SINERGI Vol. 29, No. 1, February 2025: 241-250
http://publikasi.mercubuana.ac.id/index.php/sinergi
http://doi.org/10.22441/sinergi.2025.1.022

Indonesia rupiah currency detection for visually impaired people using transfer learning VGG-19

Raissa Alfatikarani[1], Laras Suciningtyas [1], Genta Garuda Bimasakti [1], Faqisna Putra[1], Jessie R. Paragas[2], Hapsari Peni Agustin Tjahyaningtijas[3,*]
[1]Bachelor of Electrical Engineering Study Program, Faculty of Engineering, Universitas Negeri Surabaya, Indonesia
[2]Department of Information Technology, Eastern Visayas State University, Philippines
[3]Department of Electrical Engineering, UCE DIC, Universitas Negeri Surabaya, Indonesia

Abstract
People with visual impairments often face difficulties in determining the authenticity of paper money, which is a crucial skill to avoid fraud. The limitations of traditional methods, like blind codes for visually impaired people, require a more advanced and efficient solution. Previous methods of currency detection using Convolutional Neural Network (CNN) techniques, including the VGG-19 architecture, have often encountered challenges, particularly the long training times required. Therefore, we propose using transfer learning techniques and modifying the top layers of the VGG-19 model, known as fully connected layers, within a mobile application with audio feedback built using Android Studio. These modifications involve substituting the three fully connected layers with dense and flattened layers. We also implemented hyperparameter tuning, including adjusting the batch sizes and setting the number of epochs. The datasets used Indonesian Rupiah paper currency from the 2022 emission year, specifically Rp 50,000 and Rp 100,000 denominations. The best transfer learning VGG-19 model achieved a batch size of 32 and an epoch of 50, resulting in a high accuracy of 88%. Response speed testing with performance profiling on Android Studio showed an overall average response time of 458 ms. The main advantage of using transfer learning with the VGG-19 model is that it significantly reduces training time while still achieving high accuracy, differentiating this work from previous studies that relied on training from scratch, which is more time-consuming and resource-intensive. Therefore, this mobile app can be categorized as having a fast response time.

Keywords:
Application;
Blind;
Currency Detection;
Modified VGG-19;

Article History:
Received: June 18, 2024
Revised: August 14, 2024
Accepted: September,9 2024
Published: January 5, 2025

Corresponding Author:
Hapsari Peni Agustin Tjahyaningtijas
Electrical Engineering Department, UCE DIC, Universitas Negeri Surabaya, Indonesia
Email:
hapsaripeni@unesa.ac.id

INTRODUCTION

Money is essential in daily life, such as buying, selling, and transactions for people with visual impairments [1]. Visually impaired people run the risk of becoming victims of fraud or making mistakes while giving or receiving money because they are unable to identify the value of currency with their eyes [2]. According to the Payment System Statistics and Financial Market Infrastructure (SPIP) report by Bank Indonesia, the amount of money in circulation suspected to be counterfeit or of questionable authenticity rose by 24% between 2023 and early 2024. This increase corresponds to approximately IDR 192,647 banknotes. Additionally, according to information from the Republic of Indonesia's Ministry of Health (Kemenkes RI), 1.5% of the country's population is visually impaired. Given

that there are 250 million people living in Indonesia today, at least 4 million of them suffer from visual impairments, including blindness and low vision [3].

Visually impaired individuals encounter several challenges while identifying currency, mainly because banknotes often have similar designs and sizes without enough distinct features that can be easily discerned through touch [4]. Several series of Indonesian Rupiah banknotes have been equipped with blind codes, which are unique markers to facilitate identification by visually impaired users [5]. The blind code applied to the 2022 currency notes shares similarities with the one found on the 2016 currency notes, as both consist of a series of lines embedded on each denomination of money. Despite the availability of Blind codes, visually impaired individuals still encounter difficulties distinguishing the denominations of paper currency using this method. The use of Blind code on paper currency may not be effective for individuals with visual impairments for several reasons. Firstly, the raised lines can experience damage over time, making them difficult to identify. Secondly, for individuals with visual impairments who are not accustomed to or have reduced touch sensitivity, distinguishing the line patterns might provide some difficulties. Thirdly, variations in the design of banknotes across issuances might lead to confusion.

Therefore, the existence of weaknesses in traditional methods, such as blind codes for the visually impaired, requires a more sophisticated and efficient solution. Computer-based technology can provide a solution by performing image processing on Rupiah currency images to facilitate the detection of information on the denomination and authenticity of the currency. Machine learning is a branch of computer science that allows systems to learn from data and make decisions without explicit programming [6]. Machine learning relies on developing algorithms that can analyze and learn from data so that the system may identify a train, create a prediction, or provide a response with a high level of accuracy by analyzing and interpreting the data [7]. Several machine learning algorithms have been developed and refined to enhance the efficiency and accuracy of detection processes, including Logistic Regression, Linear Discriminant (LDA), K-Nearest Neighbor (*K*-NN) Decision Tree Classifier (DTC), Random Forest Classifier (RFC), Support Vector Machine (SVM), Artificial Neural Network [8][9]. Additionally, machine learning, such as logistic regression or decision trees, features must be manually extracted, which may not be efficient or effective in complex or image-based tasks [10]. Deep learning enables neural networks to automatically identify complex patterns that might cause difficulties in human perception, thus enhancing performance in tasks, including classification and recognition [11].

Deep learning, recognized for its advancements in computer vision, provides a unified approach to various tasks using algorithms that mimic human behavior. Unlike traditional machine learning, which involves multiple steps, deep learning analyzes patterns in large datasets across fields like image and text analysis, language processing, financial forecasting, and medical applications [12][13]. The Convolutional Neural Network (CNN) method, a significant advancement, uses convolutional layers to effectively process spatial patterns in images, making it suitable for detecting unique features of banknotes [14].

