Jurnal Teknik Informatika C.I.T Medicom, 17 (5) (2025)247-256

Published by: Institute of Computer Science (IOCS)

Jurnal Teknik Informatika C.I.T Medicom

Journal homepage: www.medikom.iocspublisher.org

Comparative Evaluation of Cnn Architectures for Robust Font Type Classification Under Data Augmentation Scenarios

M. Rafif Akhdan[1], Bulkis Kanata[2], Misbahuddin[3]
[1,2,3] Teknik Elektro, Universitas Mataram, NTB, Indonesia

Article Info

Article history

Received : Juli 23, 2025
Revised : Augs 26, 2025
Accepted : Sept 30, 2025

Key Words:

CNN Architectures;
Font Types;
Identification;
Augmentation.

Abstract

Traditional font identification remains shortcomings because of takes time, and a huge cost in classifying the large amount of complex fonts. Thus, it requires an automated method to accurately distinguish fonts. This study compares three CNN architectures, including DenseNet121, ResNet50, and VGG16, to determine their effectiveness in font type classification. In this experiment, we conduct several tuning stages to optimize the model performance, including data pre-processing, the augmentation process, and parameter tuning. The model was trained and validated using a dataset containing 15 font types. According to the experimental result, evaluation metrics showed that DenseNet121 achieved the highest accuracy of 96.8%, followed by VGG16 at 96.4% and ResNet50 at 92.9%. Our proposed method with DenseNet121 architecture can be a promising font type classification solution in a real application.

Corresponding Author:

M. Rafif Akhdan,
Teknik Elektro,
Universitas Mataram, NTB, Indonesia
Jl. Majapahit No.62, Gomong, Kec. Selaparang, Kota Mataram, Nusa Tenggara Bar. 83115
Email: rafifakhdan43@gmail.com

1. Introduction

The problem of font type classification has traditionally been approached through handcrafted features, structural descriptors, and rule-based systems designed to capture visual elements such as stroke width, serif shape, curvature, and spacing. Early research in typography analysis emphasized manual categorization schemes to distinguish font attributes, as demonstrated in studies proposing structured classification frameworks for improving retrieval and stylistic consistency within writing systems such as Korean script [2], [3]. These traditional approaches relied heavily on domain expertise and predefined rules, leading to limitations in scalability and adaptability when confronted with large and diverse font datasets. In previous research related to the topic [9], the research focuses on the use of CNN for Sundanese script recognition. The CNN model used was optimized using the ADAM algorithm and tested at various epochs and learning rates. The best results were achieved with the Epoch 500 and a learning rate of 0.1. The accuracy for the training data is 98.03%, and the accuracy for the test data is 92.02%. In addition, several combinations of optimization and layer pooling were used to test the CNN model with the LeNet architecture to find the Katakana script [10]. Adam Optimiser and Average Pooling provide the best results, with 90% accuracy on the test data.

As font classification increased in complexity due to the emergence of multilingual scripts, historical inscriptions, and stylistic variations, traditional handcrafted models struggled to maintain accuracy and robustness. In studies involving ancient scripts like oracle bone inscriptions, alternative learning-based methods were proposed to manage irregular shapes and degraded character structures, highlighting the limitations of rule-driven strategies in handling noisy data [1]. Similarly, research on calligraphy and stylized handwritten text required feature extraction techniques capable of representing variability in brush strokes, which exceeded the capacity of conventional classification pipelines [4]. These limitations motivated the transition from manually engineered features toward data-driven approaches.

Deep learning, particularly CNNs, introduced a major shift in font recognition tasks by enabling automatic extraction of hierarchical and discriminative visual features. CNN models demonstrated superior performance across character recognition tasks involving scripts such as Sundanese, Katakana, Hiragana, and Chinese ideographs, where traditional approaches previously faced significant challenges in handling variations in writing patterns and imaging conditions [9]–[10], [13], [16]. By leveraging convolutional operations, these models learned multi-scale patterns without requiring handcrafted rules, enabling more accurate and generalizable font classification.

Advanced neural architectures further expanded the capabilities of CNN-based classification. Siamese CNNs used multi-loss learning for calligraphy style classification, effectively embedding fine-grained differences between font classes [4], while modified networks incorporating attention mechanisms and multi-scale feature pyramids improved performance in complex visual recognition tasks beyond typography, such as animal species identification [11]. These developments demonstrated that deeper and more sophisticated CNN architectures, such as ResNet, DenseNet, and VGG. It can offer scalable alternatives to traditional feature-engineering methods.

