TELKOMNIKA Telecommunication Computing Electronics and Control
Vol. 22, No. 4, August 2024, pp. 931~940
ISSN: 1693-6930, DOI: 10.12928/TELKOMNIKA.v22i4.25849

Convolutional neural network enhancement for mobile application of offline handwritten signature verification

Wifda Muna Fatihia, Arna Fariza, Tita Karlita
Department of Informatics and Computer Engineering, Politeknik Elektronika Negeri Surabaya, Surabaya, Indonesia

Article Info

Article history:

Received Nov 8, 2023
Revised Mar 5, 2024
Accepted Mar 29, 2024

Keywords:

Convolutional neural network
Genuine
Image classification
Mobile
Signature

ABSTRACT

The increase in signature forgery cases can be attributed to the escape of forged signatures from manual signature verification systems. Researchers have developed various machine learning and deep learning methods to verify the authenticity of signatures, one of which uses convolutional neural networks (CNNs). This research aims to develop a mobile application for handwritten signature verification using CNN architecture by adding a batch normalization technique to its layer. The performance of our proposed method achieved a verification accuracy of 86.36%, with a 0.061 false acceptance rate (FAR), 0.303 false rejection rate (FRR), and 0.182 equal error rate (EER), which is compatible to be embedded in smartphones. However, there is still a need for further development of the CNN model and its integration with mobile applications.

Corresponding Author:

Arna Fariza
Department of Informatics and Computer Engineering, Politeknik Elektronika Negeri Surabaya
Jalan Raya ITS, Keputih, Sukolilo, Surabaya 60111, Indonesia
Email: arna@pens.ac.id

1. INTRODUCTION

A signature refers to a person's distinct and characteristic method of inscribing their name using their hand. The distinctive configuration of a signature serves as a distinct symbol that embodies identity, ensuring that each individual possesses a distinct pattern of signature. A signature serves as a means of authentication, expressing an approval of a policy or the document contents [1]. Signature is achieved by the distinctiveness, consistency, and clarity of the signature pattern, differentiates one person from another [2], [3]. One of the various applications of signatures encompasses its usage as a means of authentication for legal documents, including contracts, statements, and agreements [4].

Handwritten signatures provide authentication for various countries in North America, Europe, Asia, and Africa. In Indonesia, the government recognizes signatures as a valid and legal method of verifying identity in relation to a document [5]. Signatures serve as a means of biometric authentication for individuals, providing an alternative to facial recognition, iris scanning, fingerprinting, and voice recognition, all of which require the use of specific devices [3], [4], [6]. Despite the availability of modern and digital electronic authentication methods, handwritten signatures continue to be used as a means of authentication as legal precedents in numerous legal systems [7], [8]. Courts and governments are familiar with signature authentication, and laws and regulations often accommodate handwritten signatures [9]. The offline signature verification poses significant challenges in verificaton field. The inherent immutability of static data prevents the storage of dynamic information, which often contributes to the information processing [5], [10]-[12]. Offline verification is static, mainly by scanning signatures converted to digital signature images, excluding any dynamic information [2], [5]. An individual's signature can often be the result of variations in their circumstances and conditions while signing. Depending on the equipment used, such as paper and stylus, offline signature results can be vary. Offline signatures solely encompass the visual representation of a signature, posing challenges in the verification process [13], [14]. Human senses have limitations to identify signatures with similar patterns, poses a challenge in accurately verifying the authenticity of a signature [12]. Forging a signature can result in financial material, and psychological losses for the relevant parties involved in the document.

According to the problems mentioned above, performing automatic verification is possible to prevent instances of signature forgery. The domain of signature verification has been extensively studied by many scientists worldwide over a prolonged time. The recognition of its difficulty has been widely acknowledged within the discipline of computer vision research [15]. In this study, we aimed to construct a mobile-based application, enabling it to verify handwritten signature images by enhancing usability and facilitating high mobility. We integrated it with our proposed convolutional neural network (CNN) model with a substantial accuracy level. The implementation of a robust signature verification system can effectively mitigate the risk of a forged signature by applying a well-tested and accurate method. This feature proves advantageous across diverse domains, such as the verification of financial file signatures associated with banking institutions, the authentication of signatures on correspondence or legal documents involved in property transactions, and other similar applications. Our contribution can be summarized as:

a. Enhance CNN architecture by refining the process of training data generation in order to maximize the model.
b. Modify CNN architecture by adding batch normalization technique to see how reliable the model used in mobile application embedded systems.
c. Develop a mobile-based application with efficient verification of signatures in order to assist various sectors, including governments and institutions, in reducing the widespread practice of signature forgery.