CNNs are commonly used for banknote image processing, though traditional training methods can be time-consuming and prone to overfitting [15][16]. Laavanya Mohan and Vijayaraghvan[17] utilized Transfer Learning (TL) with AlexNet to identify counterfeit money, achieving 75% accuracy with a model trained on Indian banknote images. However, this method may not generalize well to currencies other than Indian banknotes, and the accuracy is relatively low for practical applications. While Kiran Kamble *et al.* [18] achieved 85.6% accuracy using a custom CNN with a dataset of real and fake banknotes, their CNN model required significant manual intervention, and their approach might overfit the dataset used, limiting its generalization to new data. Siddiki *et al.*[19], developed a system for detecting Bangladeshi banknotes with 86% accuracy using CNN and FLANN-based Matcher with SIFT, available on web and Android platforms. This approach's reliance on SIFT features can limit its adaptability to different currency types or environmental conditions. Saini *et al.* [20], relied on traditional feature extraction methods, which may not fully capture the nuances needed for accurate currency detection. This limits their model's robustness, especially in handling diverse and complex currency images. Channum Park *et al*. [21], developed a three-stage detection system, which may not be sufficient for robust and scalable applications, using Faster R-CNN, geometric constraints, and ResNet for South Korean banknotes. The performance of these models can degrade when applied to new or unseen currencies, indicating a need for more advanced feature extraction techniques. Gurjot Singh and Jasjot Singh [22] employed advanced

supervised image processing and the F-KNN method to analyze currency, achieving a precision of 68.3% using confusion matrices and ROC curves.

Therefore, in this research, we propose using the CNN approach with the VGG-19 architecture, applying transfer learning techniques, and creating modifications to the top layers of the VGG-19 model, namely the fully connected layers. These modifications involve substituting the three fully connected layers with dense and flattened layers. The modified VGG-19 architecture comprises 16 Convolutional Layers utilized for feature extraction, followed by max-pooling layers grouped into five blocks, with an additional two layers (Flatten and Dense) employed for the classification stage. This study also conducted hyperparameter tuning by adjusting the batch size and number of epochs to achieve better accuracy performance. The batch size impacts how quickly a model learns and how stable the learning process is [23]. Our approach includes developing a mobile application that performs real-time currency detection and provides audio feedback, making it accessible to visually impaired users.

The VGG-19 transfer learning model, achieving the highest accuracy, will be implemented in a mobile app using JavaScript and Android Studio. An application is software designed for specific tasks on specialized systems [24], focusing on problem-solving through various data processing techniques [25]. Java, known for its flexibility and cross-platform capability, adheres to the "write once, run anywhere" principle [26]. Android Studio aids app development with features like emulators for device previews [27][28]. For real-time object detection and audio feedback for visually impaired users, this research utilizes the gTTS (Google Text To Speech) API to convert analysis results into audio. Bhuyan et al. (2019) demonstrated that TTS technology effectively supports text detection and audio feedback, aiding in currency authenticity verification for the visually impaired [29].

This study tackles key issues in current currency detection methods, particularly those using Convolutional Neural Networks (CNN) like the VGG-19 model. Existing methods often face challenges such as long training times and poor real-time performance. We introduce a new approach using transfer learning specifically for Rupiah currency detection to address these issues. We modify the VGG-19 model by replacing its three fully connected layers with dense and flattened layers, enhancing the model's accuracy on small datasets and speeding up the training process, resulting in better performance for real-time applications compared to existing methods. Additionally, our study presents a novel use of text-to-speech technology (gTTS API) to provide immediate audio feedback, offering a more effective solution than traditional methods like blind codes for visually impaired users. By combining these new techniques, our research offers a fresh contribution to the field, overcoming the limitations of previous methods and making currency recognition technology more accessible and practical for visually impaired individuals.

METHOD

Dataset

The dataset used consists of the Indonesian Rupiah currency released in 2022. Meanwhile, the counterfeit currency dataset consists of toy money that was issued in the same year, 2022. The process of collecting data is shown in Figure 1 and Figure 2.

The dataset consists of 50,000 Real and 50,000 Fake, 100,000 Real and 100,000 Fake. The number of image data for each denomination is 200 for real currency and 200 for counterfeit currency. Therefore, a dataset consisting of 4 classes has 800 image data experiment. The dataset will be used as input data for the preprocessing step, utilizing Image Augmentation by performing flipping and rotating to enhance data diversity and the classification stage.

Methods

This study aims to detect currency authenticity using the deep learning method Convolutional Neural Network (CNN) with VGG-19 architecture and Transfer Learning. Figure 3 presents the flowchart of the Transfer Learning VGG-19 method used in the experimental design.