Across broader image recognition domains, CNNs have proven their robustness in tasks such as facial expression recognition, sign language classification, medical imaging, and face detection [12], [14], [15], [17], [18]. These applications share core challenges with font classification, including inter-class similarity, intra-class variability, and high dependence on subtle visual features. The repeated success of CNN models across diverse domains validates their suitability for font classification problems, where similar patterns of fine-grained discrimination are required. This further reinforces the need to compare different CNN architectures in font-related tasks.

In parallel, the incorporation of data augmentation techniques has emerged as a crucial strategy for addressing the scarcity and imbalance of training samples in visual classification tasks. Augmentation methods mitigate the risk of overfitting and enhance generalization by synthetically expanding the dataset through operations such as rotation, scaling, noise injection, and geometric transformations. Comprehensive surveys highlight the importance of augmentation in deep learning pipelines, especially when dealing with highly variable image data [19], [20]. For font classification, augmentation plays a critical role due to the large number of font families and styles that must be represented under inconsistent imaging conditions.

Evaluating CNN architectures under different augmentation scenarios is, therefore, an essential step toward understanding their robustness. Studies on hyperparameter tuning and model sensitivity have demonstrated that CNN performance can vary significantly depending on training conditions, choice of regularization, and data distribution [26], [27]. Moreover, research on imbalanced datasets highlights the importance of balanced or carefully controlled train–test splits to ensure fair evaluation, particularly when class distributions differ substantially across font categories [28]. These considerations emphasize the need for systematic comparative studies.

Therefore, the study presents a comparative evaluation of CNN architectures such as VGG16, ResNet50, and DenseNet121 becomes critical for advancing robust font type classification. While prior work has explored deep learning for stylistic analysis, calligraphy classification, and multilingual script recognition, limited studies have examined how different CNN architectures behave under diverse augmentation settings. Therefore, the comparison under controlled augmentation scenarios can provide deeper insights into the stability, reliability, and generalization capability of modern CNN

models. It is to address the fundamental limitations of traditional feature-based methods and advance the state of research in font classification.

2. Research Methodolgy

In this study, we gathered data as samples that consist of 15 different font types with a total sample of 11,252. Samples [17]. This dataset is largely employed to conduct various studies [18]. Data was obtained from two sources, namely an *open-source dataset* on Kaggle titled Font Recognition Data (https://www.kaggle.com/datasets/dhruvmak/font-recognition-data). In this study, we gathered up to 11,252 samples to train a model classifier. To obtain an engineered and credible dataset for the training process, we undergo data preprocessing in four steps. It involves checking the integrity of the data structure, calculating the number of categories/classes, validating the completeness of the data, and structural pre-processing to prepare the data format before analysis.

In the preprocessing stage, we divide the dataset into three main subsets: 9,001 data (80%) as training data, 1,125 data (10%) as validation data, and 1,126 data (10%) as test data [28]. We also employ a pre-processing stage to standardize the input data [19], [20]. All imagery was converted to a uniform dimension of 224×224 pixels, the batch size was set to 32 for memory efficiency, and the initial grayscale image was converted to a 3-channel RGB format to be compatible with ImageNet's pre-trained model [21], [22]. At the final stage of the pre-processing process, we undergo an augmentation phase to expand the variety of data without adding new datasets [23]. Augmentation techniques include *rotation*, translation, brightness change, and horizontal and vertical *flipping*. This augmentation is intended to make the model more resistant to real font variations, so that the accuracy evaluation can reflect the model's generalization capabilities.

To get the model performance for font type classification, this study compared DenseNet121, ResNet50, and VGG16 to construct a robust font type classification. DenseNet121 is a CNN that connects all the outputs from one layer and reuses those outputs as inputs to the next layer [11]. DenseNet121 is known for its inter-layer connectivity [12]. ResNet50 is one of CNN's architectures that introduces the concept of *shortcut connections*. The concept *of shortcut connections* emerged in the ResNet50 architecture, related to the gradient problem that disappeared when attempts to deepen the network structure were made [13]. ResNet has a skip system that aims to bypass the multiplication computing system [8]. The Visual Geometry Group (VGG) of the University of Oxford created the convolutional neural network architecture VGG16. Its architecture consists of 16 trainable layers, including 13 convolutional layers and three fully connected layers, with softmax-provided outputs [14]. On the other hand, CNN is a reliable algorithm for recognizing visual patterns and features. CNN was chosen for its ability to perform high-accuracy classification of image-formatted data, making it the right solution for font type identification [15].

