JUIT
ADMI Jurnal Ilmiah Teknik
OPEN ACCESS
JUIT Vol 5 No. 1 | January 2026 | ISSN: 2828-6936 (Print), ISSN: 2828-6901 (online), Page. 98-113

Rupiah Banknote Classification Using MobileNetV2 Based on Image Data

Aditya Rizky Fajriansyah[1*], Dina Agusten[2], Sri Rahayu Puspita Sari[3], Fauziah[4], Yuli Fitriyani[5], Mariza Wijayanti[6]
[1,2]Informatics, Industrial Technology, Gunadarma University, Depok, Indonesia
[3]Economics, Economics, Gunadarma University, Depok, Indonesia
[4,5]Computer System, Computer Science and Information Technology, Gunadarma University, Depok, Indonesia
[6]Electrical Engineering, Industrial Technology, Gunadarma University, Depok, Indonesia

Article History
Received : January 06, 2026
Accepted : January 12, 2026
Published : January 13, 2026
Available Online: January 13, 2026

Corresponding author*:
Adityafajriansyah84@gmail.com

Cite This Article:
Aditya Rizky Fajriansyah, Dina Agusten, Sri Rahayu Puspita Sari, Fauziah, Yuli Fitriyani, & Mariza Wijayanti. (2025). Rupiah Banknote Classification Using MobileNetV2 Based on Image Data. Jurnal Ilmiah Teknik, 5(1), 98–113.

DOI:
https://doi.org/10.56127/juit.v5i1.2461

Abstract: Visually impaired individuals often struggle to independently identify Indonesian rupiah denominations because banknotes share similar colors, patterns, and layouts, increasing the risk of errors during cash transactions. **Purpose:** This study aims to develop an offline, image-based banknote denomination recognition system on Android that can provide accessible assistance without relying on internet connectivity. **Methodology:** A quantitative experimental design was applied using a dataset of 4,340 banknote images covering 14 classes (seven denominations with front and back sides). The classifier was built with MobileNetV2 using transfer learning, supported by data augmentation and hyperparameter optimization, and evaluated using validation accuracy and F1-score. The trained model was converted to TensorFlow Lite and integrated into a Flutter-based Android application with text-to-speech output for user assistance. **Findings:** The proposed model achieved 96.20% validation accuracy with an average F1-score of 0.95, indicating strong performance for lightweight on-device inference. **Implications:** The system can be deployed in real time on smartphones to support inclusive and safer cash handling for visually impaired users, and it demonstrates the feasibility of offline deep learning for accessible financial technology. **Originality:** This study provides an end-to-end offline solution for Indonesian rupiah recognition that combines a lightweight deep learning model, on-device deployment, and text-to-speech feedback while distinguishing both sides of multiple denominations, offering practical value beyond approaches that depend on cloud inference or limited class coverage.

Keywords: Android; Flutter; Image Classification; MobileNetV2; Visually Impaired Users

INTRODUCTION

Indonesian rupiah banknotes exhibit high visual variability due to differences in denomination design, front–back orientation, color composition, and security patterns. In real-world usage, banknotes are frequently captured under uncontrolled conditions, including non-uniform lighting, cluttered backgrounds, partial occlusion, and varying camera angles when using mobile devices. These factors introduce significant challenges for image-based denomination identification, as visual features may be distorted, poorly illuminated, or visually similar across denominations. Consequently, reliable banknote

classification on mobile platforms requires robust feature extraction and efficient inference mechanisms capable of handling such real-world variability. Such limitations may reduce their autonomy in conducting everyday financial activities. Technology-based applications have become a promising solution to support visually impaired individuals in recognizing banknote denominations accurately and efficiently (Zakaria et al., 2023).

Recent advances in artificial intelligence have enabled computer vision systems to achieve high accuracy in image recognition tasks through deep learning methods (Chollet, 2021; Muchlis et al., 2025). Convolutional Neural Networks (CNNs) are widely used because of their ability to extract spatial features from images (Ghosh et al., 2019). MobileNetV2, designed for devices with limited computational resources through an inverted residual and linear bottleneck architecture, offers an efficient model with competitive accuracy. Research by (Aji et al., 2025) demonstrated that MobileNetV2 achieved 91.04% accuracy and an F1-score of 0.906 in classifying disease types and was successfully implemented on Android applications using TensorFlow Lite. These results indicate that MobileNetV2 is highly suitable for mobile-based image classification systems.