The following sections of the paper are structured in the following manner: section 2 comprehensively explains the theoretical underpinnings that form the basis for the approaches utilized in the proposed method. Section 3 explains the proposed method. Section 4 presents the findings and performance evaluation, while section 5 covers the final remarks of the study.

2. LITERATURE REVIEW

Previous studies have employed various methods to authenticate signature images using deep learning or machine learning algorithms. Bibi *et al.* [16] studied a comprehensive analysis of the present trends, challenges, and opportunities of machine learning techniques used in the discipline of signature verification. Dey *et al.* [17] proposed an architecture called convolutional siamese network. They used open access signature dataset which resulted 77.76% accuracy when testing with dataset from the Grupo de Procesado Digital de Señales (GPDS) synthetic signature corpus, 85.90% accuracy with the hindi database, 86.11% accuracy with the bengali database, 88.79% accuracy with the GPDS 300 signature corpus database, and 100% accuracy with the cedar signature database.

Research by Ratna *et al.* [18] gained 83% achieved an average accuracy of 83% when trained on 22 genuine signature images using the Persian dataset. There was a 94% increase in accuracy while trained on nine genuine signature images. CNN is used to identify forged signatures, and autoencoding is used to generate forged signatures from existing original signatures. Jahandad *et al.* [19] compared CNN with GoogLeNet inception-v1 and inception-v3 architecture. The researchers [20], [21], offline signature images are verified using a machine learning technique. The accuracy obtained from twenty users for Inception-v1 is 83%, the equal error rates (EER) is low with a value of 17, while with Inception-v3 it reaches an accuracy value of 75% with an EER of 24. These results are better when compared to existing research. Inception-v1 shows better classification results, training time, and fewer operations than Inception-v3 for 2D signature images. Inception-v3 performs better than Inception-v1 for ImageNet image classification. Our proposed method is based on a CNN in conjunction with batch normalization approaches. The following section provides a brief overview of these topics.

2.1. Convolutional neural network

An approach for classifying signature images is. CNN takes the original raw visual data and extracts the features directly. Rather than being pre-trained, the linked features will develop throughout training. This automatic feature extraction method is the best model for learning task in computer vision [18], [22], [23]. Three layers typically make up the CNN architecture as:

a. Convolutional layer: a series of convolution operations. In image processing, convolution is employed on the image inputted to extract distinctive characteristics.
b. Pooling layers: to decrease the dimensions of an image. This layer partition the output of the preceding convolution layer into multiple smaller grids and reorganize reduced picture matrix. During a subsequent phase, just the most significant features will undergo processing.

c. The fully connected layer: to modify the dimensions of an image in order to enable linear classification.

2.2. Batch normalization

Sergey Ioffe and Christian Szegedy, researchers affiliated with Google Inc, introduced batch normalization in 2015. It is a technique to significantly enhancing the training process of deep neural networks (DNN). This is achieved by a normalization process that adjusts the means and variances of the inputs at each layer [24], [25]. Batch normalization is the normalization of the layer input inside the network to lower the training duration. It extends the normalization process to include all layers of the network. The normalization is performed for every mini-batch. Although there are some common design concepts between dropout and batch normalization approaches, extensive research findings have demonstrated that each method possesses distinct capabilities for enhancing deep learning. Batch normalization adjusts the unit values for each batch, and the random creation of batches during training provides additional noise to the training process. The training process are stabilized by the normalization, so the learning rate could be increased. It also reduces or remove the use of dropout rates, as the effect of regularization effect [26].

3. METHOD

The proposed system is a forged and genuine signature classification system with several development stages. Figure 1 is the system design of this study. It consists of five main parts: data collection, data pre-processing, classification stage, system evaluation, and model implementation. Each stage has a crucial role in building the system.