To establish a robust font-type classification, these three architectures, including VGG16, ResNet50, and DenseNet121, offer complementary strengths grounded in their mathematical structures, with mathematical formulation as follows:

VGG16 is a purely sequential network defined as:

$$h^{(l)} = \sigma\left(W^{(l)} * h^{(l-1)} + b^{(l)}\right), l = 1, \dots, 16 \quad (1)$$

where $*$ is convolution, σ is ReLU, and max-pooling is inserted after specific layers. The final classifier is:

$$\hat{y} = \text{Softmax}(W_f h^{(16)} + b_f) \quad (2)$$

VGG16 emphasizes deep hierarchical feature extraction using sequential convolutional layers, enabling it to capture fine-grained character texture, stroke style, and edge transitions that serve as strong low-level cues for distinguishing fonts. Its straightforward chain of operations provides stable, high-resolution features suitable for typography analysis.

In the ResNet50 (Residual Learning) architecture, each residual block is formulated as:

$$h^{(l)} = F\left(h^{(l-1)}, W^{(l)}\right) + h^{(l-1)} \tag{3}$$

where $F(\cdot)$ is a stack of convolution–batchnorm–ReLU layers. The full network is:

$$\hat{y} = \text{Softmax}(W_f h^{(L)} + b_f) \tag{4}$$

with $L = 50$ layers.

ResNet50 introduces residual learning, allowing the model to learn deeper and more abstract typographic patterns without degradation. By using identity skip connections, ResNet effectively learns both subtle variations (e.g., serif curvature) and global shape structures (e.g., proportion, spacing) while overcoming vanishing gradients. This makes it robust in classifying fonts under noise, distortion, and varied text sizes that conditions often encountered in real-world documents and OCR workflows.

In DenseNet121, the dense block is expressed as:

$$h^{(l)} = H^{(l)}([h^{(0)}, h^{(1)}, \dots, h^{(l-1)}]) \tag{5}$$

where $[\cdot]$ is feature-map concatenation and$H^{(l)}(\cdot)$ is a composite function (BN–ReLU–Conv). The classifier is:

$$\hat{y} = \text{Softmax}(W_f h^{(121)} + b_f) \tag{6}$$

DenseNet121 further enhances representational power by concatenating all previous feature maps, promoting maximum feature reuse. This dense connectivity amplifies the model's ability to capture multi-scale font characteristics, from pixel-level textures to complex stylistic traits. The formulation ensures efficient gradient flow and compact parameterization, producing highly discriminative features for robust font classification. When combined or ensembled, these architectures form a powerful, resilient framework for high-accuracy font-type recognition across diverse styles and imaging conditions. [24], [25]. Table 1 describes the three architectures to construct the classifier model.

Table 1. CNN Architecture

DenseNet121			ResNet50			VGG16		
Parameter	Number of *Layers*	Value	Parameter	Number of *Layers*	Value	Parameter	Number of *Layers*	Value
Densenet121	1024	7.037.504	ResNet50	2048	23.587.712	VGG16	512	14.714.688
Custom dense 1	512	524.800	*Custom dense* 1	1024	2.098.176	*Custom dense* 1	1024	525.312
Batch normalisation	512	2.048	*Batch normalisation*	1024	4.096	*Batch normalisation*	1024	4.096
Droput	512	0	Droput	1024	0	Droput	1024	0
Custom dense 2	256	131.328	*Custom dense* 2	512	524.800	*Custom dense* 2	512	524.800
Batch normalisation 1	256	1.024	*Batch normalisation* 1	512	2.048	*Batch normalisation* 1	512	2.048
Droput 1	256	0	Droput 1	512	0	Droput 1	512	0

Custom dense 3 (dense)	128	32.896	*Custom dense 3* (dense)	256	131.328	*Custom dense 3* (dense)	256	131.328
Batch normalisation 2	128	512	*Batch normalisation 2*	256	1.024	*Batch normalisation 2*	256	1.024
Droput 2	128	0	Droput 2	256	0	Droput 2	256	0
Font prediction	15	1.935	*Font prediction*	15	3.855	*Font prediction*	15	3.855
Total Parameters: 7,732,047 (29.50 MB)			Total Parameters: 26,353,039 (100.53 MB)			Total Parameters: 15,907,151 (60.68 MB)		
Trainable Parameters: 1,768,399(6.75 MB)			*Trainable* Parameters: 10,643,983 (40.60 MB)			*Trainable* Parameters: 5,908,495 (22.54 MB)		
Non-Trainable parameters: 5,963,648 (22.75 MB)			*Non-Trainable* parameters: 15,709,056 (59.93 MB)			*Non-Trainable* parameters: 9,998,656 (38.14 MB)		