Assistive technologies for visually impaired individuals continue to grow, including applications such as Be My Eyes, which provides visual assistance through volunteers (Faruq & Harahap, 2024). Although effective, such systems require constant internet connectivity and external support, which may not always be available. This limitation highlights the need for an offline image-based classification system capable of independently recognizing banknote denominations. Such a system should be easy to use, fast, practical, and able to increase the financial autonomy of visually impaired users (Suhendro et al., 2021).

This research develops a banknote denomination classification system for Indonesian rupiah using MobileNetV2. The system is designed to operate locally on Android devices without requiring internet access. The dataset used consists of 4,340 images divided into 14 nominal classes. According to Mulyana & Akbar dataset limitations are a challenge in image classification development, particularly when many classes or varied motifs are involved (Mulyana & Akbar, 2022). Their findings emphasize that data augmentation can serve as a key technique to address dataset limitations (Saprudin et al., 2021). This supports the approach used in this research, where augmentation is applied to increase dataset diversity and model generalization.

Although several previous studies have addressed currency recognition, most focus on foreign currencies or require network connectivity. Studies on Indonesian banknotes, particularly the Indonesian rupiah banknotes from the 2022 emission series as officially released by Bank Indonesia, remain limited, and many do not integrate model deployment into an offline mobile application. Therefore, a system that incorporates deep learning-based classification, optimized for mobile deployment using TensorFlow Lite, and specifically targets the latest Indonesian banknote series can provide meaningful contributions (Novac et al., 2022). The aim of this research is to develop and evaluate a MobileNetV2-based image classification system for identifying Indonesian rupiah banknote denominations and to integrate the model into an offline Flutter-based Android application designed to support the financial independence of visually impaired individuals.

Research Objectives

This study is designed to achieve the following technical objectives, spanning model development, deployment, and validation on mobile devices:

Develop a MobileNetV2-based image classification model for 14 banknote classes (seven denominations × two sides) using transfer learning to improve accuracy under limited training data (Azunre, 2021; Lakshmanan et al., 2021).

Evaluate model performance using accuracy and loss trends across epochs, and conduct class-level analysis using a confusion matrix and precision, recall, and F1-score metrics (Palanivinayagam et al., 2023). Convert the trained model to TensorFlow Lite to enable efficient on-device inference and integrate it into a Flutter-based Android application with an accessibility-oriented interface and text-to-speech output (Suhendro et al., 2021; Warden & Situnayake, 2020). Validate application functionality and compatibility through black-box testing and multi-device testing to ensure stable operation across different Android specifications.

Technical Contribution

This work contributes an end-to-end, offline denomination-recognition pipeline for Indonesian rupiah banknotes that is practical for deployment on Android devices. The engineering contributions can be summarized as:

1. Dataset preparation that combines a public dataset with additional smartphone captures to better reflect real-world variability.
2. MobileNetV2 transfer-learning configuration optimized for multi-class banknote recognition on limited compute.
3. conversion and deployment of the trained model to TensorFlow Lite for efficient on-device inference
4. Accessibility-oriented Flutter application that provides both visual and audio feedback, validated through functional and multi-device testing.

RESEARCH METHOD

The research follows a structured workflow consisting of several sequential stages, beginning with dataset acquisition and ending with model deployment in a mobile application. The complete workflow is illustrated in Figure 1, which summarizes the overall process adopted in this study.

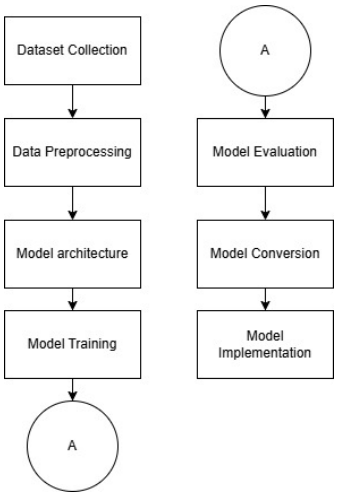


Figure 1. Research Workflow

The workflow includes collecting and preparing banknote images, applying preprocessing and augmentation techniques, designing and training a MobileNetV2-based deep learning model, evaluating its performance, converting the trained model into a mobile-compatible format, and implementing it within an Android application built using Flutter. Each stage ensures that the final system can classify Indonesian banknotes accurately and operate efficiently without depending on an internet connection. Detailed explanations of each stage are presented in the subsequent subsections.

Dataset Collection

The dataset used in this study consists of 4,340 images of Indonesian rupiah banknotes from the 2022 emission series. The images were obtained as a filtered subset of the publicly available rupiah-banknotes dataset published by Nurul Alfiyyah on the Kaggle platform, which contains banknote images from multiple Indonesian rupiah emission years ranging from 2016 to 2022.

