IJASST
International Journal of
Applied Sciences and Smart Technologies

Volume 07
Issue 02

p-ISSN 2655-8564, e-ISSN 2685-943, pp. 417-432, 2025

Braille Pattern Detection Modeling Using Inception V3 Architecture Using Median Filter Implementation and Segmentation

Abdul Latip[1*], Siti Yuliyanti [1], Muhammad Al-Husaini [1]

[1] *Department of Informatics, Faculty of Engineering, Siliwangi University, Tasikmalaya, West Java, Indonesia*

[*] *Corresponding Author: abdhul.latyp@gmail.com*

(Received 19-06-2025; Revised 27-07-2025; Accepted 25-08-2025)

Abstract

This study aims to detect Braille letter patterns using the InceptionV3 architecture combined with the application of median filter and image segmentation. The dataset consists of 4,160 Braille images, with an average of 160 images for each letter from A to Z. The data is divided into 3,900 images for training, which are then split into 3,120 images for training and 780 images for validation, and 260 images are used for testing. Each image is resized to 299x299 pixels before being fed into the model. This study uses 100 epochs and applies early stopping to avoid overfitting. Two learning rate values are tested, namely 0.001 and 0.0001. The results show that the application of a median filter and segmentation significantly improves model performance, producing better accuracy, precision, recall, and F1 values compared to models without these techniques. At a learning rate of 0.001, the model achieves 99.65% accuracy, 99.62% precision, and 99.61% recall. On the other hand, without a median filter and segmentation at a learning rate of 0.0001, although accuracy and precision decreased, the values still reached 99.65% and 99.62%.

Keywords: Brailler, Inception V3, Learning Rate, Median Filter, Segmentation

1 Introduction

Braille is a standard writing system for the blind, where each character is represented by a combination of dots in a specific pattern that can be felt with the fingertips [1]. Given the challenges in reading Braille, especially the need for high finger sensitivity, deep learning technology can play an important role in assisting automatic Braille translation [2].

Deep learning, especially Convolutional Neural Networks (CNN), has proven effective in image and pattern recognition [3]. Deep learning uses artificial neural networks with many layers to automatically learn data representations. CNN, inspired by the structure of human neural networks, works with two main phases: backpropagation for training and feedforward for image classification. Before classification, preprocessing processes such as bagging and cropping are used to focus on the objects to be classified [3]. This CNN has many architectures, one of which is InceptionV3. InceptionV3 is one of the most famous CNN architectures. This architecture is known for being easy to train and very accurate in image classification. InceptionV3 is capable of capturing features at different scales, from the smallest to the most complex, by using a combination of convolutional layers with various kernel sizes [3].

Previous studies have shown that the Inception V3 architecture can achieve high accuracy in Braille recognition, with accuracy values reaching 95.03% to 99.87% [4]. However, the detection performance of Braille models is often affected by noise in images. Therefore, preprocessing techniques such as median filtering and image segmentation become very important [5]. The median filter, which is a nonlinear filter, is effective in reducing noise without losing important image details by replacing the processed pixel value with the median value of a group of pixels. Image segmentation is also important to separate important elements from the background, thereby increasing focus on Braille patterns [6].

However, the performance of Braille detection models can be affected by noise in images. Therefore, preprocessing techniques such as median filtering and image segmentation become crucial [4]. Median filtering is effective in reducing noise without destroying important details, while segmentation helps separate Braille patterns from irrelevant backgrounds. Previous studies have shown that the combination of median filtering and segmentation can improve detection accuracy [5].This study will explore the use of the Inception V3 architecture in Braille character recognition. Inception V3, as one of the effective CNN architectures in complex object recognition, can be enhanced by the integration of median filtering and segmentation to further improve the accuracy of Braille characters.

2 Related Work

In several studies, researchers have made great progress in developing BCR systems. To improve the accuracy and efficiency of these systems, they have used various techniques, such as machine learning, artificial neural networks (ANNs), and CNNs. These techniques allow Braille readers to interact more easily with digital devices, such as computers and smartphones.