To train our model, the pre-processed dataset is fed into the three selected CNN architectures, where a series of hyperparameter optimization steps are performed to ensure stable and efficient learning. These adjustments include experimenting with different batch sizes to balance memory usage and gradient stability, tuning the learning rate to control convergence behavior, and applying dropout regularization to mitigate both overfitting and underfitting by reducing co-adaptation between neurons [26], [27]. During training, each architecture is fine-tuned to maximize its ability to extract representative features that distinguish between font types. For performance evaluation, a confusion matrix is constructed to analyze the distribution of true positives, false positives, true negatives, and false negatives across all font classes. This metric allows us to identify specific patterns of misclassification, revealing classes that are frequently confused and providing insight into underlying feature similarities or dataset imbalance issues [4].

By examining these patterns, we can diagnose weaknesses in the model and adjust training strategies accordingly. The overall evaluation aims to quantify how well each CNN architecture generalizes to unseen data and to highlight differences in predictive capability across models. Through this comparative assessment, we can determine which architecture produced the most robust performance for font-type classification in several training phases.

3. Results and Discussion

In this stage, we compare the architectures to identify 15 font types out of a total of 11,252 images with a training and testing process. Training is a step to assess the model's performance progress by measuring accuracy and loss during training. We also present several visualization results to recognize the model's ability in graph visualizations.

a. Model Accuracy and Loss

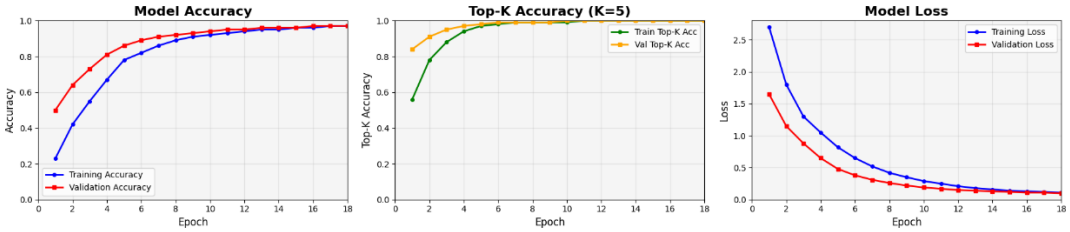
Based on Table 2, the results of the DenseNet121 training showed an accuracy of 96.31% in the 20th epoch, with a loss value of 0.1182. A visualization of the training results is shown in Figure 1(a), where the blue line represents the accuracy of the training data, while the red line represents the accuracy of the validation data. The graph shows the gradual improvement in model performance. The ResNet50 model obtained a training accuracy of 91.86% at the end of the training process with a loss value of 0.2289, as shown in Figure 2(a). Meanwhile, the VGG16 model recorded the highest accuracy during training, at 98.4%, with the lowest loss value of 0.0467, as shown in Figure 3(a). Overall, all three models were able to learn visual patterns from the font data, with the VGG16 performing best in the training phase.

Table 2. Model Training

Epoch	Architecture	*Accuracy*	*Loss*	Top k *accuracy*	Val *accuracy*	Vall *loss*
	DenseNet121	0.9631	0.1182	0.9999	0.9547	0.1278
20	ResNet50	0.9186	0.2289	0.9984	0.8978	0.3146