In this research, only banknotes corresponding to the 2022 emission series were included. The selection of the 2022 series was conducted manually by matching each banknote image with the official design specifications and visual identifiers published by Bank Indonesia, such as denomination layout, dominant color composition, portrait design, typography style, and security feature placement. These characteristics are officially documented in the Bank Indonesia publication describing the Uang Rupiah Kertas Emisi 2022 (Indonesia, 2022), which serves as the authoritative reference for identifying the 2022 rupiah banknote series.

After emission-year filtering, the final dataset comprises 14 classes, representing seven rupiah denominations with front and back sides. Each class contains 310 images, resulting in a total of 4,340 images. From each class, 300 images were obtained from the Kaggle dataset after filtering for the 2022 emission series, while an additional 10 images per class were captured manually using a smartphone camera to introduce real-world variability and support practical mobile deployment scenarios.

The smartphone-captured images were acquired under varying conditions, including natural and artificial lighting, plain and cluttered backgrounds, and non-uniform camera angles and distances. All images from both sources were combined and divided into 80% training data and 20% validation data using stratified sampling to preserve class balance across all denominations during model training and evaluation.

Data Preprocessing

Data preprocessing is an essential stage in preparing the dataset for the model training process. The overall preprocessing workflow is illustrated in Figure 2, which summarizes the key steps applied before the images were used in model development.

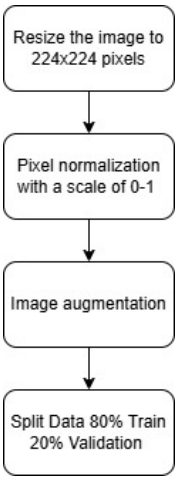


Figure 2. Data Preprocessing Workflow

All images were resized to 224 × 224 pixels to match the MobileNetV2 input requirements, followed by pixel normalization to the range of [0,1] to stabilize the learning process. The same preprocessing pipeline was applied consistently during both model training and on-device TensorFlow Lite inference to ensure identical input representation.

Data augmentation was applied to increase visual diversity and reduce overfitting, including variations in rotation (±20°), zoom (0.8–1.2), translation (±10%), brightness adjustment, shear transformation, and horizontal flipping. These transformations simulate real-world image variations commonly encountered during mobile image capture and help the model learn more robust visual features (Hossain & Sajib, 2019).

The dataset was then split into 80% training data and 20% validation data using a stratified sampling approach to maintain class balance across all classes (Andrianto et al., 2024). A brief visual inspection of augmented samples was performed to ensure that the applied transformations produced realistic and meaningful variations. All experiments were conducted using a fixed random seed to improve reproducibility. This preprocessing pipeline ensures that the dataset is sufficiently diverse and well-structured for improved model generalization.

Model Architecture

The model architecture used in this study is MobileNetV2, a lightweight convolutional neural network optimized for mobile and embedded devices through inverted residuals and linear bottleneck structures (Sandler et al., 2018). MobileNetV2 is designed to achieve a favorable balance between accuracy and computational efficiency, making it suitable for

deployment on resource-constrained platforms such as smartphones (Lakshmanan et al., 2021).

The model was initialized using ImageNet pre-trained weights, with the top classification layers removed to allow adaptation to the 14 banknote classes. A series of additional layers were incorporated, including a GlobalAveragePooling2D layer to reduce spatial dimensions while retaining essential feature representations, followed by a Dense layer with 128 units and ReLU activation. The final classification layer consists of 14 neurons with softmax activation to produce multiclass probability outputs.

During transfer learning, all pre-trained layers of MobileNetV2 were frozen to preserve the learned low-level visual features, while only the newly added layers were trained (Azunre, 2021). This approach enables efficient training while leveraging the representational power of the base model. The resulting architecture is computationally efficient and well suited for deployment on mobile platforms.

Model Training

The model was trained using the categorical cross-entropy loss function and the Adam optimizer with a learning rate of 1e-4. Training was conducted for 15 epochs with a batch size of 32 while monitoring accuracy and loss on both training and validation sets. Several epoch variations (10, 15, 20, 25) were tested, and 15 epochs produced the most stable accuracy without signs of overfitting or unnecessary training time. This configuration was therefore selected as the optimal setup for the model (McAuley, 2021).

