İSTANBUL UNIVERSITY PRESS

Acta Infologica
acin 2024
DOI: 10.26650/acin.1447456

Research Article

Turkish Lira Banknote Classification using Transfer Learning and Deep Learning

Mirsat Yeşiltepe[1], Harun Elkıran[2], Jawad Rasheed[2]

[1]Yıldız Technical University, Department of Mathematical Engineering, İstanbul, Türkiye
[2]İstanbul Sabahattin Zaim University, Department of Computer Engineering, İstanbul, Türkiye

Corresponding author : Mirsat Yeşiltepe
E-mail : mirsaty@yildiz.edu.tr

ABSTRACT

With the increasing exchange of foreign currencies due to globalization, there is a need for systems that can recognize and validate multiple currencies in real time. Such systems facilitate smooth international transactions and support the finance sector in dealing with diverse currencies. This study focuses on classifying Turkish banknotes using deep learning models. The dataset comprises 6901 images of six different denominations (5 TL, 10 TL, 20 TL, 50 TL, 100 TL, and 200 TL) under various conditions, such as flat, angled, curved, and bent. The proposed model implements pre-trained models, including VGG16, VGG19, DenseNet121, DenseNet169, DenseNet201, MobileNet, and MobileNetV2, to classify the images. Different image sizes (50x50, 100x100, 150x150, and 200x200) and optimizers (SGD, RMSprop, Adam, Adamax, etc.) were tested to determine the most effective combinations. The best result was achieved with DenseNet201 with an image size of 200 and the SGD optimizer, achieving an accuracy of 98.84% in 12 epochs. Smaller image sizes (50x50) resulted in reduced performance for all models. In addition, models such as DenseNet169 and DenseNet121 also demonstrated high performance; however, MobileNetV2 struggled with smaller images.

Keywords: DenseNet201, Optimizer, Banknote, Convolution, Accuracy.

Submitted : 06.03.2024
Revision Requested : 26.08.2024
Last Revision Received : 17.10.2024
Accepted : 18.10.2024
Published Online : 03.12.2024

1. INTRODUCTION

Banknote classification systems are increasingly used in applications such as self-service kiosks, vending machines, and public transportation systems. With the increasing exchange of foreign currencies due to globalization, the rise in counterfeit banknotes poses a significant threat to financial systems worldwide. Moreover, many developing countries have limited access to reliable financial transaction infrastructure. Automated banknote classification systems, especially when designed to work on mobile platforms, provide affordable and accessible solutions for these regions and promote financial inclusion.

The classification and detection of banknotes are essential tasks in the financial and retail sectors, particularly as global currency circulation and automated cash-handling processes increase in complexity. With the increasing threat of counterfeit banknotes infiltrating markets, developing advanced systems to accurately classify and authenticate currency has become a pressing need. Traditionally, manual banknote inspection has been used; however, it is prone to human error and is inefficient in environments that demand high-speed processing. Automated banknote classification driven by recent advances in artificial intelligence (AI) and machine learning offers robust solutions for real-time and large-scale cash-handling applications.

Deep learning, a subset of machine learning, has gained attention as a powerful tool for image classification tasks, including banknote recognition. Convolutional Neural Networks (CNNs), in particular, have proven effective in learning complex features from banknote images, such as intricate patterns, watermarks, and holograms, that distinguish real currency from counterfeit. The ability of deep learning models to automatically extract hierarchical features from raw data makes them well-suited for handling the variations and challenges in currency classification. Transfer learning, where pre-trained models are fine-tuned for banknote classification tasks, further enhances system performance, particularly when limited datasets are available.

The need for accurate and efficient banknote classification is underscored by its diverse applications in automated teller machines (ATMs), vending machines, and cashier systems, as well as in global foreign exchange operations. Furthermore, counterfeit detection is of paramount importance because the proliferation of fake currency can disrupt economies and undermine trust in financial systems. By leveraging deep learning techniques, automated systems can not only identify legitimate currency and provide a defense against fraud by providing real-time analysis in high-volume environments, such as banks, retail stores, and transportation hubs.

