stable performance measurements, where in each fold, a portion of the data is used as training data while the remainder serves as validation or test data. Subsequently, the prediction results from each fold were analyzed using a Confusion Matrix to calculate the values for accuracy, precision, recall, and F1-Score.

3. Results and Discussion

3.1 Results

In this study, the primary dataset used comprises 185 classes, with one original image for each class. To enrich the spatial and visual variability of the model, data augmentation methods were implemented. In this study, the data augmentation process is divided into two approaches. The first utilizes only geometric transformations (30°, 45°, 60°, 300°, 315°, 330°). This procedure transforms 1 original image dataset into a total of 7 images, which are divided using an 80:10:10 percentage split. This results in an allocation of 5 training sets, 1 test set, and 1 validation set.

The second approach utilizes configured augmentation parameters encompassing geometric transformations in the form of rotation (5° and 10°) and scale modification (zoom 0.8 and 1.2), as well as variations in brightness levels (0.7 and 1.3). This procedure effectively multiplies the dataset into 17 image representations per class. For the purposes of model training and evaluation, an 80:10:10 data splitting scheme is applied. This allocation distributes each class into 15 training sets, 1 testing set, and 1 validation set.

To determine the appropriate augmentation pattern for this Ogan Malay script dataset, the research began by utilizing geometric transformation data augmentation techniques (30°, 45°, 60°, 300°, 315°, 330°). The training process was conducted using tensorflow.keras and Python version 1.12.7, with parameter settings of K_Folds = 5, Channels = 3, and Batch_Size = 16. This training process utilized different numbers of epochs, namely 30 and 60, as well as different learning rates for both the 30-epoch and 60-epoch tests, employing the Adam optimizer for each configuration. Furthermore, the k-fold cross-validation used glorot_uniform initialization, which eliminates the need for the program to require a standard deviation input value. The results of this stage can be seen in Table 3.

Table 3. Training 1 results

	Epochs 30 | Learning Rate 1 x 10^{-4}	Epochs 60 | Learning Rate 1 x 10^{-5}
	Fold 1	
Accuracy	0.4919	0.0162
	Fold 2	
Accuracy	0.7405	0.0108
	Fold 3	
Accuracy	0.2757	0.0270
	Fold 4	
Accuracy	0.5243	0.0162
	Fold 5	
Accuracy	0.5297	0.0142
	Average	
Accuracy	0.5124	0.0167
Precision	0.4014	0.0052
Recall	0.5124	0.0173
F1-Score	0.4308	0.0063

Through Table 3, it can be clearly observed that the accuracy rate from the training results using 30 epochs with a learning rate of 0.0001 is far superior compared to using 60 epochs with a learning rate of 0.00001. This superiority is quantitatively evidenced by the average Accuracy achieved in each scenario, namely 0.5124 for the first scenario and only 0.0167 for the second scenario. The drastic decline in performance in the second scenario is also consistently seen across all other average evaluation metrics, with a Precision value of 0.0052, a Recall of 0.0173, and an F1-Score of 0.0063, which are extremely low and nearly approach zero. These results strongly indicate that the model is completely unable to learn the provided data patterns when using a learning rate that is excessively small.

For the research utilizing data augmentation techniques involving geometric transformations such as rotation (5° and 10°) and scale modifications (zoom 0.8 and 1.2), alongside variations in brightness levels (0.7 and 1.3), the training process was conducted under the same conditions as the program that generated the results in Table 3. However, this experiment was broken down individually to determine the more detailed impact of the epochs and learning rates.