Model Evaluation

Model evaluation was performed using the validation dataset to monitor performance and detect potential overfitting. The assessment focused on two main metrics: accuracy and loss observed across training epochs (Zhou, 2022). Trends in training and validation curves were used to determine the stability and learning behavior of the model. In addition to these metrics, a classification report and confusion matrix were generated to evaluate class-level performance, including precision, recall, and F1-score (Palanivinayagam et al., 2023). The confusion matrix helped identify the most accurately recognized classes as well as classes that were more prone to misclassification. Overall, the evaluation provided a comprehensive understanding of the model's predictive capabilities and generalization performance.

Model Conversion

The trained TensorFlow model was converted into TensorFlow Lite (TFLite) format to ensure efficient deployment on mobile devices (Warden & Situnayake, 2020). The conversion process utilized the TensorFlow TFLiteConverter API, producing a compact .tflite file optimized for fast inference and reduced resource usage. This lightweight model format is suitable for Android environments, allowing the classification system to operate effectively on devices with limited computational capacity. The converted model was subsequently stored and prepared for integration into the mobile application (Al Ghani & Andrian, 2023).

Model Implementation

The implementation phase involved integrating the trained TensorFlow Lite model into a Flutter-based mobile application. The development process adhered to the Waterfall Software Development Life Cycle, as illustrated in Figure 3, to ensure a systematic progression from requirements analysis to system testing (Abdillah, 2024).

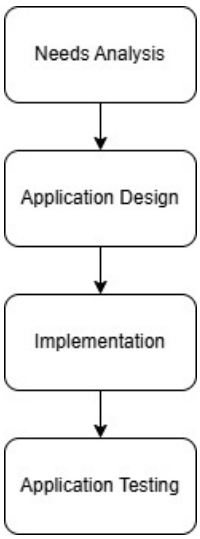


Figure 3. Waterfall SDLC Model

The requirements analysis stage identified both functional and non-functional specifications essential for the system. Functional requirements included the capability to acquire images through the camera or gallery, process them for model inference, and present classification results in real time. Non-functional requirements addressed performance, compatibility with various Android devices, and usability considerations to support users with accessibility needs. In the design stage, the system architecture and user

interface were structured to facilitate seamless interaction between the image acquisition module, preprocessing pipeline, and the TensorFlow Lite inference component. The interface layout and data flow were designed to ensure clarity, operational efficiency, and accessibility (Akbar et al., 2021).

The implementation stage translated the system design into a working application using Flutter. The TensorFlow Lite model was integrated to enable on-device inference, while the preprocessing procedures such as image resizing and normalization were implemented to align input data with the model's specifications. Finally, the application underwent black-box testing to evaluate its functional correctness, responsiveness, and stability under various usage scenarios. Testing across multiple Android devices confirmed that the system performed as expected, with accurate classification results and consistent real-time operation (Prayogi, 2023).

RESULTS AND DISCUSSION

Experimental Setup

All experiments were conducted using the MobileNetV2 architecture trained with transfer learning on a dataset of 4,340 banknote images across 14 classes. The model was trained for 15 epochs using TensorFlow and later converted into TensorFlow Lite for deployment in a Flutter-based Android application. Evaluation was performed using a dedicated validation set to assess generalization and real-time performance on mobile devices.

Training Results

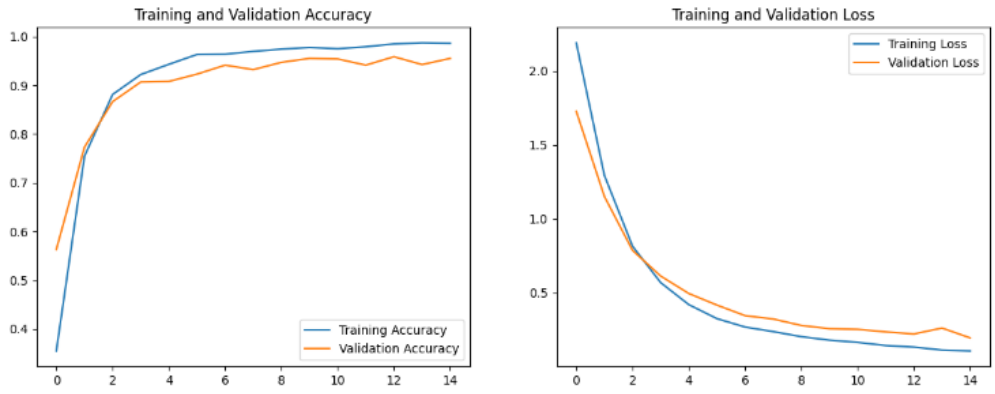


Figure 4. Training Results

The reported accuracy values correspond to different evaluation points. The validation accuracy of 95.51% was obtained at the final training epoch (epoch 15), while the highest validation accuracy of 96.20% was achieved at epoch 13. The latter is reported as the final model performance and was selected for TensorFlow Lite conversion.