In previous studies, such as [6], this study proposes a new approach for automatic recognition of Braille characters, which consists of two main stages: image alignment and enhancement using preprocessing techniques, and character recognition using a lightweight convolutional neural network (CNN). This approach replaces some modules in CNN with IRB blocks to reduce computational costs, resulting in an efficient and accurate model. Experiments on English Braille and Chinese double-sided Braille (DSBI) datasets show prediction accuracies of 95.2% and 98.3%. This method is more robust and effective than current approaches, with a prediction time of 0.01s for English Braille images and 0.03s for DSBI.

Meanwhile, [7] discusses the conversion of Braille to English text using deep learning. In this study, 26 English Braille images are used as the dataset that has gone through the segmentation process. The proposed method converts Braille visuals to English text using convolutional neural network (CNN) models such as LeNet, VGG-16, DenseNet121, ResNet50, and Inceptionv3. Among these models, the Inceptionv3-based system showed a high prediction accuracy rate, reaching 92%. Experimental results showed that the proposed Braille character recognition method produced accurate results.

Research [2] showed that combining multiple models can improve accuracy, but this method is often limited to a specific combination of models and datasets. In recent studies, models such as DarkNet-53, GoogleNet, SqueezeNet, and DenseNet-201 were incorporated into a more general transfer learning-based ensemble approach. This group approach achieved high F1 scores (89.42%, 99.58%, and 97.11% on HBO, BC, and AB datasets), and lower error values compared to a single model, such as DarkNet-53, which only achieved an F1 score of 87.54%. In addition, this method helps the development of more efficient Braille-based assistive technologies.

3 Methods

This research methodology is designed to develop a Braille pattern detection model using the Inception V3 architecture by implementing median filter and segmentation as part of the data preprocessing process. The steps of this research methodology are explained as follows, shown in Fig. 1.

This research includes several stages, namely:

Step 1: Literature Study

Literature study was conducted from data collection to evaluation to guide the research process. The sources include books, scientific journals, and other scientific works, related to detection using Inceptio-V3, median filter, segmentation and other related topics.