Figure 1. Real Currency

Figure 2. Fake Currency

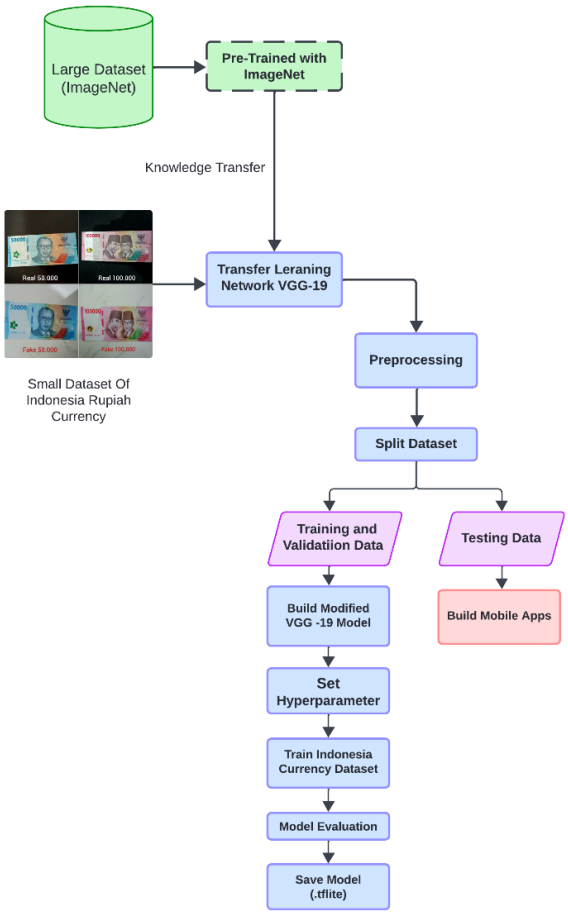

Figure 3. Flowchart Of The Transfer Learning VGG-19 Method

Pre-Trained Model with ImageNet

The pretraining process on the ImageNet dataset is performed by utilizing a transfer learning model architecture (VGG-19) and training it on the ImageNet dataset. During the pretraining phase, the model learns useful feature representations from the images in the ImageNet dataset, such as edges, textures, and more complex features that enable the model to distinguish between different types of objects accurately. Algorithm 1 is the pseudocode for a pre-trained VGG-19 model.

Algorithm 1. Pseudocode of Transfer Learning VGG-19

Input: Image Dataset
Output: Prediction layer, Trained Model

1. Input image data of (224, 224, 3) 'input_shape=(224, 224) + tuple([3]) '
2. Initialize ImageNet pre-trained model
3. Exclude the fully-connected layers (top layers) of the VGG-19 model 'include_top=False'
4. Set the pre-trained layers of VGG-19 to be non-trainable to prevent re-training 'layer.trainable = False'
5. Set the 'Flatten' layer to convert the multi-dimensional output of VGG-19 into a one-dimensional vector
6. Set custom output layer 'Dense' for the specific task with 'softmax' activation function
7. Connect the output of the Flatten layer to the Dense layer
8. Create the final Modified VGG-19 model

Preprocessing

During preprocessing, image augmentation is applied by flipping images to increase dataset size and prevent overfitting [30]. Images are also resized to 224 x 224 pixels to standardize input resolution, speeding up training. This study applied horizontal flipping by performing a transformation that mirrors the image along the vertical axis for preprocessing the dataset.

$$I_{aug}(x', y') = T_{flipping}(I(x, y)) \quad (1)$$

$$T_{flipping} = \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} W - x - 1 \\ y \end{pmatrix} \quad (2)$$

Explanation:
$I(x, y)$: The original image with pixel coordinates (x,y)
$T_{flipping}$: The transformation function that mirrors the image horizontally
$I_{aug}(x', y')$: The augmented image after flipping, with new pixel coordinates (x',y')
W : The Width of the original image I(x,y)

Split Dataset

Dividing the dataset into training, validation, and testing. After augmentation, the number of training, validation, and testing data for four classes consists of 960 training data, 320 validation data, and 320 testing data. Algorithm 2 shows the pseudocode for a splitting data process.

Algorithm 2. Pseudocode of Splitting Dataset

Input: Image Dataset
Output: Data training, Validasi, Testing

1. Define Split Data 'def copy_data(source_dir, training_dir, validation_dir, testing_dir, split_size) '
2. Shuffle data splits 'random.shuffle(files) '
3. Calculate the number of files for each split 'total_files = len(files) '
4. Calculate the number of training files 'training_length = int(total_files * split_size) '
5. Calculate the number of validation files which is half of the remaining files after training 'validation_length = int((total_files - training_length) / 2) '
6. Calculate the number files for testing 'testing_length = total_files - training_length - validation_length'

Build Modified VGG-19 Model

The VGG-19 model will be adapted for transfer learning by modifying its convolutional, max pooling, flattening, and dense layers. The

Flatten layer converts the output from convolutional layers into a one-dimensional vector for classification by the Dense layers [31]. His adaptation omits the usual fully connected layers in VGG-19. The Dense layer functions as the output layer for multi-class classification, with neurons corresponding to the number of classes in the dataset. The VGG-19 model with this transfer learning technique is illustrated in Figure 4.

Set Hyperparameter and Training

At this stage, the hyperparameter tuning is performed, including adjusting the Batch Size to 32, 64, and 128 and setting the number of epochs to 20,30,40 and 50.

Creating Mobile Apps

The system was developed using Android Studio. The initial Set-Up Environment phase involves installing Android Studio, the Android SDK, and other components [32]. Android Virtual Devices (AVDs) are created for testing on various device configurations [33]. In the front-end development, the resources (res) folder includes subfolders like drawable, layout, mipmap, raw, values, and XML, with XML files defining the app's UI layout [34]. The best model is exported with MP3 audio files for the app. Back-end development uses Java to manage UI and application logic interactions [35].

The debugging & Testing phase resolves code errors [36], featuring currency detection and audio feedback, and is prepared for deployment and publication [37]. The steps are shown in Figure 5.

RESULTS AND DISCUSSION

Runze Lin's study indicates optimal CNN accuracy with a batch size of 16 to 32 [38], and Kandel and Casteli (2020) support smaller batch sizes for better data training [39]. Raniah and Humera [40] found the Adam optimizer superior to RMSprop increasing epochs generally reduces training loss [41]. This study modified the VGG-19 architecture by replacing three fully connected layers with a flattened and dense layer, adjusting batch sizes (32, 64, 128) and epochs (20, 30, 40, 50) using the SoftMax function and Adam optimizer on a 4-class dataset.