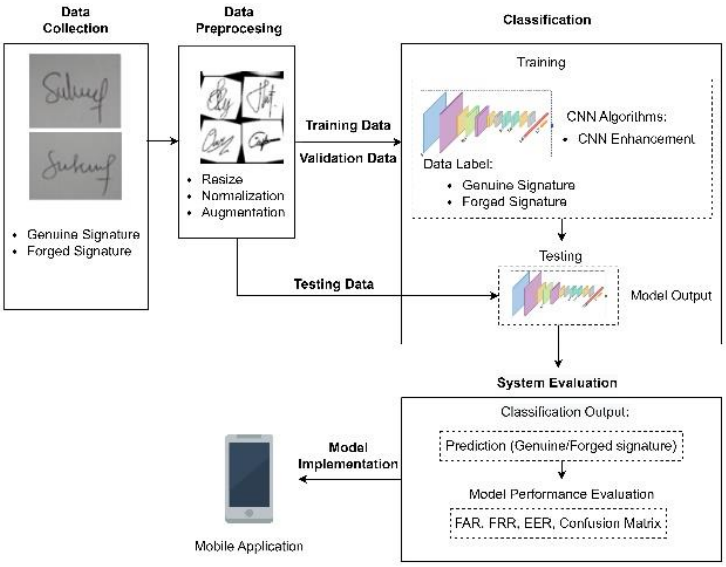


Figure 1. Proposed design system

3.1. Data collection

The dataset used in this study comprises signature images collected by the author. It includes signatures from 101 IDs, each contributed three genuine and three forged signatures. The signatures were generated by placing the individuals' signatures onto images of signatures received from the general public. The dataset was obtained using a combination of direct collection by researchers and a Google Form, wherein each participant completed the form and uploaded three genuine signatures. At that time, some individuals engaged in forging three forged signatures. Figures 2(a)-(e) shows the example of collected signature image.

(a) (b)

Figure 2. Raw signature image example; (a) genuine signature, and (b) forged signature

The raw signature images are shown in Figures 2(a) and (b). Figure 2(a) is an example of genuine signatures. Figure 2(b) is an example of forged signatures. From these raw images, there are several things that will affect the image processing stage, including the use of different paper, lighting, and pen used to sign. Therefore, an image pre-processing stage is needed before entering the classification stage to reduce bias.

3.2. Data pre-processing

The training dataset receives pre-processing to extract the primary features the network will use for signature identification. The primary features is a numerical convolution value to distinguish between distinct images derived from the feature extraction procedure within the architecture. The feature extraction involves the extraction of pertinent features from input data, such as images or signals, through the utilization of convolutional layers. At this stage, pre-processing and augmentation of the signature image data is carried out as follows:

a. Resize: altering the image dimensions to a specific pixel size corresponding to the dimensions of the initial convolution layer.
b. Image normalization: the image passes a conversion process, transforming into a single channel representation consisting of binary values, specifically 0 and 1. In this representation, the value 0 signifies the image’s background, while the value 1 signifies the presence of informative features within the image.
c. Augmentation: enhances the training process by incorporating additional sample signature variants, improving the network’s performance [23], [27].

3.3. Classification using CNN enhancement

The classification stage consists of data training and testing stages. During the training, the dataset used consist of training and validation data. This partitioning will be carried out based on a predetermined split ratio, where 90% was allocated for training purposes, and the 10% rest was designated for validation purposes. The training dataset constitutes a subset of the larger dataset comprising handwritten signature images used on the training stage to build the network. The validation dataset is a crucial part of the model validation process to mitigate the risk of overfitting. CNN architecture was used with an additional number of Batch Normalization shown in Table 1. A CNN model consists of convolutional, pooling, and fully connected layer. The total parameters used for this model was 542,142.

Table 1. CNN enhancement layers and parameters

Layer (type)	Output shape	Parameters
Convolution 1	[-1, 4, 101, 101]	20
Batch normalization	[-1, 4, 101, 101]	8
ReLU	[-1, 4, 101, 101]	0
Convolution 2	[-1, 8, 102, 102]	136
Batch normalization	[-1, 8, 102, 102]	16
ReLU	[-1, 8, 102, 102]	0
Max pooling	[-1, 8, 51, 51]	0
Convolution 3	[-1, 16, 52, 52]	528
Batch normalization	[-1, 16, 52, 52]	32
ReLU	[-1, 16, 52, 52]	0
Max pooling	[-1, 16, 26, 26]	0
Max pooling	[-1, 16, 13, 13]	0
Linear 1	[-1, 200]	541,000
Linear 2	[-1, 2]	402

As shown in Table 1, the CNN Enhancement has three convolutional layers, ReLU as the activation method, and two pooling layers. The batch normalization layers added to fasten the training process, normalize the data, and avoid overfitting. On the testing stage, developed model in the preceding training stage will be evaluated using testing data. The testing dataset underwent pre-processing procedures to transform the signature image into a binary image and mitigate bias variables, such as the image pixel size. Subsequently, the dataset is subjected to testing using the pre-existing model. The testing process involves distinct signature images excluded from the training dataset, ensuring that the neural network is not familiar with them.