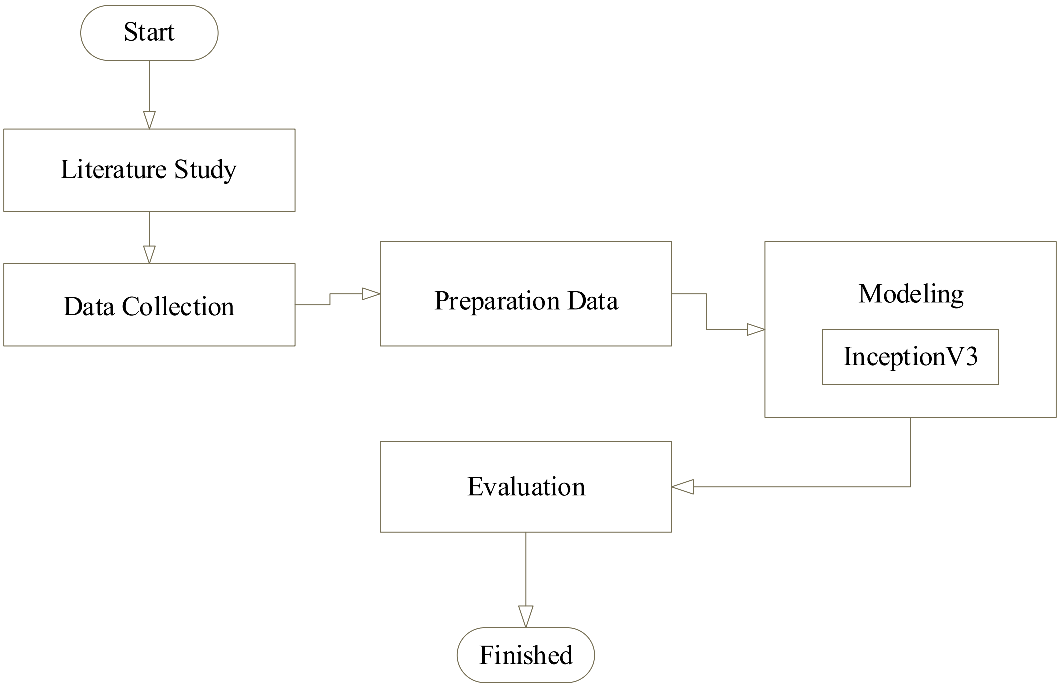

Figure 1. Research stages

Step 2: Data Collection

This study collects datasets using datasets taken from the github website belonging to user HelenGezahegn [4]. The data used has been filtered to select good quality data, so that it has been reduced and consists of 4,160 data samples, with an average of 160 samples for each character from A to Z, the image will be divided into 2 parts 3900 training data and 260 testing data, and for 3900 data divided into 3120 as training data and 780 as validation data [8].

Step 3: Data Preparation

Data preparation in this study includes several processes to ensure that the images used in the Convolutional Neural Network model using the inception V3 architecture have the appropriate quality and format. The first step taken is to resize the image. The image with the original size $H \times W$ is resized to the standard dimensions $H' \times W'$. This process can be formulated as:

$$I'(x,y) = I\left(\frac{H\prime}{H} \times x, \frac{W\prime}{W} \times y\right) \quad (1)$$

After resizing, the image is converted from color format (RGB) to grayscale using the following formula [9]:

$$\mathrm{Y} = 0.299\mathrm{R} + 0.587\mathrm{G} + 0.144B \quad (2)$$

Where R , G , and B are the red, green, and blue channel values at pixel (x , y).

To remove noise, a median filter is applied to the grayscale image. Median filtering is done by taking the median of the intensity values in the filter window $k \times k$ [10]:

$$I_{filtered}(x,y) = median\{I_{gray}(x+i, y+j)\}(3)$$

$$\text{for all } i , j \in [-k/2 , k/2]$$

The segmentation process is carried out using k-means clustering [11] with the following steps. Determine the number of k clusters. Randomly select k data points as the initial centroids.Once the initial centroids are determined, calculate the distance of each data point to the nearest centroid using the Euclidean Distance formula:

$$d\,(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (4)$$

Where:

d(x,y) = Euclidean distance between points x and y.

x and y = Two points in n-dimensional space whose distances will be calculated.

Group members are identified based on the smallest data distance to the centroid. Then calculate the new centroid using the centroid finding formula:

$$\text{centroid}_i = \frac{\sum_{i=1}^{n} x_j}{n} \tag{5}$$

Where:

Centroid = Centroid for the i-th cluster.

n = Total number of data points in the i-th cluster.

Repeat steps 2 through 4 until no more data changes cluster.

Step 4. Modeling

This study uses Convolutional Neural Network (CNN) with Inception-V3 model architecture, shown in Fig. 2. In the Inception-V3 architecture layer [13], all layers before the fully connected layer in each architecture are frozen first to maintain the weight parameters obtained from ImageNet. The fully connected layer is removed because transfer learning will be applied to the model to train a new fully connected layer that will be used to classify datasets with different numbers of categories. After freezing the layers before the fully connected layer, there is a flatten layer followed by two fully connected layers [14]. Braille pattern detection model architecture using InceptionV3 is shown in Fig. 3.