Table 1 shows that the best model, with a batch size of 32 and 50 epochs, achieved 88% accuracy, highlighting that smaller batch sizes and more epochs improve VGG-19 accuracy. The VGG-19 architecture, with its deep convolutional layers, captures complex image features. Replacing fully connected layers with dense and flattened layers helps the model focus on relevant features, reducing overfitting and improving generalization. This adjustment enhances accuracy for different currency images, with dense layers crucial for combining features and focusing on significant details.

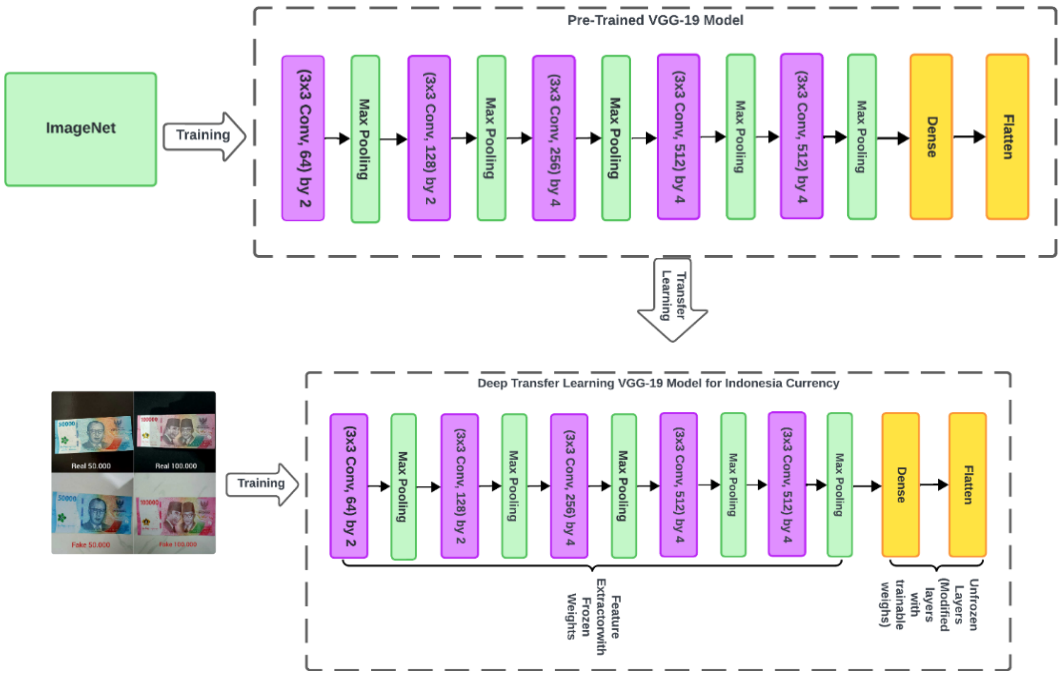

Figure 4. VGG-19 Architecture with Transfer Learning Model for Indonesia Currency Detection

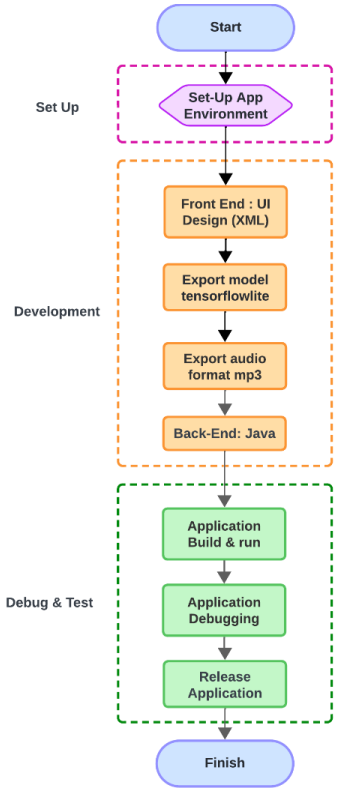

Figure 5. Steps For Creating Mobile Apps

Table 1. Comparison Performances Model

Batch Size	Epoch	Accuracy %	Precision %	Recall %	FI Score %
32	20	85.9	86	86	86
	30	86.5	87	87	87
	40	87.8	88	88	88
	50	88	88	88	88
64	20	85	85	84	84
	30	85.6	86	86	86
	40	86.8	87	87	87
	50	87.8	88	88	88
128	20	82.8	83	83	83
	30	83.5	84	84	84
	40	85	83	83	83
	50	85.62	88	88	88

The flattened layer converts 2D data into a 1D vector, essential for classification by dense layers while preserving spatial features. Pre-trained VGG-19 uses features from a large dataset, reducing extensive training, which is beneficial for limited data and real-time use. Smaller batch sizes, like 32, allow frequent weight updates, enhancing learning but may introduce noise. More epochs, such as 50, help adjust weights for better accuracy, though excessive epochs can cause overfitting. A confusion matrix evaluates a classification model's performance on test data [42], as shown for the modified VGG-19 model in Figure 6.

Figure 7 and Figure 8 plot validation accuracy and loss over 50 epochs. Initially, both training and validation losses decrease sharply, indicating effective early learning.

The unsuccessful reading of certain rupiah denominations is due to factors observed during testing. While the VGG-19 model effectively captures complex features, it sometimes misclassifies similar denominations due to limitations in the fully connected layers. Hyperparameters, such as batch size and number of epochs, also influence performance, with smaller batch sizes potentially introducing noise. Despite extensive training, the model shows a 35% validation loss, indicating difficulty in generalizing to new data. To address this, refined data augmentation and ensemble methods combining multiple models are recommended to reduce misclassification.