3.4. Model evaluation

The evaluation of model accuracy can be conducted by analyzing the result of the confusion matrix, to measure the relationship between image classification results and the model predictive performance. The EER can be used to gain the accuracy of the network. A higher level of accuracy in the biometric system is associated with a lower EER rating. The component of confusion matrix consists elements as:

a. True positive (TP): correctly predicted positive data
b. True negative (TN): correctly predicted negative data
c. False positive (FP): incorrectly predicted negative data
d. False negative (FN): positive data incorrectly predicted

The network performance is evaluated by transforming it into a confusion matrix, enabling the computation of scores:

3.4.1. Accuracy

Accuracy of the model refers to its ability to predict data correctly. It can be defined as the outcome obtained by evaluating the correctness of predictions in relation to the total amount of data. The presented in (1) is used to calculated the accuracy.

$$accuracy = (TP + TN)/(TP + TN + FP + FN) \tag{1}$$

3.4.2. Precision

The ratio of true positive with all number of data instances predicted as positive. Precision shows how often a model is correct in predicting the target class. The precision score can be computed using (2).

$$precision = TP/(TP + FP) \tag{2}$$

3.4.3. Recall or sensitivity

The data presented illustrates the model rate of effectively obtaining information. Recall shows whether or not a model can find all objects in the target class. In (3) provides the mathematical representation for computing the recall value.

$$recall = TP/(TP + FN) \tag{3}$$

3.5. Model implementation

Once the model has been trained and tested and achieve optimal accuracy in predicting signature images, it will be integrated into a mobile application. The trademark image intended for testing, referred to the prediction dataset, is a new image unused for training or testing purposes. Prior to evaluation, this image will go through pre-processing to ensure its compatibility with the input format required by the neural network. Then, it will be used for testing using a pre-trained and saved model. The subject matter pertains to the classification of signatures. The result of the verification process will be presented in textual format, indicating whether the inputted image is a genuine or forged signature.

4. RESULTS AND DISCUSSION

Experiments were conducted using CNN Enhancement classification, following the methods explained in the preceding subsection. Before that, the input dataset will undergo pre-processing and augmentation. The following results are the experimental procedures conducted within the scope of this study. Table 2 shown the usage of the image signature dataset on each stage by 9:10 split ratio of the training and testing stage.

Table 2. Dataset usage

Stage	Number of ID	Genuine signature per ID	Forged signature per ID	Total data used
Training	90	3	3	486
Testing	11	3	3	66
Validation	54	27	27	54

A total of 101 IDs took part in this study. Every person provided three genuine signatures. Then, multiple others replicated the genuine signatures. The comparison of the training, testing, and validation datasets is displayed in Table 2. As a result, the test data consisted of 11 IDs while the training data consists of 90 IDs.

4.1. Image pre-processing and augmentation

The variability in colour and lighting of the collected signature is seen in Figure 2 when using a mobile camera. Variations in pixel size require the implementation of pre-processing techniques. Table 3 shows the pre-processing parameters of the handwritten signature image, used as the input image for signature classification.

Table 3. Image pre-processing parameter

Parameter	Value	Information
Resize	100×100	Adjust the image dimensions to a specified pixel compatible with the convolution layer
Normalization	std = 0.1; mean = 0.5	Modify the image channel
Gaussian blur	True	Reduction of noise present in the image
Augmentation	Rotation (-20, 20) Horizontal flip	Image rotation within a range of -20 degrees to 20 degrees Reverse or flip the image horizontally, resulting in a reflection of the image

The results of the pre-processing and augmentation procedures applied to the input image are illustrated in Figure 3. The signature pattern was successfully differentiated from the background in the provided image. Nevertheless, there are some unproper signature images pre-processing, resulting in complete black recognition. There needs to be more lighting during the capture of the signature image, as well as the requirement for additional pre-processing techniques can contribute to this issue.

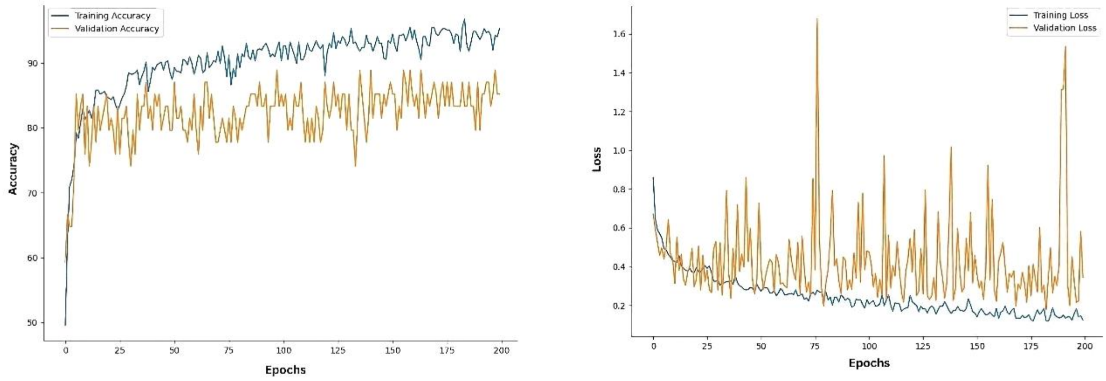
Figure 3. Image pre-processing and augmentation result