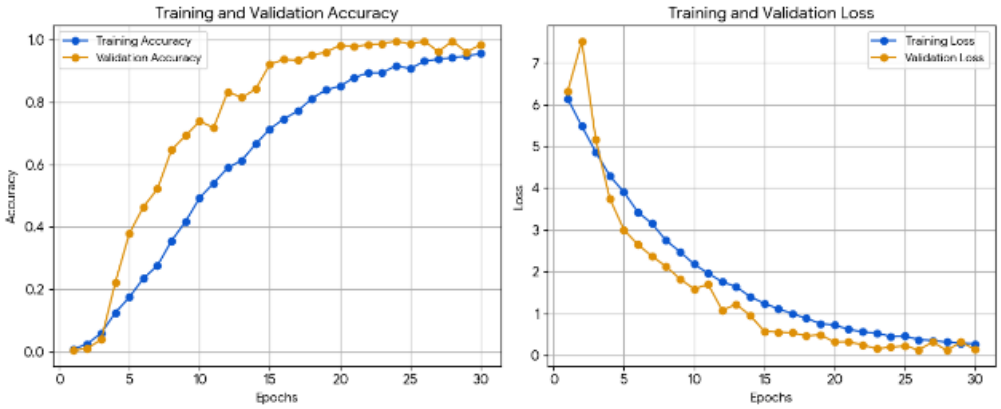

Figure 4. Fold 1 Epochs 30

Figure 4 presents a visualization of the fold 1 model's performance based on accuracy and loss metrics over 30 epochs during the training and validation phases. In the accuracy graph, a consistent upward trend is visible, where the validation accuracy increases significantly and remains stably above the training accuracy, reaching a saturation point near 100% after the 20th epoch. Concurrently, the loss graph shows an exponential decrease toward zero, with the validation loss persistently lower than the training loss after the fourth epoch. Overall, the

movement of these curves indicates that the model optimization process proceeded exceptionally well and achieved convergence without exhibiting any symptoms of overfitting. This suggests highly optimal model generalization capabilities, most likely due to the effectiveness of the regularization mechanisms during the training phase.

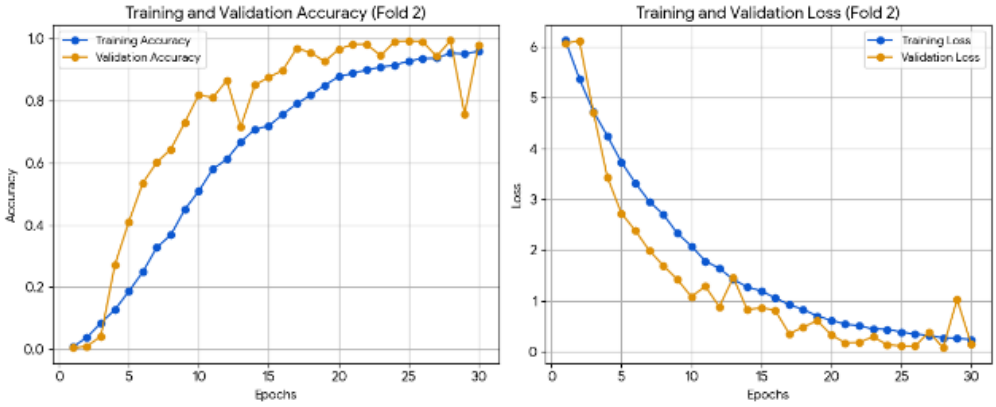

Figure 5. Fold 2 Epochs 30

Figure 5 presents a visualization of the accuracy and loss function metrics specifically for the Fold 2 testing iteration over 30 epochs. Similar to the previous general trend, the training metric curves demonstrate a stable learning process: accuracy increases consistently, and the loss value decreases exponentially toward convergence. However, a rather striking difference in this Fold 2 test lies in its validation curve, which exhibits a much higher level of volatility or fluctuation. Although the validation performance generally continues to outperform the training metrics, there are several anomalies in the form of sudden, sharp drops in accuracy and spikes in loss values, as clearly seen around the 13th and 29th epochs. This fluctuation indicates that the batch or validation data distribution in Fold 2 likely contains variances or samples that are much more challenging for the model to predict during that iteration. Nevertheless, on a macro level, the model proves capable of quickly recovering from these error spikes and re-achieving an excellent level of generalization by the end of the training process, without triggering overfitting.

From this, it can be understood that the impact of applying folds (in the context of k-fold cross-validation) is clearly visible in revealing the variability of the model's performance when encountering different data subsets. The first figure displays a very smooth learning curve, while the Fold 2 iteration shows quite high fluctuation and volatility during the validation phase. This difference proves that a model might perform very stably and optimally on one portion of the test data but encounter unusual instances (data with more complex distributions or features) on another portion of the test data. Therefore, folds play a crucial role in evaluating how robust and consistent the model's generalization capabilities are against various variations of previously unseen data, thus preventing researchers from drawing biased conclusions that might occur if relying solely on a single train-test split.