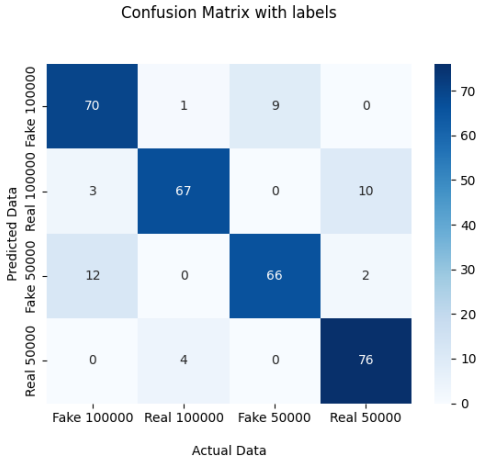

Figure 6. Confusion Matrix Of 4 Class For Money Detection

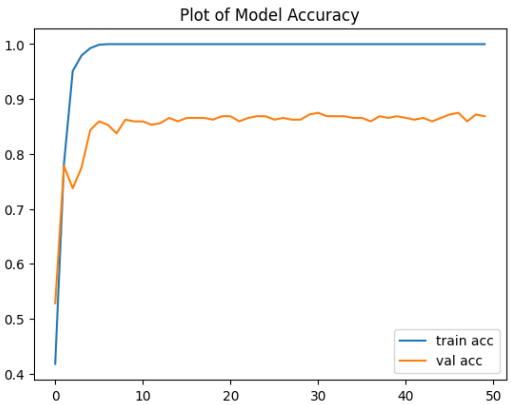

Figure 7. Transfer Learning VGG-19 Validation Accuracy

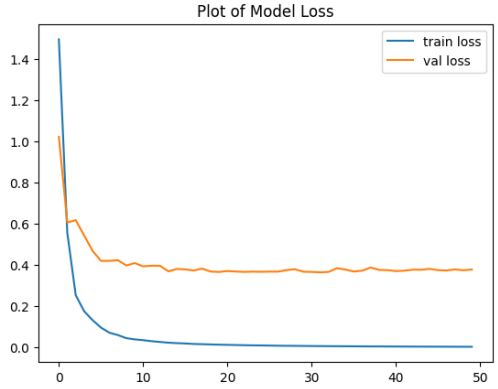

Figure 8. Transfer Learning VGG-19 Validation Loss

The comparative results of the previous methods are presented in Table 2. The proposed VGG-19 approach outperforms other methods like AlexNet, as it uses smaller filters (3x3) that better capture details compared to AlexNet's larger filters (11x11) [17][43][44]. Unlike Laavanya's webcam-based method [17], which is less effective for visually impaired users, this research integrates audio feedback. It also surpasses the methods of Kiran Kamble et al. [18] and Chanum Park et al. [21] used VGG-16, as VGG-19 has more convolutional layers, extracting more detailed features. Additionally, it improves upon Siddiki et al.'s work [19] by using VGG-19 to extract image automatic features more effectively than traditional algorithms like SIFT, which are slower and less efficient with scale and rotation changes [45][46].

Table 2. Comparison Results of Previous Method

Authors	Year	Dataset	Method	Accuracy (%)
Laavanya [17]	2019	Indian currency (50,200,500,2000 rupee)	Transfer Learning Alex-Net	75
Kiran Kamble, et al [18]	2019	Indian currency (50 ,2000 rupee)	CNN (Transfer Learning VGG-16)	85.6
M.A Siddiki, et al [19]	2023	Bangladesh currency (200 Tk)	CNN dan FLANN Based matcher	86
Saini, et al [20]	2022	India currency (10 & 2000 rupee)	Inception V3	69.7
Chanum Park, et al. [21]	2020	South Korea currency (10, 50, 100, 500, 1000, 5000, 10000, 50000 KRW	Faster R- CNN (VGG-16)	85
Gurjot Singh [22]	2021	Indian Currency	Fine K-Nearest Neighbour (f-KNN)	68.7
Proposed Model	**2024**	**Indonesia Currency (IDR 50,000 and IDR 100,000)**	**CNN (Transfer Learning VGG-19)**	**88**

The method also outperforms Saini et al.'s InceptionV3 [20], which has a larger model size, a more complicated structure, and stops learning at lower accuracy levels. Finally, VGG-19 is more effective than Gurjot Singh's F-KNN approach [22], which suffers from limited feature extraction and dataset issues. Overall, this study shows that the VGG-19 method with transfer learning is highly suitable for mobile applications in real-time counterfeit detection, especially with the added benefit of audio feedback for visually impaired users.

Implementation of Mobile Applications

The mobile application processes the input image and provides detection results with audio feedback. The audio will say "This money has a real tendency of 50,000 Rupiah" if the money is real or "This money has a fake tendency of 50,000 Rupiah" if it is fake. Android Studio's performance profiling toolset, shown in Figure 9, measures the app's responsiveness. The application's response time ranges from 459 ms to 592 ms, with an average of 458 ms. Figure 10 illustrates the mobile application interface, showing the detection of a Rp 50,000 Indonesian banknote.