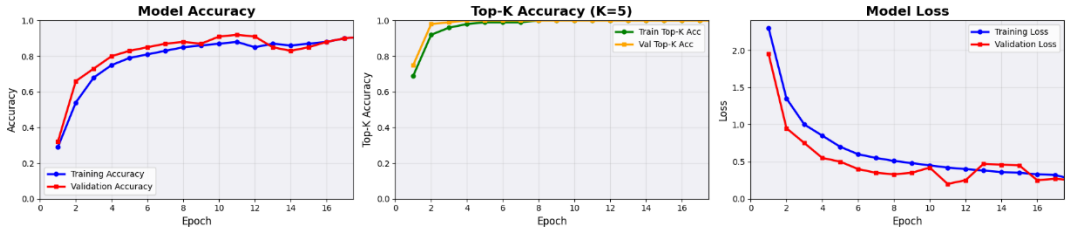
VGG16	0.9849	0.0467	0.9998	0.9271	02270

Further visualization using top-k metrics provides additional information regarding the model's ability to consider multiple possible predictions at once. In Figures 1(b), 2(b), and 3(b), the green and yellow lines respectively show the top-k accuracy on the training and validation data with k = 5. These results show that all three models are not only able to generate accurate top predictions, but also consider alternative predictions well. Furthermore, the loss graph on each model (Figure 1(c), 2(c), 3(c)) shows that the loss value in the training and validation data is relatively low and stable, indicating adequate generalization capabilities of the model.



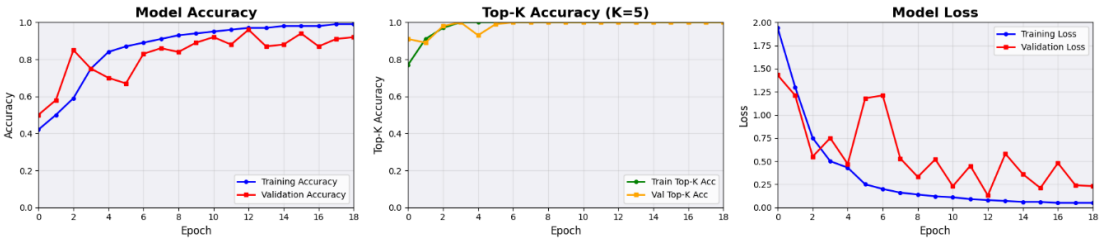
(a) (b) (c)

Figure 1. Accuracy and Loss of the DenseNet121 Model



(a) (b) (c)

Figure 2. ResNet50 Model Accuracy and Loss



(a) (b) (c)

Figure 3. Accuracy and Loss of the VGG16 Model

b. Model Evaluation

Table 3. Font Classification Report for Three Models

Jenis *Font*	DenseNet121			ResNet50			VGg16		
	Precision	*Recall*	F1-*Score*	Precision	*Recall*	F1-*Score*	Precision	*Recall*	F1-*Score*
Algerian	1.00	1.00	1.00	0.98	0.98	0.98	1.00	1.00	1.00

Arial	0.95	0.90	0.93	0.80	0.97	0.87	1.00	0.94	0.97
Courier	0.98	1.00	0.99	1.00	0.98	0.99	0.94	1.00	0.97
Garamond	0.96	0.96	0.96	0.86	0.93	0.89	0.98	0.88	0.92
Monotype Corsiva	1.00	1.00	1.00	0.98	1.00	0.99	1.00	1.00	1.00
Elephant	1.00	1.00	1.00	1.00	0.97	0.98	1.00	1.00	1.00
Consolas	0.97	0.97	0.97	0.95	0.81	0.87	0.94	0.96	0.96
Corbel	0.98	1.00	0.99	0.83	0.88	0.85	0.96	0.96	0.96
Helvetica	1.00	0.84	0.91	0.97	0.96	0.96	1.00	0.81	0.89
Calibri	0.91	0.89	0.90	0.83	0.82	0.83	0.80	1.00	0.87
News Gothic	0.82	1.00	0.90	0.89	0.82	0.86	0.93	0.98	0.96
Perpetua	0.96	0.96	0.96	0.86	0.88	0.87	0.93	0.98	0.96
Rockwell	1.00	1.00	1.00	1.00	0.98	0.99	1.00	1.00	1.00
Special Elite	1.00	1.00	1.00	1.00	0.98	0.99	1.00	0.98	0.99
Times New Roman	0.98	0.98	0.98	0.97	0.92	0.94	0.98	1.00	0.99

The model evaluation was carried out using test data of 1,126 images; each class consisted of 75 images, except for the Rockwell font, which had 76 images. The results of the evaluation per category in Table 3 show that most fonts can be identified well by all three models, with high precision, recall, and F1-score.

The results of the evaluation of DenseNet121 achieved an accuracy of 96.8%, ResNet50 92.9%, and VGG16 96.4%. Some fonts, such as Calibri and News Gothic, show relatively low scores (precision and recall 0.82–0.83), while most other fonts show consistently high scores. The advantage of DenseNet121 in identifying fonts over ResNet50 and VGG16 is due to its dense connectivity mechanism, which allows for more efficient information flow between layers and optimal feature extraction capabilities. The results of the evaluation of each model were also visualized using a confusion matrix, as shown in Figure 4, to determine the pattern of misclassification in each class.