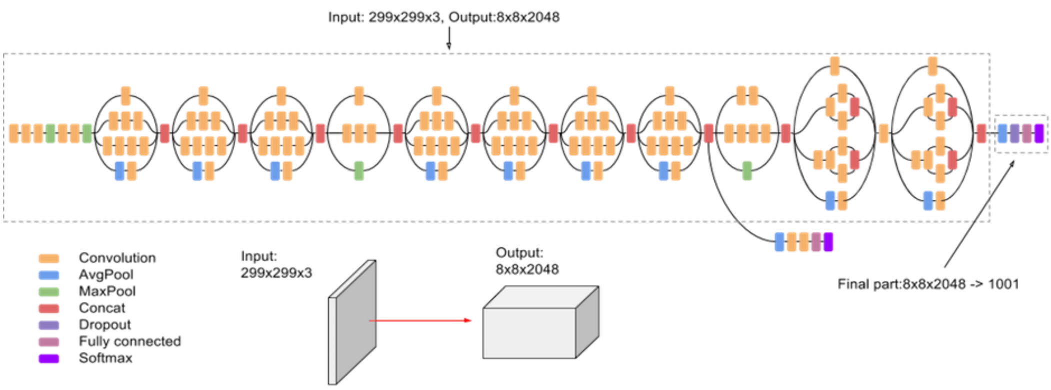

Figure 2. Inception V3 Architecture

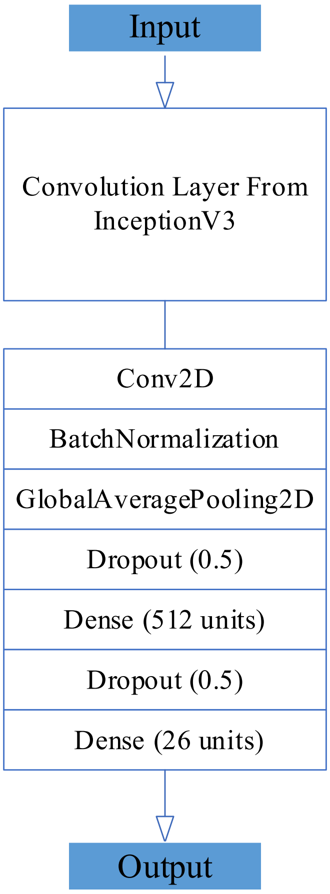

Figure 3. Braille Pattern Detection Model Architecture Using InceptionV3

Fig. 2 is a convolutional neural network (CNN) model architecture that uses convolution layers from InceptionV3 for feature extraction from input images. After that, Conv2D is applied to the image or spatial data. followed by BatchNormalization to stabilize and speed up training. And GlobalAveragePooling2D, which converts the convolution output into a feature vector. Next, a dropout layer is applied to reduce overfitting, followed by several dense layers with 512 neuron units. Then there is another Dropout layer and finally, a thick layer with 64 and 26 neuron units. 26 neuron units are used for classification into 26 output classes. This architecture combines the powerful feature extraction capabilities of InceptionV3 with regularization and normalization techniques to improve model performance and generalization.

Step 5. Evaluation

At this stage, various scenarios of Recall, Precision, F1 Score, and Accuracy settings are evaluated to improve the modeling results [17]. Finally, it will be known which scenario produces the best accuracy value and the lowest error rate for the braille pattern detection model using the Inception V3 architecture by applying median filter and segmentation. The formulas for Recall, Precision, F1 Score, and Accuracy are as follows:

$$\text{Akurasi} = \frac{TP + TN}{TP + FP + FN + TN} \tag{6}$$

$$\text{Presisi} = \frac{TP}{TP + FP} \tag{7}$$

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

$$F1\ Score = 2 \times \frac{Recall\ x\ Presisi}{Recall + Presisi} \tag{9}$$

Explanation of the equations TP (True Positive), TN (True Negative), FP (False Positive), and FN (False Negative).

4 Results and Discussions

In this study, the InceptionV3 model pre-trained on the ImageNet dataset was modified by adding new layers. The model was then optimized by specifying an image size of 299x299 [18], a batch size of 32, a number of epochs of 100, and a learning rate of 0.001 and 0.0001. Early stopping was also used to prevent overfitting. These parameters are important to ensure the model learns well from the data and achieves rapid convergence.

In the training phase, the InceptionV3 model implemented with median filter and segmentation showed quite good performance. During the training process, accuracy and loss metrics were observed to evaluate the model performance. The training and validation accuracy graphs showed a significant increase at the beginning of the epoch and stabilization at the end of the training process as shown in Fig. 4.