Algorithm 3. Pseudocode Of Implementation on Mobile Apps

1. Import all required packages
2. Define the MainActivity class extending AppCompatActivity
   public class
   'MainActivity extends AppCompatActivity '
3. Define variables
   TextView result, confidence; ImageView imageView;
   Button picture; int imageSize = 224;
   ActivityResultLauncher<Intent> cameraLauncher;
   String[] classes = { " Real 50000 ", "Fake 50000",
   "Real 100000", "Fake 100000"};
4. Initialize ActivityResultLauncher to launch camera
5. Initialize camera permission
6. Classify image
   TensorBuffer.createFixedSize(new int[]{1, 224, 224, 3}, DataType.FLOAT32)
7. Initialize to get audio resource based on class name
   int getAudioResource(String className)

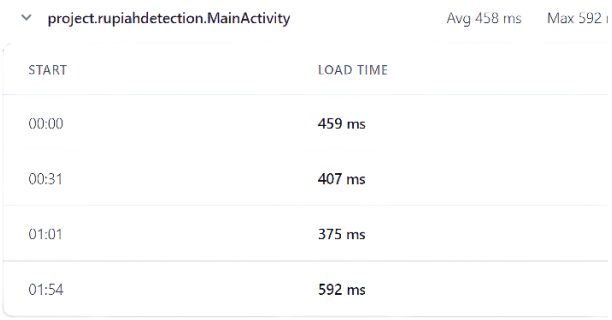

Figure 9. Profiling Performance of Mobile Apps

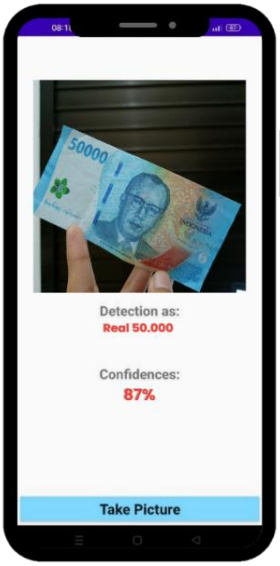

Figure 10. Mobile Application View

CONCLUSION

Implementing the CNN method with the VGG-19 transfer learning model uses convolutional, max pooling, dense layers, and hyperparameter tuning of flattens, epoch, and batch size to achieve high accuracy. After 50 epochs with a batch size of 32, the model reached an accuracy of 88% for classifying four classes of Rupiah currency. Performance profiling shows the mobile app's response time for detecting Rupiah authenticity ranges from 459 ms to 592 ms, with an average of 458 ms, categorizing it as fast.

A limitation of this study is the dataset, which may not fully capture the diversity of counterfeit currency cases in society. Future research should focus on creating a more diverse dataset, including counterfeit notes currently circulating, to enhance the model's robustness and real-world applicability.

ACKNOWLEDGMENT

This research is supported and funded by the Research and Community Service Institution of Surabaya State University (LPPM UNESA) – Non-APBN Research Fund 2024, contract number: B/23301/UN38.III.1/LT.00/2024.

REFERENCES

[1] J. Huebner, E. Fleisch, and A. Ilic, “Assisting mental accounting using smartphones: Increasing the salience of credit card transactions helps consumer reduce their spending,” *Comput. Human Behav.*, vol. 113, no. September 2019, p. 106504, 2020, doi: 10.1016/j.chb.2020.106504.

[2] S. Singh, N. Jatana, S. Sehgal, R. Anand, B. Arunkumar, and J. V. N. Ramesh, “Making Digital Payments Accessible Beyond Sight: A Usability Study of UPI-Based Smartphone Applications,” *IEEE Access*, vol. 12, no. December 2023, pp. 6830–6841, 2024, doi: 10.1109/ACCESS.2023.3348840.

[3] Siswoyo, N. Widayati, and N. Habibah, “An Overview Of The Knowledge Of People With Diabetes Mellitus About Diabetic Retinopathy: A Literature Review,” *J. Rural Community Nurs. Pract.*, vol. 1, no. 1, pp. 52–67, 2023, doi: 10.58545/jrcnp.v1i1.84.

[4] S. C. Ng, C. P. Kwok, S. H. Chung, Y. Y. Leung, and H. S. Pang, “An Intelligent Banknote Recognition System by using Machine Learning with Assistive Technology for Visually Impaired People,” *10th Int. Conf. Inf. Sci. Technol. ICIST 2020*, pp. 185–193, 2020, doi: 10.1109/ICIST49303.2020.9202087.

[5] B. Mounira, “Blockchain Technology Applications in the Islamic Financial Industry-The Smart Sukuk of Blossom Finance’s Platform in Indonesia Model,” *Econ. Sci. Manag. Commer. Sci. Rev.*, vol. 02, pp. 309–325, 2020.

[6] I. H. Sarker, “Machine Learning: Algorithms, Real-World Applications and Research Directions,” *SN Comput. Sci.*, vol. 2, no. 3, pp. 1–21, 2021, doi: 10.1007/s42979-021-00592-x.

[7] B. Mahesh, “Machine Learning Algorithms - A Review,” *Int. J. Sci. Res.*, vol. 9, no. 1, pp. 381–386, 2020, doi: 10.21275/ART20203995.

[8] A. D. Rochendi, L. M. Silalahi, and I. U. V. Simanjuntak, “Touchless palm print recognition system design using Gray Level Co-occurrence Matrix feature with K-Nearest Neighbor classification in MATLAB,” *Sinergi (Indonesia)*, vol. 28, no. 2, pp. 337–346, 2024, doi: 10.22441/sinergi.2024.2.013.

[9] I. Izzati, I. K. Sriwana, and S. Martini, “Drug forecasting and supply model design using Artificial Neural Network (ANN) and

Continuous Review (r, q) to minimize total supply cost," *Sinergi (Indonesia)*, vol. 28, no. 2, pp. 219–230, 2024, doi: 10.22441/sinergi.2024.2.002.

[10] S. Hakak, M. Alazab, S. Khan, T. R. Gadekallu, P. K. R. Maddikunta, and W. Z. Khan, "An ensemble machine learning approach through effective feature extraction to classify fake news," *Futur. Gener. Comput. Syst.*, vol. 117, pp. 47–58, 2021, doi: 10.1016/j.future.2020.11.022.