The classification of banknotes using machine learning and deep learning techniques has attracted considerable attention in recent years due to the increasing need for automated systems capable of distinguishing between real and counterfeit notes. Early research in this area primarily relied on traditional machine learning techniques, such as support vector machines (SVMs) and decision trees, which manually extracted features like edge detection, color histograms, and texture analysis for classification tasks. Although these methods provided moderate success, their reliance on manual feature engineering limited their scalability and accuracy when confronted with the complex patterns present in modern currency designs. These limitations have paved the way for more advanced techniques that leverage the power of deep learning models, particularly CNNs, to address the shortcomings of earlier approaches.

One of the key breakthroughs in this field was the application of neural networks to banknote recognition, as introduced by (Baek et al., 2018). Their study demonstrated that CNNs can effectively classify banknotes by automatically learning hierarchical features directly from raw images without requiring manual feature extraction. This study demonstrated that deep learning models can outperform traditional methods in terms of both accuracy and efficiency. The authors employed a multilayer CNN architecture that captured spatial hierarchies, making it highly suitable for identifying intricate patterns on banknotes, such as watermarks, holographic images, and fine textures. This approach provides a foundation for the subsequent adoption of deep learning models in the field of banknote classification and counterfeit detection.

Subsequently, other researchers explored the use of transfer learning to enhance the performance of banknote classification systems, particularly when limited data were available. (Prakash et al., 2023) introduced a deep learning approach combining CNNs for image analysis and recurrent neural networks (RNNs) for security feature assessment in counterfeit banknote detection. Testing on a dataset of real and fake notes, the ensemble model shows superior accuracy of 98.36% and precision of 96.8% compared to traditional methods. The transfer learning approach enables the system to generalize well to unseen data, thereby making it more robust and scalable across different currencies and denominations

In addition, multispectral imaging combined with deep learning has been explored as a way to further improve counterfeit detection. (Wang et al., 2022) presented an automated approach using optical coherence tomography (OCT) and machine learning to classify counterfeit banknotes by analyzing internal features. By training classifiers on OCT

image-derived features, the study achieved high accuracy in detecting and categorizing counterfeits, with the support vector machine model reaching a sensitivity of 96.55% and specificity of 98.85%. Their research underscored the potential of integrating multispectral data with CNNs, expanding the scope of banknote classification systems to better handle sophisticated counterfeiting techniques that rely on non-visible features

In addition to CNNs and multispectral imaging, other studies have focused on lightweight architectures for real-time banknote classification in mobile and embedded systems. (Linkon et al., 2020) evaluated lightweight CNN models with transfer learning for Bangladeshi banknote classification using ResNet152v2, MobileNet, and NASNetMobile on two datasets with 8000 and 1970 images, respectively. The experiment shows that MobileNet achieved 98.88% on the larger dataset, NASNetMobile reached 100% on the smaller dataset, and MobileNet achieved 97.77% on the combined dataset. This development has broadened the applicability of automated banknote classification, making it accessible in areas where high-end hardware is unavailable.

Several hybrid approaches that combine image processing techniques with machine learning models have also been proposed for counterfeit detection. (Pachón, Ballesteros, and Renza, 2023) presented a pruning technique for sequential CNNs that reshaped network layers and was tested on models including AlexNet, VGG11, and VGG16, using a dataset of Colombian peso banknotes. The pruned models achieved up to a 75% reduction in parameters and computational load (FLOPs) with minimal accuracy loss, while models with higher pruning rates (up to 95%) showed more significant accuracy drops, especially for AlexNet and VGG16. Although more computationally intensive, these hybrid approaches show promise in handling difficult cases where counterfeiters use advanced printing techniques.

Similarly, much research has been conducted on the classification and recognition of banknotes from different countries. Authors in (Galeana Pérez and Bayro Corrochano, 2018) proposed a scheme to classify Mexican and Euro banknotes. In contrast, some work on Indian banknotes has also been done, such as (Mittal and Mittal, 2018) proposed a CNN-based deep learning model to classify Indian banknotes according to various states and positions. Their work achieved an accuracy of 0.966. In (Veeramsetty, Singal, and Badal, 2020), the authors tested various transfer learning models with several parameters to achieve an accuracy of 84.4