Training performance improved steadily, as shown in Figure 4, where training accuracy increased from 21.23% to 98.54%, and validation accuracy rose from 56.34% to 95.51%. Loss curves in Figure 4 also indicate consistent convergence, with no signs of overfitting.

```
/usr/local/lib/python3.11/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: UserWarning: Your `PyDataset`
  self._warn_if_super_not_called()
Epoch 1/15
109/109 ━━━━━━━━━━━━━━━━━━━━ 1724s 16s/step - accuracy: 0.2123 - loss: 2.5343 - val_accuracy: 0.5634 - val_loss: 1.7262
Epoch 2/15
109/109 ━━━━━━━━━━━━━━━━━━━━ 261s 2s/step - accuracy: 0.6981 - loss: 1.4586 - val_accuracy: 0.7730 - val_loss: 1.1506
Epoch 3/15
109/109 ━━━━━━━━━━━━━━━━━━━━ 256s 2s/step - accuracy: 0.8710 - loss: 0.8956 - val_accuracy: 0.8664 - val_loss: 0.7859
Epoch 4/15
109/109 ━━━━━━━━━━━━━━━━━━━━ 223s 2s/step - accuracy: 0.9095 - loss: 0.6129 - val_accuracy: 0.9067 - val_loss: 0.6123
Epoch 5/15
109/109 ━━━━━━━━━━━━━━━━━━━━ 220s 2s/step - accuracy: 0.9446 - loss: 0.4344 - val_accuracy: 0.9078 - val_loss: 0.4942
Epoch 6/15
109/109 ━━━━━━━━━━━━━━━━━━━━ 229s 2s/step - accuracy: 0.9582 - loss: 0.3460 - val_accuracy: 0.9228 - val_loss: 0.4159
Epoch 7/15
109/109 ━━━━━━━━━━━━━━━━━━━━ 220s 2s/step - accuracy: 0.9606 - loss: 0.2820 - val_accuracy: 0.9412 - val_loss: 0.3444
Epoch 8/15
109/109 ━━━━━━━━━━━━━━━━━━━━ 223s 2s/step - accuracy: 0.9702 - loss: 0.2366 - val_accuracy: 0.9320 - val_loss: 0.3217
Epoch 9/15
109/109 ━━━━━━━━━━━━━━━━━━━━ 227s 2s/step - accuracy: 0.9710 - loss: 0.2123 - val_accuracy: 0.9470 - val_loss: 0.2780
Epoch 10/15
109/109 ━━━━━━━━━━━━━━━━━━━━ 223s 2s/step - accuracy: 0.9782 - loss: 0.1820 - val_accuracy: 0.9551 - val_loss: 0.2562
Epoch 11/15
109/109 ━━━━━━━━━━━━━━━━━━━━ 230s 2s/step - accuracy: 0.9797 - loss: 0.1599 - val_accuracy: 0.9539 - val_loss: 0.2518
Epoch 12/15
109/109 ━━━━━━━━━━━━━━━━━━━━ 222s 2s/step - accuracy: 0.9834 - loss: 0.1347 - val_accuracy: 0.9412 - val_loss: 0.2346
Epoch 13/15
109/109 ━━━━━━━━━━━━━━━━━━━━ 223s 2s/step - accuracy: 0.9862 - loss: 0.1278 - val_accuracy: 0.9585 - val_loss: 0.2204
Epoch 14/15
109/109 ━━━━━━━━━━━━━━━━━━━━ 227s 2s/step - accuracy: 0.9868 - loss: 0.1141 - val_accuracy: 0.9424 - val_loss: 0.2606
Epoch 15/15
109/109 ━━━━━━━━━━━━━━━━━━━━ 258s 2s/step - accuracy: 0.9854 - loss: 0.1107 - val_accuracy: 0.9551 - val_loss: 0.1940
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is
```