[11] J. Huixian, "The Analysis of Plants Image Recognition Based on Deep Learning and Artificial Neural Network," *IEEE Access*, vol. 8, pp. 68828–68841, 2020, doi: 10.1109/ACCESS.2020.2986946.

[12] D. A. Bashar, "Survey on Evolving Deep Learning Neural Network Architectures," *J. Artif. Intell. Capsul. Networks*, vol. 2019, no. 2, pp. 73–82, 2019, doi: 10.36548/jaicn.2019.2.003.

[13] S. Sengupta *et al.*, "A review of deep learning with special emphasis on architectures, applications and recent trends," *Knowledge-Based Syst.*, vol. 194, p. 105596, 2020, doi: 10.1016/j.knosys.2020.105596.

[14] A. S. Alene and M. Meshesha, "Ethiopian Paper Currency Recognition System:"An Optimal Feature Extraction," *Ieee-Sem*, vol. 7, no. 8, pp. 2320–9151, 2019.

[15] M. A. Hossain and M. S. Alam Sajib, "Classification of Image using Convolutional Neural Network (CNN)," *Glob. J. Comput. Sci. Technol.*, vol. 19, no. May, pp. 13–18, 2019, doi: 10.34257/gjcstdvol19is2pg13.

[16] S. K. Noon, M. Amjad, M. A. Qureshi, and A. Mannan, "Overfitting Mitigation Analysis in Deep Learning Models for Plant Leaf Disease Recognition," *Proc. - 2020 23rd IEEE Int. Multi-Topic Conf. INMIC 2020*, 2020, doi: 10.1109/INMIC50486.2020.9318044.

[17] M. Laavanya and V. Vijayaraghavan, "Real Time Fake Currency Note Detection using Deep Learning," *Int. J. Eng. Adv. Technol.*, vol. 9, no. 1S5, pp. 95–98, 2019, doi: 10.35940/ijeat.a1007.1291s52019.

[18] K. Kamble, A. Bhansali, P. Satalgaonkar, and S. Alagundgi, "Counterfeit Currency Detection using Deep Convolutional Neural Network," *2019 IEEE Pune Sect. Int. Conf. PuneCon 2019*, pp. 31–34, 2019, doi: 10.1109/PuneCon46936.2019.9105683.

[19] M. A. Siddiki, M. N. Hossain, K. Akhter, and M. R. Rahman, "Bangladeshi Currency Identification and Fraudulence Detection Using Deep Learning and Feature Extraction," *Int. J. Comput. Sci. Mob. Comput.*, vol. 12, no. 1, pp. 1–13, 2023, doi: 10.47760/ijcsmc.2022.v12i01.001.

[20] P. Saini, A. Goyal, B. Pokhrel, J. Singh, and N. Kumar, "Indian Currency Recognition:A Lightweight Transfer Learning Approach for Visually Impaired," *PDGC 2022 - 2022 7th Int. Conf. Parallel, Distrib. Grid Comput.*, pp. 248–252, 2022, doi: 10.1109/PDGC56933.2022.10053304.

[21] C. Park, S. W. Cho, N. R. Baek, J. Choi, and K. R. Park, "Deep feature-based three-stage detection of banknotes and coins for assisting visually impaired people," *IEEE Access*, vol. 8, pp. 184598–184613, 2020, doi: 10.1109/ACCESS.2020.3029526.

[22] G. Singh Sodhi and J. Singh Sodhi, "A Robust Invariant Image-Based Paper-Currency Recognition Based on F-kNN," *Int. Conf. Intell. Technol. Syst. Serv. Internet Everything, ITSS-IoE 2021*, pp. 1–6, 2021, doi: 10.1109/ITSS-IoE53029.2021.9615287.

[23] Y. Du, W. Shen, B. Liu, W. Lu, and H. Gong, "Dual Batch Size Training: An efficient MGD adaptive batch size method," *Proc. - Int. Conf. Tools with Artif. Intell. ICTAI*, vol. 2021-Novem, no. 1, pp. 873–879, 2021, doi: 10.1109/ICTAI52525.2021.00140.

[24] H. Abdullah and S. R. M. Zeebaree, "Android Mobile Applications Vulnerabilities and Prevention Methods: A Review," *Proc. 2021 2nd Inf. Technol. to Enhanc. E-Learning other Appl. Conf. IT-ELA 2021*, pp. 148–153, 2021, doi: 10.1109/IT-ELA52201.2021.9773615.

[25] A. Elmahdi Bourahla and M. Bourahla, "A technique for developping mobile applications," *ISIA 2020 - Proceedings, 4th Int. Symp. Informatics its Appl.*, no. Figure 1, 2020, doi: 10.1109/ISIA51297.2020.9416548.

[26] B. P. D. Putranto, R. Saptoto, O. C. Jakaria, and W. Andriyani, "A Comparative Study of Java and Kotlin for Android Mobile Application Development," *2020 3rd Int. Semin. Res. Inf. Technol. Intell. Syst. ISRITI 2020*, pp. 383–388, 2020, doi: 10.1109/ISRITI51436.2020.9315483.

[27] B. Vaishak, S. Hoysala, V. H. Pavankumar, and Mohana, "Currency and Fake Currency Detection using Machine Learning and Image Processing - An Application for Blind People using Android Studio," *Int. Conf. Autom. Comput. Renew. Syst. ICACRS 2022 - Proc.*, no. Icacrs, pp. 274–277, 2022, doi: 10.1109/ICACRS55517.2022.10029296.