The image pre-processing results shown in Figure 3 are some examples resulting from the resize, normalization, Gaussian blur, and augmentation processes. The different image sizes are generalized to a single size to fit the input layer. Next, the normalization process is used to modify the image channels. It speeds up convergence in gradient-based learning algorithms by ensuring that the coefficients are close together within a reasonable range. Gaussian blur is an image enhancement technique used to reduce noise in the image. The augmentation method used image rotation and inversion to multiply the appearance of images with different positions. So that, the network can learn more data variations.

4.2. Training stage and model performance

The training stage was conducted on the CNN enhancement model in this experiment. It was determined by careful evaluation of the characteristics and volume of the input data. All models are exposed to the same learning parameters. The learning parameters used in this experiment are 0.0001 as the learning rate, 200 number of epoch, and Adam as the optimizer. The expected results of this experiment include high training and testing accuracy, without overfitting or underfitting, despite the network's inability to recognize prior data. Thus, it can be concluded that the network has successfully learned to differ genuine with forged signatures. During the first phase of learning, the loss value and model accuracy evaluation is the indicator of the performance of the network.

Figures 4(a) and (b) illustrate the accuracy and loss during the training and validation stages of the CNN Enhancement model, reaching up to 200 th epoch. An epoch refers to utilizing all the training data simultaneously, representing the cumulative count of iterations involving the entire training dataset within a single cycle for the machine learning model's training process. The horizontal axis shows the number of epochs. The vertical axis in Figure 4(a) shows the accuracy value, while the vertical axis in Figure 4(b) is the loss value in the model learning and validation stage.



(a) (b)

Figure 4. Model performance of CNN enhancement; (a) training and validation accuracy graphic and (b) training and validation loss graphic

Table 4 displays the results of the CNN enhancement model training stage. It has a small model size and displays an effective average training time per epoch. It was brought on by the model's comparatively low number of parameters and the application of the batch normalizing procedure. One important factor to take into consideration when integrating the model into mobile applications is its compact size.

Table 4. Model training stage result

Training accuracy	Training loss	Validation accuracy	Validation loss	Testing accuracy	Average training time per epoch	Saved model size
95.47 %	0.135 %	87.04 %	0.310 %	86.36 %	22 milliseconds	2.07 Mega byte

4.3. Model performance evaluation

Confusion matrix provides an illustration of the classification results. There are four parts to the Confusion matrix: TN, FN, TP, and FP. These elements are utilized in the computation of metrics for performance evaluation, including false acceptance rate (FAR), false rejection rate (FRR), and EER. Two axes comprise up the confusion matrix: the horizontal axis indicates the prediction, and the vertical axis indicates the actual. This study is divided into two groups: forged and genuine signatures. The result of the confusion matrix as shown in Figure 5. These values, as shown in (1) through (3), represent the formula for determining the model performance.

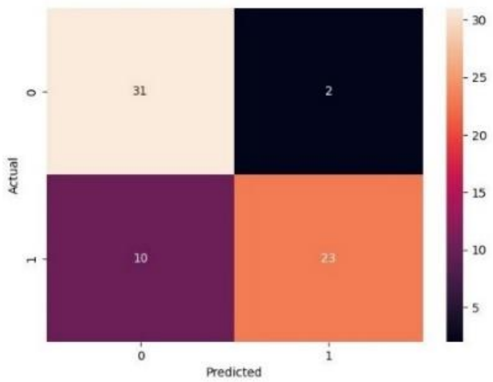


Figure 5. Confusion matrix of CNN enhancement

Figure 5 shows the confusion matrix with two class labels, forged and genuine. Point [0][0] is a true positive, point [0][1] is a false positive, point [1][0] is a false negative, and point [1][1] is a true negative.

The TP, FP, FN, and TN results are 31, 10, 2, and 23, respectively. Table 5 presents a classification report containing F1-score, precision, recall metrics for the model used in the experiment.