Figure 5. Training Results

Evaluation Results

A. Confusion Matrix

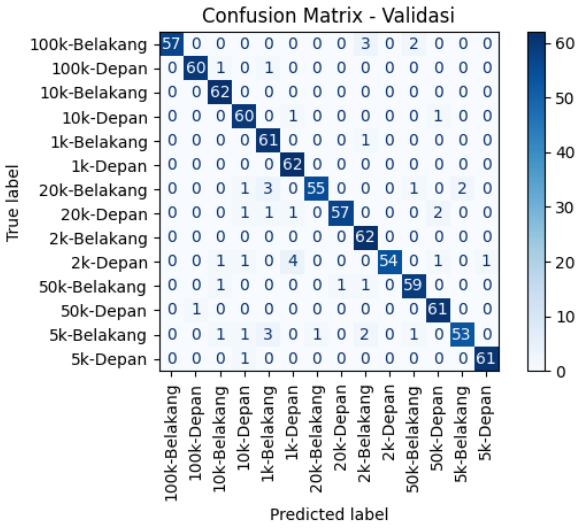


Figure 6. Confusion Matrix

As illustrated in Figure 5, most predictions lie along the diagonal, demonstrating strong correctness across all 14 classes. Minor misclassification occurred primarily between front–back variants of the same denomination.

B. Classification Report

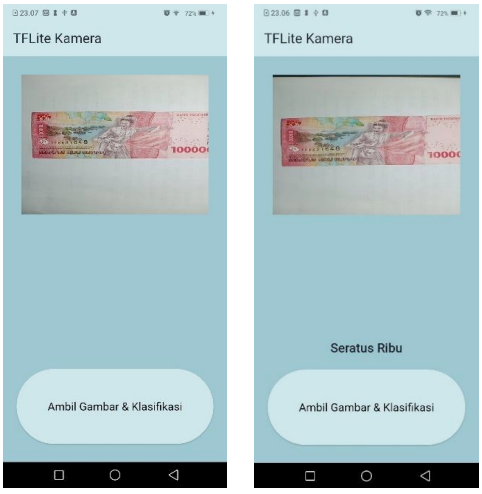
[✓] Classification Report:

	precision	recall	f1-score	support
100k-Belakang	1.00	0.92	0.96	62
100k-Depan	0.98	0.97	0.98	62
10k-Belakang	0.94	1.00	0.97	62
10k-Depan	0.92	0.97	0.94	62
1k-Belakang	0.88	0.98	0.93	62
1k-Depan	0.91	1.00	0.95	62
20k-Belakang	0.98	0.89	0.93	62
20k-Depan	0.98	0.92	0.95	62
2k-Belakang	0.90	1.00	0.95	62
2k-Depan	1.00	0.87	0.93	62
50k-Belakang	0.94	0.95	0.94	62
50k-Depan	0.94	0.98	0.96	62
5k-Belakang	0.96	0.85	0.91	62
5k-Depan	0.98	0.98	0.98	62
accuracy			0.95	868
macro avg	0.95	0.95	0.95	868
weighted avg	0.95	0.95	0.95	868

Figure 7. Classification Report

The classification report in Figure 6 shows high consistency across all metrics, with overall precision, recall, and F1-score of 0.95. Class “10k-Belakang” achieved 0.94 precision, 1.00 recall, and 0.97 F1-score.

Mobile Application Performance



(a) (b)

Figure 8. Mobile Application Performance

The trained MobileNetV2 model was converted into TensorFlow Lite format to enable efficient on-device inference within the Flutter-based Android application. The resulting TensorFlow Lite model has a compact size of 9.3 MB, making it suitable for deployment on mobile devices with limited storage and computational resources.

Application-level performance was evaluated through functional testing and multi-device testing across several Android smartphones with different hardware specifications. During testing, the application demonstrated responsive behavior, where classification results were produced shortly after image capture without noticeable delay from a user perspective. This indicates that the proposed system is capable of near real-time performance for practical mobile-based banknote recognition.

Table 1. Blackbox Testing

Test Function	Expected Output	Result Description
Capture image from camera	Image is displayed in the application and ready for classification	Successful
Banknote denomination classification	Output label appears according to the detected banknote value (e.g, "Lima Puluh Ribu")	Successful
Display classification result	Classification results is presented in text and audio form	Successful
Application interface response	All buttons and navigation respond quickly without lag or crashes	Stable and responsive
Classification audio output (TTS)	System reads the classification result clearly and accurately	Audio output clear and correct

Table 2. Multi Device Testing

Device Spesification	Result	Result Description
• Exynos 9611 • Android 13 (One UI) • 4 GB + 64 GB • 2340 × 1080 FHD+ 6.4 inches Super AMOLED	Successful	The application runs properly. The interface layout matches the intended design.
• MediaTek Helio P65 • Android 12 • 6 GB + 128 GB • 2340 x 1080 6.53 inches IPS LCD	Successful	The application runs properly. The interface layout matches the intended design.
• MediaTek Helio G95 • Android 11 • 8 GB + 128 GB • 2400 X 1080 6.43 inches AMOLED	Successful	The application runs properly. The interface layout matches the intended design.