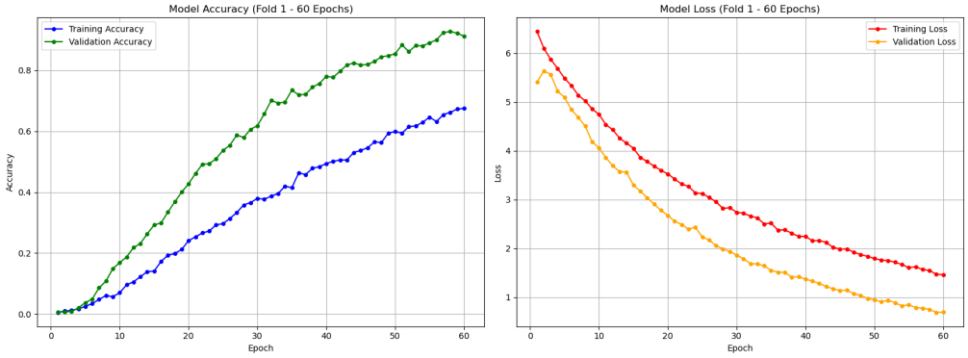

Figure 6. Fold 1 Epochs 60

Based on the comparison between Figure 4 and Figure 6, the impact of the learning rate and epoch combination on the model's convergence speed is highly significant, although specific classification metrics such as precision, recall, and F1-score require separate confusion matrix calculations as they are not represented in these loss and accuracy graphs. In Figure 4 (30 epochs, learning rate 0.0001), the model demonstrates an efficient and optimal learning process, characterized by an accuracy curve that climbs exponentially toward maximum convergence (approaching 100%) in just 30 iterations. Conversely, in Figure 6 (60 epochs, learning rate 0.00001), the use of a learning rate ten times smaller causes the model's weight update steps to become overly slow. Consequently, even though the number of iterations has been doubled to 60 epochs, the curve still moves linearly and sluggishly, indicating that the model has not yet reached its optimal point (underfitting). Overall, this phenomenon confirms that utilizing an excessively small learning rate demands compensation through a vastly larger number of epochs; thus, for this dataset, the first configuration proves to be far superior and computationally efficient in achieving the highest accuracy.

Table 4. Training 2 results

	Epochs 30 | Learning rate $1\ x\ 10^{-4}$	**Epochs 60 | Learning rate $1\ x\ 10^{-5}$**
	Fold 1	
Accuracy	0.9892	0.9351
	Fold 2	
Accuracy	1.0000	0.9189
	Fold 3	
Accuracy	0.9892	0.9351
	Fold 4	
Accuracy	0.9892	0.9243
	Fold 5	
Accuracy	0.9892	0.9027
	Average	
Accuracy	0.9914	0.9232
Precision	0.9870	0.8883
Recall	0.9914	0.9232
F1-Score	0.9885	0.8994

The quantitative data in Table 4 comprehensively validates the findings of the previous analysis, proving that the first scenario (30 epochs, learning rate 0.0001) consistently outperforms the second scenario (60 epochs, learning rate 0.00001) across all evaluation metrics. The efficient performance and optimal convergence in the first scenario are numerically evidenced by achieving an average Accuracy of 0.9914 (99.14%), which is accompanied by an excellent inter-class prediction balance with Precision (0.9870), Recall (0.9914), and F1-Score (0.9885) values that stably approach perfection. Conversely, the lagging performance in the second scenario, with the average Accuracy dropping to 0.9232 and an F1-Score of 0.8994, strongly confirms the hypothesis of underfitting due to the slow process of updating the model's learning weights. These test results definitively confirm that doubling the number of iterations to 60 epochs cannot compensate for the inefficiency of setting a learning rate that is too small, thereby solidifying the first configuration as the most optimal and computationally efficient parameter to produce robust model generalization.

3.2 Discussion

Based on the testing and evaluation results, the implementation of the VGG-19 architecture, supported by data augmentation techniques, has proven highly effective in classifying the Ogan Malay script. Performance analysis demonstrates that a hyperparameter configuration of 30 epochs with a learning rate of 0.0001 represents the most optimal and efficient scenario,

wherein the model successfully achieved maximum convergence stably without experiencing symptoms of overfitting. Under this best-case scenario, the model recorded an exceptionally high accuracy rate of 99.14%, accompanied by an excellent balance in class predictions, achieving a precision of 0.9870, a recall of 0.9914, and an F1-Score of 0.9885. Conversely, the application of a learning rate ten times smaller (0.00001) even when accompanied by doubling the training iterations to 60 epochs resulted in overly sluggish weight updates, thereby trapping the model in an underfitting state with an accuracy of 92.32%. These findings definitively assert that the effectiveness of the model's weight update steps through a precise learning rate parameter is far more crucial in establishing robust model generalization than merely increasing the number of training iterations.