Table 5. CNN enhancement model performance

Class	Precision	Recall	F1-score	Accuracy	FAR	FRR	EER
0	0.76	0.94	0.84	0.82	0.061	0.303	0.182
1	0.92	0.70	0.79				

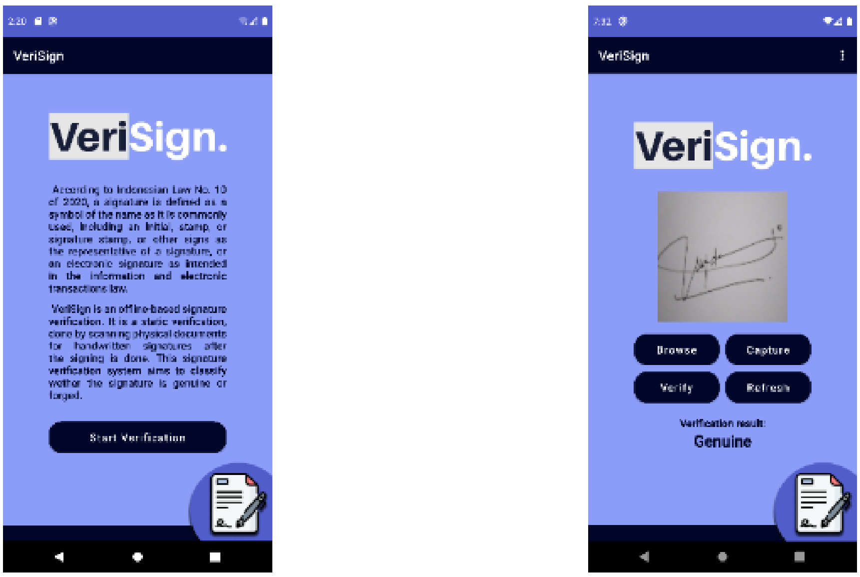
False-positive recognitions divided by the total number of identification attempts is how false-positive recognition rate (FPRR) is calculated within 200 epochs. It is possible to define the FRR as the proportion of valid user identity cases that are wrongly denied. Meanwhile, the EER is derived by calculating the mean value of the FAR and the FRR for each model. According to the information presented in Table 5, CNN enhancement achieved an EER value of 0.182, which is in closer proximity to 0. The network or model performance is considered superior when the values of FAR, FRR, and EER are smaller and closer to zero. The value could be improved by using more complex architecture [14].

4.3. Mobile application development

The model implementation stage comprises two components: the output model, which involves saving the experimental model result and gain the suitable model for the mobile-based application development stage, including integrating and testing the model within the mobile application. The first aim of developing a mobile application in this study is to enhance user mobility and facilitate ease of use. Hence, it is imperative to develop an application that possesses attributes such as low memory usage, prompt responsiveness, high speed, and precise accuracy.

This study examines the current constraints of mobile application development, specifically focusing on embedded mobile applications. Therefore, the size of the model file saved directly impacts the amount of space occupied by the program. The model compact size is a consideration factor when developing mobile applications while integrating the model. The decrease in the size of an embedded model directly correlates with the decreased computational burden imposed on the application. It enhances its usability and mitigates the application crashes. Therefore, in this study, we used CNN Enhancement as our first experiment of mobile application development for offline handwritten genuine signature verification, namely VeriSign.

VeriSign application has been successfully installed on an Android mobile device. The present application uses a total of 88.01 Mega Bytes of internal memory. The memory usage of applications could have a positive correlation with usage and the accumulation of application cache. Figures 6(a) and (b) depict screenshots of the VeriSign mobile application in functioning.



(a) (b)

Figure 6. Running application screenshots; (a) home page and (b) verification page result

Figure 6(a) shows the home page, precisely after the three-second splash screen of the application. This page explains the description and the use of signature. There is a "start verification" button to start the signature verification function. The page displaying the verification results is denoted as Figure 6(b). After clicking the verify button, the model will initiate the process of classifying and verifying the uploaded signature image. The results acquired manifest as a signature class derived from the classification performed by the model. This class might be denoted as "genuine" if the model determines the signature to be genuine or "forged" if the model classifies it as a forged signature. Hence, it is important to implement numerous enhancements to the system and application program code.

5. CONCLUSION

The conclusions drawn in this study were derived from the analysis and experimentation conducted. The influence of the pre-processing strategy on the learning stages and model performance of handwritten signature image datasets is a crucial factor to consider. CNN enhancement model demonstrated performance of 86.36% testing accuracy with 0.061 FAR, 0.303 FRR, and 0.182 EER. The model with the small output model size is deemed more suitable for integration into mobile-based image classification applications. Current mobile application needs to be more capable of accurately identifying genuine signature images and require substantial enhancements. The present study reveals a few gaps that require additional research, particularly in the dataset acquisition, suitable pre-processing techniques, and deep learning architectures employed. It is possible to construct an optimization method for a CNN transfer learning model by using quantization and other approaches aimed at diminishing the storage requirements of the model. In the mobile application development, it is essential to have an in-depth knowledge of the tools employed to integrating CNN model used for classification purposes within the mobile platform. The choice of tools, libraries, and methods may affect the application effectiveness of identification.