Device Spesification	Result	Result Description
• Snapdragon 7s Gen 2 • Android 13 (MUI for POCO/HyperOS) • 8 GB + 256 GB • 2400 X 1080 6.67 inches AMOLED 120Hz	Successful	The application runs properly. The interface layout matches the intended design.

Device compatibility tests summarized in Table 1 (Blackbox Testing) and Table 2 (Multi-device Testing) confirm that the system ran successfully on all tested Android devices, with correct image capture, classification output, and TTS feedback.

DISCUSSION

The observed performance of the proposed MobileNetV2-based classification system reflects a balanced trade-off between accuracy and computational efficiency for mobile deployment. Rather than merely indicating high predictive scores, the results suggest that the model has learned denomination-specific visual representations that remain stable across diverse capture conditions commonly encountered during smartphone-based image acquisition.

A closer examination of the confusion patterns reveals that most misclassifications occur between the front and back sides of the same denomination. This behavior is primarily driven by strong visual similarities, including shared color distributions, dominant portrait regions, and repetitive security patterns present on both sides of the banknotes. Since the system relies on single-frame RGB images without explicit side-orientation cues, these subtle spatial differences may not always be sufficiently discriminative, particularly when images are captured under non-ideal lighting or angled viewpoints.

Additional error sources were identified under challenging lighting conditions, such as low illumination or uneven light distribution, which can reduce contrast and obscure fine-grained visual features. Although data augmentation improves robustness to moderate variations, extreme lighting conditions still affect feature clarity, highlighting an inherent limitation of lightweight convolutional architectures optimized for mobile inference.

Concrete failure modes were also observed during application-level testing. In cases where banknotes were partially captured, folded, or photographed at steep angles, the model occasionally predicted the incorrect side of the correct denomination rather than a

completely different value. This indicates that the learned features remain denomination-consistent but are sensitive to spatial completeness and orientation. Importantly, such errors did not compromise the overall usability of the system, as incorrect predictions were typically confined to closely related classes.

From an engineering perspective, these findings underscore the practical implications of deploying compact deep learning models on resource-constrained devices. While MobileNetV2 enables efficient offline inference suitable for real-time mobile applications, improving robustness against extreme capture conditions may require additional mechanisms, such as orientation-aware preprocessing or lightweight attention modules. Nevertheless, the current system demonstrates sufficient reliability and efficiency to support visually impaired users in independently identifying Indonesian rupiah banknotes in real-world scenarios.

CONCLUSION

This study successfully developed an image-based classification system for Indonesian banknotes using the MobileNetV2 architecture integrated into a Flutter-based Android application. The model accurately recognized 14 banknote classes and achieved a final validation accuracy of 96.20%, supported by strong precision, recall, and F1-scores across all classes. The system demonstrated reliable performance under various lighting conditions and camera orientations, and the TensorFlow Lite deployment enabled efficient real-time inference on mobile devices. The integrated text-to-speech feature further enhanced usability for visually impaired users, enabling them to identify banknote denominations independently without internet connectivity.

Despite its strong performance, several areas can be improved in future development. The application interface can be made more adaptive to increase accessibility, particularly for visually impaired users. Model robustness could be enhanced to correctly classify partially captured banknotes, worn banknotes, or images with low visual quality. Additionally, sensitivity to low-light conditions should be further optimized, as performance declines in dim or uneven illumination. Future work may explore advanced data augmentation, improved mobile optimization, or broader applicability such as counterfeit detection or real-time financial accessibility tools.

REFERENCES

Abdillah, S. P. N. (2024). Perancangan Aplikasi Website Sistem Informasi Alumni Menggunakan PHPRad. *Indonesian Journal of Computer Science*. https://doi.org/10.33022/ijcs.v13i1.3657

Aji, N. B. et al. (2025). Application of MobileNetV2-Based Deep Learning in Detecting Diseases in Chili Plants. *Journal of Informatics, Information System, Software Engineering and Applications (INISTA)*, *7*(2), 203–212. https://doi.org/10.20895/INISTA.V7I2.1825

Akbar, R. et al. (2021). Penerapan Aplikasi Berbasis Web Untuk Monitoring Pengobatan Pasien Gangguan Jiwa Pada UPT Puskesmas Pasar Usang. *Jurnal Nasional Teknologi Dan Sistem Informasi*. https://doi.org/10.25077/teknosi.v7i3.2021.130-137

Al Ghani, A. I., & Andrian, R. (2023). Pengembangan Presensee: Aplikasi Presensi Mahasiswa Mobile Menggunakan Framework Flutter (Studi Kasus: Studi Independen Alterra Academy. *Jurnal Media Infotama*, *19*(2), 447–453. https://doi.org/10.37676/jmi.v19i2.4351

Andrianto, F. et al. (2024). Linear Kernel Optimization of Support Vector Machine Algorithm on Online Marketplace Sentiment Analysis. *Komputasi: Jurnal Ilmiah Ilmu Komputer Dan Matematika*, *21*(1), 68–82. https://doi.org/10.33751/komputasi.v21i1.9266

Azunre, P. (2021). *Transfer Learning for Natural Language Processing*. Manning.