Research into automated banknote classification helps develop more sophisticated detection techniques that are vital for preventing fraud, maintaining currency integrity, and ensuring trust in financial systems. Researchers worldwide have implemented various automated banknote recognition and classification systems that reduce the need for manual intervention in industries that handle cash, such as retail, banking, and public transportation. Thus, such an automatic system can lower operational costs, streamlines workflows, and ensure high-speed processing of cash transactions. However, few studies on Turkish banknote classification have been reported. For example, Researchers in (Baykal et al., 2018) exploited a dataset, created using images of banknotes used in Turkey obtained under different conditions, and the relationship between classes was examined in terms of color and features. They used a pretrained DenseNet121 and other models to classify Turkish banknotes to achieve an accuracy of 93.15%. In another study (Khashman, Ahmed, and Mammadli, 2019), the authors focused on classifying countries based on banknotes used in different countries, including those used in Turkey. Their aim was to classify countries, not banknotes. The banknotes in the dataset's images are flat; thus, different shapes of banknotes were not tested. Similar studies like (Khashman and Sekeroglu, 2005) and (Khashman, Sekeroglu, and Dimililer, 2005) used Turkish notes to classify deformation rates of Turkish and Euro banknotes using artificial neural networks. However, the banknotes used in these two studies are no longer used in Turkey. In (İyikesici and Erçelebi, 2023), researchers exploited deep learning techniques to identify and detect counterfeit Turkish banknotes. However, they considered limited classes, including 5- and 10-unit Turkish banknotes.

These studies provide a comprehensive overview of the advances in banknote classification using deep learning and related techniques. The transition from traditional machine learning models to deep learning, along with innovations in transfer learning, multispectral imaging, and lightweight architectures, has significantly enhanced the accuracy and scalability of banknote classification systems. These studies collectively emphasize the importance of continuously improving automated systems to handle the evolving complexities of currency fraud and classification, thereby ensuring financial security and operational efficiency across industries.

Existing studies on banknote classification and counterfeit detection using deep learning have several shortcomings. First, less work is required to classify new Turkish banknotes. Second, most research does not focus on all units of banknotes (such as banknotes of 5 Turkish Lira (TL) and 10 TL), limiting their scalability to multi-unit scenarios, which are crucial in global financial systems. Third, prior studies considered a limited amount of data due to data availability and scarcity. Fourth, although some efforts have been made to develop lightweight models for mobile and embedded platforms, the real-time performance of such models, especially in low-resource environments, remains underexplored. In addition, counterfeit detection techniques primarily focus on visible features, often failing to detect more sophisticated counterfeits that use non-visible features such as ultraviolet ink and watermarks. Moreover, there are

limited emphasis on making these deep learning models interpretable and explainable, which is essential for regulatory compliance and trust in financial applications. These gaps suggest further research to address the complexity and scalability of banknote classification systems.

This study explores the application of deep learning models, such as CNNs and transfer learning, to develop a robust system for banknote classification. This study investigates various architectures and optimization techniques to enhance the accuracy and efficiency of banknote recognition systems. Ultimately, the goal is to contribute to advancing secure, scalable, and automated solutions that can be deployed across various industries to ensure operational efficiency, financial security, and improved customer experiences when handling physical currency. This study makes the following contributions:

- Merges two existing datasets to create a comprehensive dataset of Turkish banknotes, totaling 6901 images. These images represent banknotes in various conditions (flat, angled, curved) across six denominations (5 TL, 10 TL, 20 TL, 50 TL, 100 TL, and 200 TL), providing diversity in angles, sides, and shapes.
- We tested various pretrained deep learning models for banknote classification, including VGG16, VGG19, DenseNet121, DenseNet169, DenseNet201, MobileNet, and MobileNetV2.
- Examined the impact of different optimizers (SGD, RMSprop, Adam, etc.) and image sizes (50×50, 100×100, 150×150, and 200×200) to identify the most efficient combinations for accurate classification.
- In this study, the performance of different deep learning models in classifying Turkish banknotes is analyzed, and the most effective model is determined.
- By analyzing the confusion patterns between different banknote values, the learning process of the model and potential improvement areas can be evaluated.
- The training efficiency and overlearning problem will be addressed, and the necessary measures will be determined to optimize the performance of the model.
- Provides detailed analysis of how different combinations of model architectures, optimizers, and image sizes influence classification accuracy. It was found that no single parameter alone could guarantee high efficiency, and testing different parameter combinations was crucial for high performance.

The remainder of this paper is divided as follows: Section 2 describes the dataset and outlines the methods and models. Section 3 presents the experimental results, and Section 4 analyzes the results. Section 5 concludes the study.