ACKNOWLEDGEMENTS

This work was supported by Direktorat Jenderal Pendidikan Vokasi, Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi, Republik Indonesia, through the Master's Thesis grant under contract number 168/SPK/D.D4/PPK.01.APTV/VI/2023.

REFERENCES

[1] A. Foroozandeh, A. Askari Hemmat, and H. Rabbani, "Offline Handwritten Signature Verification and Recognition Based on Deep Transfer Learning," in *2020 International Conference on Machine Vision and Image Processing (MVIP)*, pp. 1-7, 2020, doi: 10.1109/MVIP49855.2020.9187481.

[2] M. Taha, M. M. Selim, and A. Yousry, "A Secured Digital Handwritten Signature Prototype for Visually Impaired People," *International Journal of Intelligent Engineering and Systems*, vol. 13, no. 5, pp. 307–316, 2020, doi: 10.22266/ijies2020.1031.28.

[3] K. Tamilarasi and S. N. Kalyani, "A Survey on Signature Verification Based Algorithms," *2017 IEEE International Conference on Electrical, Instrumentation and Communication Engineering (ICEICE)*, Karur, India, pp. 1–3, 2017, doi: 10.1109/ICEICE.2017.8192438.

[4] M. Diaz, M. A. Ferrer, D. Impedovo, M. I. Malik, G. Pirlo, and R. Plamondon, "A Perspective Analysis of Handwritten Signature Technology," *ACM Computing Surveys*, vol. 51, no. 6, 2019, doi: 10.1145/3274658.

[5] N. M. T. Tahir, A. N. Ausat, U. I. Bature, K. A. Abubakar, and I. Gambo, "Off-line Handwritten Signature Verification System: Artificial Neural Network Approach," *International Journal of Intelligent Systems and Applications (IJISA)*, vol. 13, no. 1, pp. 45–57, 2021, doi: 10.5815/ijisa.2021.01.04.

[6] W. Xiao and Y. Ding, "A Two-Stage Siamese Network Model for Offline Handwritten," *Symmetry*, vol. 14, no. 6, 2022, doi: 10.3390/sym14061216.

[7] S. Pal, M. Blumenstein, and U. Pal, "Hindi Off-line Signature Verification," *2012 International Conference on Frontiers in Handwriting Recognition*, pp. 373–378, 2012, doi: 10.1109/ICFHR.2012.212.

[8] Y. Zhou, J. Zheng, H. Hu, and Y. Wang, "Handwritten Signature Verification Method Based on Improved Combined Features," *Applied* Sciences, vol. 11, no. 13, p. 5867, 2021, doi: 10.3390/app11135867.

[9] G. Suhas, S. Chiranjeevi, S. S. Mokshagundam, and S. Suraj, "SIFR-signature fraud recognition," *2018 International Conference on Networking, Embedded and Wireless Systems (ICNEWS)*, pp. 1–6, 2018, doi: 10.1109/ICNEWS.2018.8903995.

[10] A. B. Jagtap, R. S. Hegadi, and K. C. Santosh, "Feature Learning for Offline Handwritten Signature Verification using Convolutional Neural Network," *International Journal of Technology and Human Interaction*, vol. 15, no. 4, pp. 54–62, 2019, doi: 10.4018/IJTHI.2019100105.

[11] F. Noor, A. E. Mohamed, F. A. S. Ahmed, and S. K. Taha, "Offline Handwritten Signature Recognition using Convolutional Neural Network Approach," 2020 International Conference on Computing, Networking, Telecommunications & Engineering Sciences Applications (CoNTESA), pp. 51–57, 2020, doi: 10.1109/CoNTESA50436.2020.9302868.

[12] T. M. Ghanim and A. M. Nabil, "Offline Signature Verification and Forgery Detection Approach," *2018 13th International Conference on Computer Engineering and Systems (ICCES)*, Cairo, Egypt, pp. 293-298, 2018, doi: 10.1109/ICCES.2018.8639420.

[13] O. El Melhaoui and S. Benchaou, "An Efficient Signature Recognition System Based on Gradient Features and Neural Network Classifier," *Procedia Computer Science*, vol. 198, pp. 385–390, 2022, doi: 10.1016/j.procs.2021.12.258.

[14] V. Ruiz, I. Linares, A. Sanchez, and J. F. Velez, "Off-line Handwritten Signature Verification using Compositional Synthetic Generation of Signatures and Siamese Neural Networks," *Neurocomputing*, vol. 374, pp. 30–41, 2020, doi: 10.1016/j.neucom.2019.09.041.