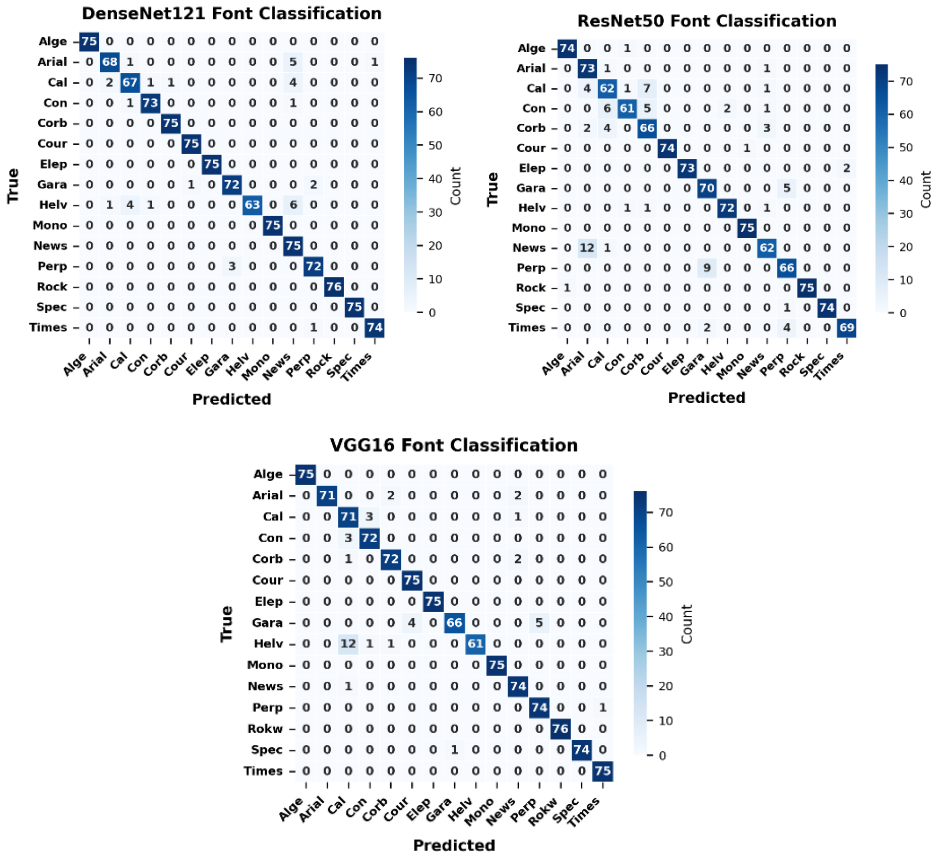


Figure 4. Confusion Matrix

c. Learning Rate Influences

Table 4. Accuracy and Loss Based on learning Rate Values

Learning Rate	DenseNet121		ResNet50		VGG16	
	Accuracy	*Loss*	Accuracy	*Loss*	Accuracy	*Loss*
0.1	94.2	0.2660	74.6	0.9454	46.1	0.5732
0.01	98.3	0.0580	86.5	0.4698	85.3	0.2013
0.0001	96.8	0.1182	92.9	0.2289	96.4	0.0467

Table 4 shows the analysis of the effect of learning rate variation through three values, 0.1, 0,01, and 0.0001. In general, the results of the experiment show that the selection of the learning rate has a significant impact on the stability of training and the achievement of the final accuracy of the model. In DenseNet121, the use of a medium learning rate (0.01) produces the most optimal performance compared to other values, both in terms of accuracy and loss stability. In contrast, ResNet50 shows a gradual improvement in performance as the learning rate decreases, with the best achievement at 0.0001. Meanwhile, VGG16 only achieved low accuracy at high learning rates (0.1), but showed significant improvements at low learning rates (0.0001).

According to the experimental results, this research can be a promising solution for font classification models with larger samples. In addition, this research is expected to make a practical contribution to the development of automated systems for examining documents, detecting forgery, and improving the performance of text-based image processing applications. The results of this research are expected to also benefit the digital security, printing, and creative industries.