[28] R. Sharma, T. U. Bux, B. Varshney, and K.

Tomar, “Real-time Student Management Application Using Google Firebase and Android Studio,” *2021 Int. Conf. Intell. Technol. CONIT 2021*, pp. 1–6, 2021, doi: 10.1109/CONIT51480.2021.9498494.
[29] P. K. Bhuyan, M. S., Chakravarty, S., Das, S., & Bora, “Real time text detection and audio feedback system for the visually impaired,” 2019.
[30] K. Maharana, S. Mondal, and B. Nemade, “A review: Data pre-processing and data augmentation techniques,” *Glob. Transitions Proc.*, vol. 3, no. 1, pp. 91–99, 2022, doi: 10.1016/j.gltp.2022.04.020.
[31] S. Abdoli, P. Cardinal, and A. Lameiras Koerich, “End-to-end environmental sound classification using a 1D convolutional neural network,” *Expert Syst. Appl.*, vol. 136, pp. 252–263, 2019, doi: 10.1016/j.eswa.2019.06.040.
[32] A. Sarkar, A. Goyal, D. Hicks, D. Sarkar, and S. Hazra, “Android Application Development: A Brief Overview of Android Platforms and Evolution of Security Systems,” *Proc. 3rd Int. Conf. I-SMAC IoT Soc. Mobile, Anal. Cloud, I-SMAC 2019*, pp. 73–79, 2019, doi: 10.1109/I-SMAC47947.2019.9032440.
[33] K. Gupta, D. Oladimeji, C. Varol, A. Rasheed, and N. Shahshidhar, “A Comprehensive Survey on Artifact Recovery from Social Media Platforms: Approaches and Future Research Directions,” *Inf.*, vol. 14, no. 12, 2023, doi: 10.3390/info14120629.
[34] M. M. Hannuksela and Y. K. Wang, “An Overview of Omnidirectional MediA Format (OMAF),” *Proc. IEEE*, vol. 109, no. 9, pp. 1590–1606, 2021, doi: 10.1109/JPROC.2021.3063544.
[35] M. Shamsujjoha, J. Grundy, L. Li, H. Khalajzadeh, and Q. Lu, “Developing Mobile Applications Via Model Driven Development: A Systematic Literature Review,” *Inf. Softw. Technol.*, vol. 140, no. March, 2021, doi: 10.1016/j.infsof.2021.106693.
[36] T. Hirsch and B. Hofer, “A systematic literature review on benchmarks for evaluating debugging approaches,” *J. Syst. Softw.*, vol. 192, p. 111423, 2022, doi: 10.1016/j.jss.2022.111423.
[37] A. Z. H. Yang, S. Hassan, Y. Zou, and A. E. Hassan, “An empirical study on release notes patterns of popular apps in the Google Play Store,” *Empir. Softw. Eng.*, vol. 27, no. 2, 2022, doi: 10.1007/s10664-021-10086-2.
[38] R. Lin, “Analysis on the Selection of the Appropriate Batch Size in CNN Neural Network,” *Proc. - 2022 Int. Conf. Mach. Learn. Knowl. Eng. MLKE 2022*, pp. 106–109, 2022, doi: 10.1109/MLKE55170.2022.00026.
[39] I. Kandel and M. Castelli, “The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset,” *ICT Express*, vol. 6, no. 4, pp. 312–315, 2020, doi: 10.1016/j.icte.2020.04.010.
[40] R. Zaheer and H. Shaziya, “A Study of the Optimization Algorithms in Deep Learning,” *Proc. 3rd Int. Conf. Inven. Syst. Control. ICISC 2019*, no. Icisc, pp. 536–539, 2019, doi: 10.1109/ICISC44355.2019.9036442.
[41] A. Taherkhani, G. Cosma, and T. M. McGinnity, “AdaBoost-CNN: An adaptive boosting algorithm for convolutional neural networks to classify multi-class imbalanced datasets using transfer learning,” *Neurocomputing*, vol. 404, pp. 351–366, 2020, doi: 10.1016/j.neucom.2020.03.064.
[42] M. Muhajir, K. Muchtar, M. Oktiana, and A. Bintang, “Students ’ emotion classification system through an ensemble approach,” vol. 28, no. 2, pp. 413–424, 2024.
[43] D. Vetrithangam, P. M. Ebin, P. Nareshkumar, B. Arunadevi, A. Fathima, and A. Ramesh Kumar, “Enhanced Vgg-19 Model for Accurate Brain Tumor Prediction,” *2023 Int. Conf. Network, Multimed. Inf. Technol. NMITCON 2023*, no. Ml, pp. 1–6, 2023, doi: 10.1109/NMITCON58196.2023.10276235.
[44] T. Sivakumari and R. Vani, “Implementation of AlexNet for Classification of Knee Osteoarthritis,” *7th Int. Conf. Commun. Electron. Syst. ICCES 2022 - Proc.*, no. Icces, pp. 1405–1409, 2022, doi: 10.1109/ICCES54183.2022.9835835.
[45] A. Bhattacharyya, D. Bhaik, S. Kumar, P. Thakur, R. Sharma, and R. B. Pachori, “A deep learning based approach for automatic detection of COVID-19 cases using chest X-ray images,” *Biomed. Signal Process. Control*, vol. 71, no. PB, p. 103182, 2022, doi: 10.1016/j.bspc.2021.103182.
[46] S. Ji, Z. Qin, J. Shan, and M. Lu, “Panoramic SLAM from a multiple fisheye camera rig,” *ISPRS J. Photogramm. Remote Sens.*, vol. 159, no. November 2019, pp. 169–183, 2020, doi: 10.1016/j.isprsjprs.2019.11.014.