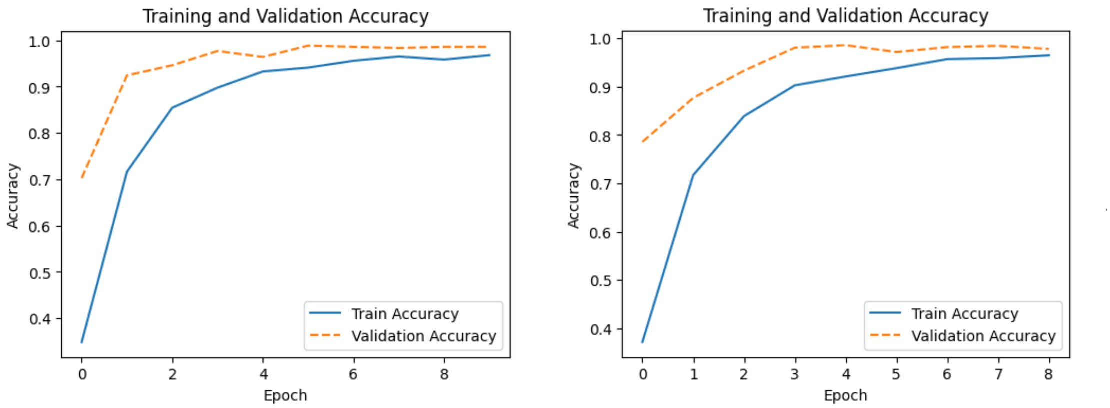

Figure 4. Accuracy graph trains the model with learning rate 0.001

The Fig. 4 above show the results of the training process with a learning rate of 0.001, presented in the form of graphs. In both graphs, dots represent the training data, while lines represent the validation data. The graph on the right demonstrates that the training accuracy reaches above 0.9664, with the validation accuracy showing similar results. Similarly, the graph on the left shows that both the training and validation accuracy exceed 0.9669.

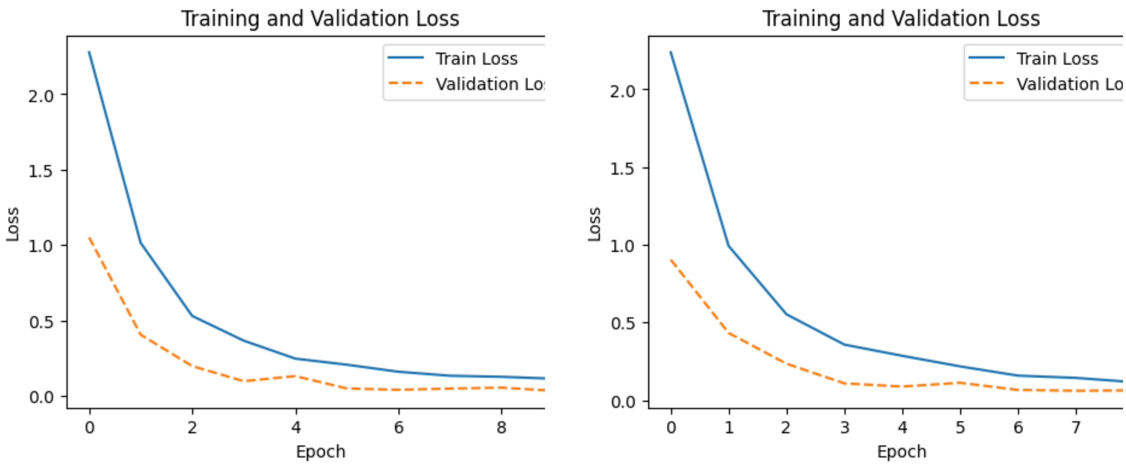

Figure 5. Loss graph of training the model with a learning rate 0.001

The Fig. 5 depict the results of the training process with a learning rate of 0.001, shown in graph form. In both graphs, dots represent the training data, and lines represent the validation data. The graph on the right shows that the training loss reached 0.1144. Similarly, the graph on the left indicates that the training loss reached 0.1289, then training was also carried out using a learning rate of 0.0001.