4. Conclusion

This study compared three CNN architectures, including DenseNet121, ResNet50, and VGG16, in font types classification. According to the experimental results, all three models demonstrated good generalization capability, with good accuracy and low error in the training process. According to experimental results, VGG16 achieved the highest training accuracy (98.4%) with the lowest loss (0.0467), while DenseNet121 showed strong validation accuracy (95.47%). ResNet50, although achieving reasonable results, consistently lagged behind the other two models in both accuracy and stability. To evaluate the model performance, we calculate evaluation metrics on test data. According to the evaluation model result, the DenseNet121 reached the best accuracy (96.8%), VGG16 (96.4%), and ResNet50 obtained 92.9%. The precision, recall, and F1-scores across most font classes were high, particularly for fonts like Algerian, Monotype Corsiva, Rockwell, and Special Elite, where performance reached perfect scores. However, certain fonts such as Calibri and News Gothic were more challenging, showing lower precision and recall (0.82–0.83). In connectivity and sensitivity calculation, the DenseNet121 with a learning rate (0.01 proved optimal for DenseNet121, yielding the best trade-off between accuracy and loss stability. ResNet50 benefited from smaller learning rates (0.0001), while VGG16 was highly sensitive, performing poorly at high learning rates but excelling at lower ones. Future research can focus on expanding the dataset with more diverse and challenging font variations, applying data augmentation for better robustness, and exploring hybrid or ensemble models to combine the strengths of different CNN architectures. Transfer learning from large-scale text datasets and further hyperparameter optimization, including batch size and regularization strategies, may also improve performance. In addition, integrating advanced approaches such as attention mechanisms or transformer-based vision models could enhance fine-grained feature extraction, while real-time

implementation would enable practical applications in document analysis, digital forensics, and typography design.

References

[1] Y. J. "R-GNN: recurrent graph neural networks for font classification of oracle bone inscriptions," Heritage Science, vol. 12, 2024. doi: 10.1186/s40494-024-01133-4

[2] L. A. "A Proposed Font Distinguishing Element Categorization for Improvement of the Korean Font Classification System," Archives of Design Research, vol. 36, pp. 67-89, 2023. doi: 10.15187/adr.2023.08.36.3.67.

[3] P. J. "Hangeul Font Classification System Proposal of a Network Structure for Improved Font Search Result Similarity," Archives of Design Research, vol. 36, pp. 145-169, 2023. doi: 10.15187/adr.2023.05.36.2.145

[4] C. W. "Multi-loss siamese convolutional neural network for Chinese calligraphy font and style classification," Journal of Image and Graphics, vol. 28, pp. 2370-2381, 2023. doi: 10.11834/jig.220252

[5] V.D., & N. (2023). Extending OCR Model for Font and Style Classification. Lecture Notes in Networks and Systems 734 LNNS, 193-204. https://doi.org/10.1007/978-3-031-36886-8_16

[6] M. I. "Robustness of Contrastive Learning on Multilingual Font Style Classification Using Various Contrastive Loss Functions," Applied Sciences Switzerland, vol. 13, 2023. doi: 10.3390/app13063635

[7] L. M. "Enhancing Chinese Font Recognition with Convolutional Neural Networks," Proceedings 2023 International Conference on Networks Communications and Intelligent Computing Ncic 2023, pp. 291-295, 2023. doi: 10.1109/NCIC61838.2023.00055.

[8] Y. R. "Handwritten Font Classification Method Based on Ghost Imaging," Guangzi Xuebao Acta Photonica Sinica, vol. 51, 2022. doi: 10.3788/gzxb20225111.1111001.

[9] S. N. Rahmawati, E. W. Hidayat, and H. Mubarok, "Implementasi Deep Learning pada Pengenalan Aksara Sunda Menggunakan Metode Convolutional Neural Network," *Inser. Inf. Syst. Emerg. Technol. J.*, vol. 2, no. 1, pp. 46–58, 2021, doi: 10.23887/insert.v2i1.37405.

[10] E. A. Winardi and E. Hartati, "Identifikasi Aksara Katakana Menggunakan Convolutional Neural Network Arsitektur LeNet," *J. Algoritm.*, vol. 2, no. 2, pp. 92–101, 2022, doi: 10.35957/algoritme.v2i2.2359.

[11] C. C. Ukwuoma *et al.*, "Animal species detection and classification framework based on modified multi-scale attention mechanism and feature pyramid network," *Sci. African*, vol. 16, p. e01151, 2022, doi: 10.1016/j.sciaf.2022.e01151.

[12] B. Li and D. Lima, "Facial expression recognition via ResNet-50," *Int. J. Cogn. Comput. Eng.*, vol. 2, no. January, pp. 57–64, 2021, doi: 10.1016/j.ijcce.2021.02.002.