Chollet, F. (2021). *Deep Learning with Python*. Manning.

Faruq, F., & Harahap, L. S. (2024). Analisis Sentimen Ulasan Aplikasi Be My Eyes Menggunakan Data Orange Mining. *Jurnal Multidisiplin Saintek*, *5*(Table 10), 4–6.

Ghosh, A. et al. (2019). Fundamental concepts of convolutional neural network. In *Intelligent Systems Reference Library* (Vol. 172). https://doi.org/10.1007/978-3-030-32644-9_36

Hossain, A., & Sajib, S. A. (2019). Classification of Image using Convolutional Neural Network (CNN). *Global Journal of Computer Science and Technology*, *19*(2). https://doi.org/10.34257/GJCSTDVOL19IS2PG13

Indonesia, B. (2022). *Uang rupiah kertas emisi 2022*. Bank Indonesia. https://www.bi.go.id/id/fungsi-utama/sistem-pembayaran/uang-rupiah/Contents/Default.aspx

Lakshmanan, V. et al. (2021). *Practical Machine Learning for Computer Vision: End-to-End Machine Learning for Images*. O'Reilly Media, Inc.

McAuley, J. (2021). Personalized Machine Learning. In *CAMBRIDGE UNIVERSITY PRESS*. https://doi.org/10.1017/9781009003971

Muchlis, A. et al. (2025). The Effect Of Data Augmentation On Accuracy Values In Fabric Defect Detection. *Revista de Gestão Social e Ambiental*, *19*(3), 1–17. https://doi.org/10.24857/rgsa.v19n3-048

Mulyana, D. I., & Akbar, A. (2022). Optimasi Klasifikasi Batik Betawi Menggunakan Data Augmentasi Dengan Metode KNN dan GLCM. *Jurnal Aplikasi Teknologi Informasi Dan Manajemen (JATIM)*, *3*(2), 92–101. https://doi.org/10.31102/jatim.v3i2.1577

Novac, O. C. et al. (2022). Analysis of the Application Efficiency of TensorFlow and PyTorch in Convolutional Neural Network. *Sensors*, *22*(22), 1–23. https://doi.org/10.3390/s22228872

Palanivinayagam, A. et al. (2023). Twenty Years of Machine-Learning-Based Text Classification: A Systematic Review. *Algorithms*, *16*(5). https://doi.org/10.3390/a16050236

Prayogi, N. (2023). Aplikasi Presensi Kegiatan Menggunakan QR Code Dan Digital Signature Pada Dinas Kominfo Kabupaten Gresik. *Jurnal Teknologi Dan Informasi*. https://doi.org/10.34010/jati.v13i2.9432

Sandler, M. et al. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 4510–4520). https://doi.org/10.1109/CVPR.2018.00474

Saprudin et al. (2021). Klasifikasi Citra Menggunakan Metode Random Forest dan Sequential Minimal Optimization (SMO). *Jurnal Sistem Dan Teknologi Informasi (Justin)*, *9*(2), 132. https://doi.org/10.26418/justin.v9i2.44120

Suhendro, J. M. et al. (2021). Rancang Bangun Aplikasi Seluler Penyedia Jasa Perawatan dan Kecantikan Menggunakan Framework Flutter. *Jurnal Spektrum*, *8*(2), 68–82.

Warden, P., & Situnayake, D. (2020). Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers. In *O'Reilly Media*. O'Reilly Media.

Zakaria, P. A. L. et al. (2023). Pengenalan Nilai Mata Uang Kertas Untuk Tunanetra Berbasis Android. *Jurnal Sintaks Logika*, *3*(3), 40–44. https://doi.org/10.31850/jsilog.v3i3.2587

Zhou, H. (2022). Research of Text Classification Based on TF-IDF and CNN-LSTM. *Journal of Physics: Conference Series*, *2171*(1), 1–9. https://doi.org/10.1088/1742-6596/2171/1/012021