The Fig. 6 illustrate the results of the training process, presented as graphs where dots represent training data and lines represent validation data. The graph on the right shows that both training and validation accuracy exceed 0.9779. Similarly, the graph on the left indicates that the training accuracy reaches 0.9822.

The Fig. 7 depict the results of the training process with a learning rate of 0.001, shown in graph form. In both graphs, dots represent the training data, and lines represent the validation data. The graph on the right shows that the training loss reached 0.1400. Similarly, the graph on the left indicates that the training loss reached 0.1497.

After the training process is complete, the next step is to test the model. At this stage, the trained data is compared with the data that has been prepared during the preprocessing process. The dataset used for testing is 260 datasets.

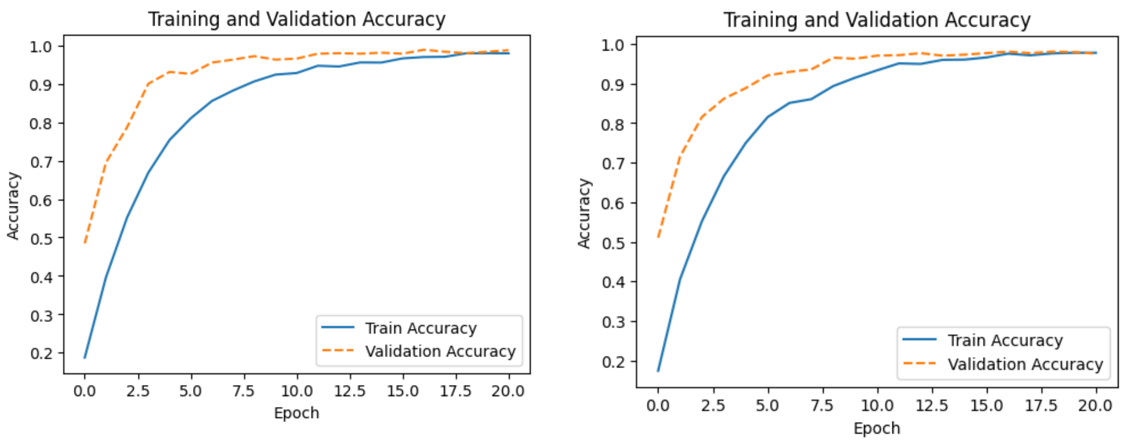

Figure 6 Accuracy graph train the model with learning rate 0.0001

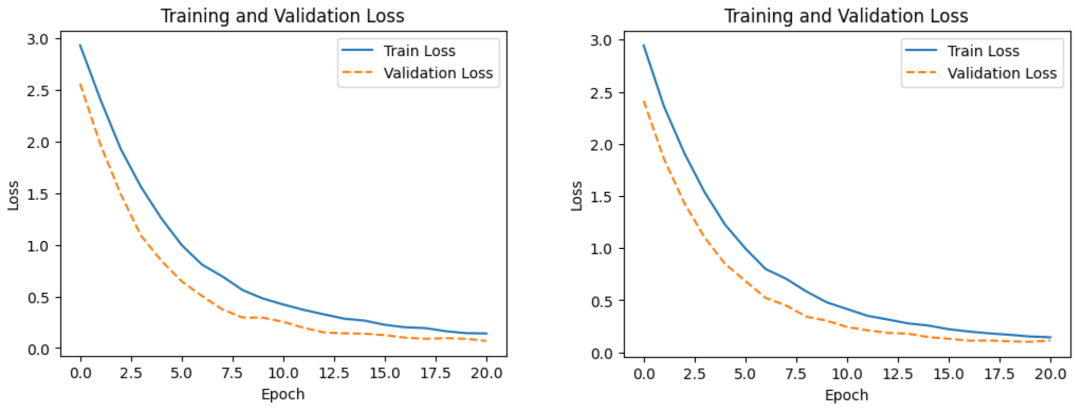

Figure 7. loss graph of training the model with learning rate 0.0001

Table 1. Performance evaluation metrics

Learning rate	Accuracy	precision	recall	f1 Score
Using Median Filter and Segmentation				
0.001	0.9965	0.9962	0.9961	0.9962
0.0001	0.9930	0.9923	0.9923	0.9923
Not Using Median Filter and Segmentation				
0.001	0.9936	0.9923	0.9924	0.9923
0.0001	0.9865	0.9846	0.9845	0.9846