4. Conclusions

Based on the testing and analysis results conducted on the implementation of a CNN using the VGG-19 architecture for Ogan Malay script classification, it can be concluded that hyperparameter combinations have a highly significant impact on the convergence speed and overall performance of the model. Testing proves that the first scenario, utilizing 30 epochs with a learning rate of 0.0001, is the most optimal and computationally efficient configuration. In this configuration, the model successfully achieved maximum generalization without exhibiting symptoms of overfitting, as evidenced by an Accuracy rate of 0.9914 (99.14%), Precision of 0.9870, Recall of 0.9914, and F1-Score of 0.9885. Conversely, using an excessively small learning rate, namely 0.00001 in the second scenario, caused the weight update steps to become too slow. Consequently, even though the training process was doubled to 60 epochs, the model still experienced underfitting with a lower Accuracy rate of 0.9232 (92.32%).For future research development, it is recommended that the capabilities of this optimally trained Ogan Malay script classification model be tested in real-time. Direct field testing for example, by integrating the model into a scanner system based on a smart mobile application or webcam is highly necessary to evaluate how well and robustly the model performs when faced with physical handwritten objects under dynamically varying real-world conditions of lighting, camera angles, and writing media textures (backgrounds). This step is expected to refine the implementation of the script recognition system, making it more applicable and ready for widespread use in efforts to preserve regional culture.

References

[1] L. G. Abed, M. G. Abed, and T. K. Shackelford, “Interpersonal communication style and personal and professional growth among Saudi Arabian employees,” *Int. J. Environ. Res. Public Health*, vol. 20, no. 2, 2023, doi: 10.3390/ijerph20020910.

[2] O. Mailani, I. Nuraeni, S. A. Syakila, and J. Lazuardi, “Bahasa sebagai alat komunikasi dalam kehidupan manusia,” vol. 1, no. 2, pp. 172–181, 2022, doi: 10.35335/kampret.v1i1.8.

[3] H. W. Susianti, Y. J. Moon, and I. S. Budi, “The relationship between culture and language: an anthropological linguistics study,” *LACULTOUR: Journal of Language and Cultural Tourism*, vol. 3, no. 2, pp. 74–83, 2024, doi: 10.52352/lacultour.v3i2.1611.

[4] F. Tazakka, S. A. Aurell, S. A. Alamsyah, Z. A. Fahrizan, and S. Hamidah, “Analisis persebaran dan faktor variasi bahasa di perbatasan Jawa-Sunda,” *Locus Journal of Academic Literature Review*, vol. 4, no. 6, pp. 421–429, 2025, doi: 10.56128/ljoalr.v4i6.567.

[5] Badan Pengembangan dan Pembinaan Bahasa, *Konservasi Bahasa Tahun 2021*. 2021.

[6] H. F. N. Istiqomah, G. Pratama, and Mushoffa, “Fenomena keberagaman bahasa daerah di Banyuwangi Jawa Timur, Indonesia,” *Media Bahasa, Sastra, dan Budaya Wahana*, vol. 30, no. 1, pp. 73–84, 2024, doi: 10.33751/wahana.v30i1.10386.

[7] N. F. Ramadhona, Wasino, and A. Aricindy, “Bahasa Komering sebagai cermin identitas budaya : sebuah kajian review tentang manusia dan kebudayaan di Sumatera Selatan,” *Seulas Pinang*, vol. 8, no. 1, pp. 31–34, 2026, doi: 10.30599/qzfpdr66.

[8] Rahmatullah, “Sumatera language interference to Indonesian language,” *Iconties*, vol. 1, pp. 219–228, 2023.