[13] N. Wagaa, H. Kallel, and N. Mellouli, "Improved Arabic Alphabet Characters Classification Using Convolutional Neural Networks (CNN)," *Comput. Intell. Neurosci.*, vol. 2022, 2022, doi: 10.1155/2022/9965426.

[14] D. Correia and S. Jardim, "A deep learning approach for face detection," *Procedia Comput. Sci.*, vol. 263, pp. 34–41, 2025, doi: 10.1016/j.procs.2025.07.005.

[15] S. S. Halim, B. Kanata, and S. I. Akbar, "The Klasifikasi Suara Paru-Paru Menggunakan Mel Frequency Cepstral Coefficient dan Convolutional Neural Network," *J. Bumigora Inf. Technol.*, vol. 6, no. 2, pp. 163–176, 2024, doi: 10.30812/bite.v6i2.4487.

[16] E. H. Rachmawanto and P. N. Andono, "Deteksi Karakter Hiragana Menggunakan Metode Convolutional Neural Network," *J. Nas. Pendidik. Tek. Inform.*, vol. 11, no. 3, pp. 183–191, 2022, doi: 10.23887/janapati.v11i3.50144.

[17] S. Arooj, S. Altaf, S. Ahmad, H. Mahmoud, and A. S. N. Mohamed, "Enhancing sign language recognition using CNN and SIFT: A case study on Pakistan sign language," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 36, no. 2, p. 101934, 2024, doi: 10.1016/j.jksuci.2024.101934.

[18] A. M. Ismael and A. Şengür, "Deep learning approaches for COVID-19 detection based on chest X-ray images," *Expert Syst. Appl.*, vol.. 164, no. September 2020, 2021, doi: 10.1016/j.eswa.2020.114054.

[19] M. Xu, S. Yoon, A. Fuentes, and D. S. Park, "A Comprehensive Survey of Image Augmentation Techniques for Deep Learning," *Pattern Recognit.*, vol. 137, p. 109347, 2023, doi: 10.1016/j.patcog.2023.109347.

[20] A. Mumuni and F. Mumuni, "Data augmentation: A comprehensive survey of modern approaches," *Array*, vol. 16, no. August, p. 100258, 2022, doi: 10.1016/j.array.2022.100258.

[21] S. Xu, C. Guo, Y. Zhu, G. Liu, and N. Xiong, "CNN-VAE: An intelligent text representation algorithm," *J. Supercomput.*, vol. 79, no. 11, pp. 12266–12291, 2023, doi: 10.1007/s11227-023-05139-w.

[22] Q. Zhu, X. Jiang, and R. Ye, "Sentiment Analysis of Review Text Based on BiGRU-Attention and Hybrid CNN," *IEEE Access*, vol. 9, pp. 149077–149088, 2021, doi: 10.1109/ACCESS.2021.3118537.

[23] T. Chen and Z. Xie, "Japanese Short Text Classification Based on CNN-BiLSTM-Attention," *Procedia Comput. Sci.*, vol. 262, pp. 320–329, 2025, doi: 10.1016/j.procs.2025.05.059.
[24] A. Nagar, A. Bhasin, and G. Mathur, "Text classification using gated fusion of n-gram features and semantic features," *Comput. y Sist.*, vol. 23, no. 3, pp. 1015–1020, 2019,
[25] Y. Wu, J. Li, C. Song, and J. Chang, "Words in Pairs Neural Networks for Text Classification," *Chinese J. Electron.*, vol. 29, no. 3, pp. 491–500, 2020, doi: 10.1049/cje.. 2020.03.005.
[26] C. Grazian, Q. Jin, and G. Tangari, "Assessing the invertibility of deep biometric representations: Investigating CNN hyperparameters for enhanced security against adversarial attacks," *Expert Syst. Appl.*, vol.. 264, no. November 2024, p. 125848, 2025, doi: 10.1016/j.eswa.2024.125848.
[27] S. Sharma and K. Guleria, "A Deep Learning based model for the Detection of Pneumonia from Chest X-Ray Images using VGG-16 and Neural Networks," *Procedia Comput. Sci.*, vol. 218, pp. 357–366, 2022, doi: 10.1016/j.procs.2023.01.018.
[28] A. A. Khan, "Balanced Split: A new train-test data splitting strategy for imbalanced datasets," arXiv preprint arXiv:2212.11116, 2022.