Based on Table 1, it can be seen that the use of median filter and segmentation results in higher accuracy, precision, recall, and F1 score compared to not using both techniques, especially at a learning rate of 0.001. At this learning rate, the highest accuracy is achieved, namely 99.65% with precision and recall of 99.62% and 99.61%, respectively. In contrast, without median filter and segmentation, model performance decreases, especially at a learning rate of 0.0001, where accuracy drops to 98.65%, and precision and recall reach 98.46% and 98.45%. This shows that median filter and segmentation play an important role in improving model performance, especially at higher learning rates.

Table 2. Comparison table with previous research

Research	Methods	Accuracy
[1]	EfficientNetV2M and InceptionV3	82.07%
[2],[4]	Cnvolutional Neural Network (CNN).	81.54%
[3]	CNN using Segementation	95.77%
Our Reserach	InceptionV3 using Median Filter and Segmentation	99.65%

Table 2 presents a comparison between the proposed method and previous research in \terms of classification accuracy. The first referenced study employed EfficientNetV2M and InceptionV3 architectures, achieving an accuracy of 82.07%. The second study utilized a Convolutional Neural Network (CNN) model with image segmentation, resulting in an accuracy of 95.77%. In contrast, our proposed approach, which combines InceptionV3 with median filtering and Segmentation preprocessing, outperformed the others with a significantly higher accuracy of 99.65%. This demonstrates the effectiveness of the proposed method in enhancing classification performance.

5 Conclusions

The InceptionV3 model shows excellent performance with high training accuracy on all tested configurations, in this study the use of median filter and segmentation produces higher accuracy, precision, recall, and F1 scores compared to when not using both techniques, especially at a learning rate of 0.001, where the accuracy reaches 99.65%, and precision and recall reach 99.62% and 99.61%, respectively. Conversely, without median filter and segmentation, the model performance decreases, especially at a learning rate of 0.0001, where the accuracy reaches 99.65%, and precision reaches 99.62%.

Acknowledgements

The author would like to express his deepest gratitude to Siti Yuliyanti and Muhammad Al-Husaini, as academic supervisors, for their invaluable guidance and support during this research. Gratitude is also extended to the Informatics Study Program, Siliwangi University, which has facilitated the research environment. Special gratitude to all colleagues and friends who have contributed in the form of corrections, feedback, and moral support during the preparation of this work.

References

[1] K. M. Farrand, K. Koehler, and A. Vasquez, “Literary Braille Instruction: A Review of University Personnel Preparation Programs,” https://doi.org/10.1177/0145482X221130356, vol. 116, no. 5, pp. 617–628, Nov. 2022, doi: 10.1177/0145482X221130356.

[2] N. Elaraby, S. Barakat, and A. Rezk, “A generalized ensemble approach based on transfer learning for Braille character recognition,” Inf Process Manag, vol. 61, no. 1, p. 103545, Jan. 2024, doi: 10.1016/J.IPM.2023.103545.

[3] Jahandad, S. M. Sam, K. Kamardin, N. N. Amir Sjarif, and N. Mohamed, “Offline Signature Verification using Deep Learning Convolutional Neural Network (CNN) Architectures GoogLeNet Inception-v1 and Inception-v3,” Procedia Comput Sci, vol. 161, pp. 475–483, Jan. 2019, doi: 10.1016/J.PROCS.2019.11.147.

[4] M. F. Herlambang, A. N. Hermana, and K. R. Putra, “Pengenalan Karakter Huruf Braille dengan Metode Convolutional Neural Network,” Systemic: Information System and Informatics Journal, vol. 6, no. 2, pp. 20–26, 2021, doi: 10.29080/systemic.v6i2.969.