[15] M. Ajij, S. Pratihar, S. R. Nayak, T. Hanne, and D. S. Roy, "Off-line Signature Verification using Elementary Combinations of Directional Codes from Boundary Pixels," *Neural Computing and Applications*, vol. 35, no. 7, pp. 4939–4956, 2023, doi: 10.1007/s00521-021-05854-6.
[16] K. Bibi, S. Naz, and A. Rehman, "Biometric Signature Authentication using Machine Learning Techniques: Current Trends, Challenges and Opportunities," *Multimedia Tools and Applications*, vol. 79, no. 1–2, pp. 289–340, 2020, doi: 10.1007/s11042-019-08022-0.
[17] S. Dey, A. Dutta, J. I. Toledo, S. K. Ghosh, J. Llados, and U. Pal, "SigNet: Convolutional Siamese Network for Writer Independent Offline Signature Verification," *Pattern Recognition Letters*, no. 1, pp. 1–7, 2017, doi: 10.48550/arXiv.1707.02131.
[18] P. Ratna, S. Poudel, M. Baduwal, S. Burlakoti, and S. P. Panday, "Signature Verification using Convolutional Neural Network and Autoencoder," *Journal of the Institute of Engineering*, vol. 16, no. 1, pp. 33–40, 2021, doi: 10.3126/jie.v16i1.36533.
[19] 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 Computer Science*, vol. 161, pp. 475–483, 2019, doi: 10.1016/j.procs.2019.11.147.
[20] F. E. Batool *et al.*, "Offline signature verification system: a novel technique of fusion of GLCM and geometric features using SVM," *Multimedia Tools and Applications*, vol. 83, pp. 14959-14978, 2020, doi: 10.1007/s11042-020-08851-4.
[21] S. Pal, A. Alaei, U. Pal, and M. Blumenstein, "Performance of an Off-Line Signature Verification Method Based on Texture Features on a Large Indic-Script Signature Dataset," *2016 12th IAPR Workshop on Document Analysis Systems (DAS)*, pp. 72–77, 2016, doi: 10.1109/DAS.2016.48.
[22] J. O. Pinzón-Arenas, R. Jiménez-Moreno, and C. G. Pachón-Suescún, "Offline signature verification using DAG-CNN," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 4, pp. 3314–3322, 2019, doi: 10.11591/ijece.v9i4.pp3314-3322.
[23] Md. T. F. Rabbi, S. T. Rahman, P. Biswash, J. Kim, and A. Sheikh, "Handwritten Signature Verification Using CNN with Data Augmentation," *The Journal of Contents Computing*, vol. 1, no. 1, pp. 25–37, 2019, doi: 10.9728/jcc.2019.12.1.1.25.
[24] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *32nd* International *Conference on Machine Learning, ICML 2015*, vol. 1, pp. 448–456, 2015, doi: 10.48550/arXiv.1502.03167
[25] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.
[26] C. Garbin, X. Zhu, and O. Marques, "Dropout vs. Batch Normalization: an Empirical Study of Their Impact to Deep Learning," *Multimedia Tools and Applications*, vol. 79, no. 19–20, pp. 12777–12815, 2020, doi: 10.1007/s11042-019-08453-9.
[27] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *Journal of Big Data*, vol. 6, no. 60, 2019, doi: 10.1186/s40537-019-0197-0.

BIOGRAPHIES OF AUTHORS

Wifda Muna Fatihia in 2021 and 2023, she received her Bachelor's and Master's degrees in Informatics Engineering from the Department of Informatics and Computer Engineering at the Politeknik Elektronika Negeri Surabaya, Indonesia. Her current areas of research interest are computer vision and mobile application development. She can be contacted at email: wifdamu@gmail.com.

Arna Fariza in 1996 and 2003, she received her Bachelor's and Master's degrees in Computer Science from the Department of Computer Science at the Institut Teknologi Sepuluh Nopember, Indonesia. In 2020, she graduated with a Doctorate from the Electrical Engineering Department at Institut Teknologi Sepuluh Nopember, Indonesia. She participates in the EEPIS Computer Vision Research Group. Her current areas of research interest are computer vision, image processing, and geographic information system. She can be contacted at email: arna@pens.ac.id.

Tita Karlita in 2002 and 2006, she received her Bachelor's and Master's degrees in Computer Science from the Department of Computer Science at the Institut Teknologi Sepuluh Nopember, Indonesia. Additionally, in 2020, she graduated with a Doctorate from the Electrical Engineering Department at Institut Teknologi Sepuluh Nopember, Indonesia. She participates in the EEPIS Knowledge Engineering and Health Engineering Research Group. Her current areas of research interest are computer vision, image processing, and 3D reconstruction. She can be contacted at email: tita@pens.ac.id.