[9] U. Kozok, "Kitab undang-undang Tanjung Tanah: naskah Melayu yang tertua," 2006.
[10] S. Sarwono and N. Rahayu, "Pusat penulisan dan para penulis manuskrip ulu di Bengkulu," 2017.
[11] R. I. Saputra, S. Suryati, and M. Muzaiyanah, "Analisis penggunaan bahasa daerah dalam berkomunikasi pada masyarakat Desa Gumai kecamatan gelumbang kabupaten Muara Enim," *Jurnal Bahasa Daerah Indonesia*, vol. 1, no. 1, pp. 10, 2024, doi: 10.47134/jbdi.v1i2.2279.
[12] G. Asita, D. Ningrum, W. Darmawan, S. Dwiatmini, and P. A. Budaya, "Peranan perbase dalam kehidupan masyarakat suku bangsa-bangsa," *Jurnal Budaya Etnika*, vol. 6, no. 1, pp. 29–42, 2022, doi: 10.26742/jbe.v6i1.2077.
[13] F. Rahmawan, R. Habibi, and M. Y. H. Setyawan, "Rekognisi huruf tulisan tangan menggunakan convolutional neural network," *Jurnal Sistem Cerdas*, vol. 6, no. 3, pp. 262–276, 2023, doi: 10.37396/jsc.v6i3.240.
[14] F. Syah, P. W. Ciptadi, A. E. Frinayanti, D. F. Anggraini, E. Amalia, and J. Informatika, "Implementasi CNN untuk penerjemahan bahasa melalui pengenalan citra tulisan tangan," *Jurnal Dinamika Informatika*, vol. 13, no. 2, pp. 59–70, 2024, doi: 10.31316/jdi.v13i2.510.
[15] M. E. AL Rivan, H. Irsyad, Kevin, and A. T. Narta, "Pengenalan alfabet American sign language menggunakan k-nearest neighbors dengan ekstraksi fitur histogram of oriented gradients," *Jutisi*, vol. 5, no. 3, pp. 328–339, 2019, doi: 10.28932/jutisi.v5i3.1936.
[16] R. K. Wati and H. Irsyad, "Pengenalan aksara Arab menggunakan metode JST dengan fitur HOG dan LBP," *Algoritme*, vol. 2, no. 1, pp. 39-55, 2021, doi: 10.35957/algoritme.v2i1.1453.
[17] M. B. S. Bakti and Y. M. Pranoto, "Pengenalan angka sistem isyarat bahasa Indonesia dengan menggunakan metode convolutional neural network," *Prosiding SEMNAS INOVTEK (Seminar Nasional Inovasi Teknologi)*, vol. 3, no. 1, pp. 11–16, 2019, doi: 10.29407/inotek.v3i1.504.
[18] C. Umam and L. B. Handoko, "convolutional neural network (CNN) untuk identifkasi karakter Hiragana," *Prosiding Seminar Nasional Lppm Ump*, vol. 2, pp. 527–533, 2020.
[19] J. R. T. Baldi, Yohannes, and S. Devella, "Penggunaan convolutional neural network sebagai pengenalan huruf bahasa ibrani," *Jutisi*, vol. 12, no. 1, pp. 177-191, 2023, doi: 10.35889/jutisi.v12i1.1090.
[20] A. Willyanto, D. Alamsyah, and H. Irsyad, "Identifikasi tulisan tangan aksara Jepang Hiragana menggunakan metode CNN arsitektur VGG-16," *Algoritme*, vol. 2, no. 1, pp. 1–11, 2021, doi: 10.35957/algoritme.v2i1.1450.
[21] S. N. Rahmawati, E. W. Hidayat, and H. Mubarok, "Implementasi deep learning pada pengenalan aksara sunda menggunakan metode convolutional neural network," *INSERT*, vol. 2, no. 1, pp. 46–58, 2021, doi: 10.23887/insert.v2i1.37405.
[22] A. Fendiawati and M. E. Al Rivan, "Klasifikasi american sign language dengan metode VGG-19," *MDP Student Converence*, vol. 2, no. 1, pp. 192–197, 2023, doi: 10.35957/mdp-sc.v2i1.4466.
[23] A. Z. Noorizki and W. I. Kusumawati, "Perbandingan performa algoritma VGG16 dan VGG19 melalui metode CNN untuk klasifikasi varietas beras," *Complete*, vol. 4, no. 2, pp. 1-16, 2023, doi: 10.52435/complete.v4i2.387.
[24] M. M. Santoni, N. Chamidah, H. N. Prasvita, D. S. Irmanda, R. Astriratma, and R. A. Prayoga, "Penerapan convolutional neural networks untuk mesin penerjemah bahasa daerah minangkabau berbasis gambar," *Jurnal Resti*, vol. 5, no. 6, pp. 1153–1160, 2026, doi: 10.29207/resti.v5i6.3614.
[25] K. E. Simanungkalit, and J. D. Damanik, "Deep learning untuk pembelajaran bahasa dan sastra Indonesia: strategi, efektivitas, dan peluang," *Boras Pati Journal / Prosiding*, vol. 2, no. 1, pp. 203–215, 2025, doi: 10.64674/boraspatijournal.v2i3.25.