[5] H. Tang, R. Ni, Y. Zhao, and X. Li, “Median filtering detection of small-size image based on CNN,” J Vis Commun Image Represent, vol. 51, pp. 162–168, Feb. 2018, doi: 10.1016/J.JVCIR.2018.01.011.

[6] R. H. Hasan, I. S. Aboud, and R. M. Hassoo, “Braille Character Recognition System : Review,” vol. 18, no. 1, pp. 30–39, 2024.

[7] A. Al-Salman and A. AlSalman, "Fly-LeNet: A deep learning-based framework for converting multilingual braille images," Heliyon, vol. 10, no. 4, p. e26155, Feb. 2024, doi: 10.1016/J.HELIYON.2024.E26155.

[8] H. J. Andriansyah, Marindo, "Pengenalan Karakter Braille Memanfaatkan Convolutional Neural Network," vol. 4, no. 1, pp. 41–54, 2021.

[9] M. Minarni, K. Rizka, Y. Yuhendra, D. W. T. Putra, and I. Warman, "Penerapan Deteksi Sobel Berbasis Algoritma Backpropagation pada Pengenalan Pola Huruf Vokal," Jurnal Minfo Polgan, vol. 12, no. 2, pp. 1829–1839, 2023, doi: 10.33395/jmp.v12i2.13019.

[10] Mardiah et al, "Application of Image Median Filter and Histogram Equalization on Old Building Images Pendahuluan," vol. 1, no. September, pp. 0–7, 2023.

[11] M. A. Rajab and L. E. George, "Stamps extraction using local adaptive k- means and ISODATA algorithms," Indonesian Journal of Electrical Engineering and Computer Science, vol. 21, no. 1, pp. 137–145, 2021, doi: 10.11591/ijeecs.v21.i1.pp137-145.

[12] A. C. O. Reddy and K. Madhavi, "Hierarchy based firefly optimized k-means clustering for complex question answering," Indonesian Journal of Electrical Engineering and Computer Science, vol. 17, no. 1, pp. 264–272, 2019, doi: 10.11591/ijeecs.v17.i1.pp264-272.

[13] Jahandad, S. M. Sam, K. Kamardin, N. N. Amir Sjarif, and N. Mohamed, "Offline Signature Verification using Deep Learning Convolutional Neural Network (CNN) Architectures GoogLeNet Inception-v1 and Inception-v3," Procedia Comput Sci, vol. 161, pp. 475–483, Jan. 2019, doi: 10.1016/J.PROCS.2019.11.147.

[14] S. Raghuwanshi, A. Sukhad, A. Rasool, V. K. Meena, A. Jadhav, and K. Shivakarthik, “Early Detection of Brain Tumor from MRI Images Using Different Machine Learning Techniques,” Procedia Comput Sci, vol. 235, pp. 3094–3104, Jan. 2024, doi: 10.1016/J.PROCS.2024.04.293.

[15] Maad M. Mijwel, “Artificial Neural Networks Advantages and Disadvantages,” Mesopotamian journal of BigData, vol. Vol. (2021, pp. 29–31, 2021.

[16] S. Samidin and A. Fadjeri, “Klasifikasi Gambar Batu-Kertas-Gunting Menggunakan Convolutional Neural Network dengan Fungsi Callback untuk Mencegah Overfitting,” Jurnal Penelitian Inovatif, vol. 4, no. 2, pp. 785–794, 2024, doi: 10.54082/jupin.413.

[17] N. Saqib, K. F. Haque, V. P. Yanambaka, and A. Abdelgawad, “Convolutional-Neural-Network-Based Handwritten Character Recognition: An Approach with Massive Multisource Data,” Algorithms, vol. 15, no. 4, 2022, doi: 10.3390/a15040129.

[18] V. C. P. A. M. L. S. Shilpi Yadav, “Recognition of Pandemic virus Covid-19 using Inception-V3,” International Journal of Advanced Science and Technology, vol. 29, no. 12s, pp. 1675–1679, Jun. 2020, Accessed: Sep. 10, 2024. [Online]. Available: http://sersc.org/journals/index.php/IJAST/article/view/23906

This page intentionally left