2. DATA AND METHODOLOGY

With the increasing exchange of foreign currencies due to globalization, systems that can recognize multiple denominations of currency in real time are required. Such systems facilitate smooth international transactions and support the finance sector in dealing with diverse currencies. Thus, this study exploited various state-of-the-art deep learning models to classify Turkish banknotes. Figure 1 shows the proposed scheme in this study. The information from two distinct data sources is combined to create the dataset. In addition to separating the data into training and test sets, the system performs preprocessing tasks, such as resizing images and other similar tasks. Then, it trains seven technologically advanced models, including VGG16, VGG19, DenseNet121, DenseNet169, DenseNet201, MobileNet, and MobileNetV2, in a separate manner. The trained models were then validated with a test set to determine whether the proposed scheme is valid.

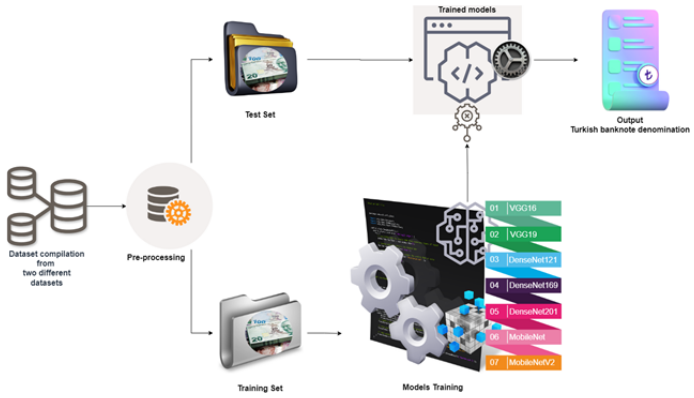

Figure 1. The workflow of the proposed system. The dataset was created by merging data from two different sources. The system performs pre-processing, such as image resizing, etc., and splits the data into training and test sets. Then, seven different customized state-of-the-art models are trained separately. The trained models were then tested on a test set to evaluate the validity of the proposed scheme.

2.1. Dataset

The dataset used in this study was created by merging two existing datasets from previous studies, namely (Baltaci, 2020) and (Sahin, 2018). resulting in a comprehensive collection of Turkish banknotes. The final dataset comprised a total of 6,901 images representing six Turkish Lira denominations: 5, 10, 20, 50, 100, and 200 TL. Each class is represented by a substantial number of images, providing a well-balanced dataset for training and testing deep learning models. The images feature banknotes under various conditions, including flat, angled, curved, and bent conditions, providing a challenging and realistic dataset that simulates real-world conditions in banknote handling. Figure 2 shows sample images of banknotes under various conditions.

Figure 2. The sample banknote images with various conditions (angle, curved, bent, and flat) were taken in different brightness environments. Turkish Banknotes are represented by 6 classes: 5, 10, 20, 50, 100, and 200 TL.

The diversity of the dataset is a significant strength because it includes images captured from different sides, angles, and orientations of the banknotes. This ensures that deep learning models trained on these data can generalize well to banknotes that may be presented under non-ideal conditions. The variations in the dataset—such as bends or curved notes—mimic how banknotes appear in everyday transactions, which is critical for real-world applicability. In addition, the representation of multiple denominations in the dataset allows for classification across a range of values, ensuring that models can distinguish between banknotes of different monetary values.

Moreover, the proposed dataset provides a valuable contribution to the field of currency classification, especially with the inclusion of challenging conditions such as different angles and bent banknotes. Such features increase the dataset's complexity, which makes it ideal for training deep learning models to be robust to variations in banknote presentation. This dataset could be useful for not only this particular study but also future research in the areas of currency recognition, counterfeit detection, and financial automation systems, where variability in banknote appearance poses a critical challenge.

2.2. Methodology

VGG16: The VGG16 is a deep convolutional neural network model comprising 16 weight layers (13 convolutional and 3 fully connected). The architecture follows a simple, uniform design in which convolutional layers are stacked with small 3x3 kernels and ReLU activation functions. Max pooling is applied after groups of convolutional layers to reduce the spatial dimensions. The network ends with three fully-connected layers and a softmax output. Despite its simplicity, VGG16 has high computational and memory costs due to its numerous parameters.

VGG19: VGG19 is an extension of VGG16 with 19 weight layers (16 convolutional and 3 fully connected). Like VGG16, it uses small 3×3 filters and ReLU activations in each layer but contains more convolutional layers, increasing the depth. The additional layers improve the capacity to learn complex features but further increase the computational requirements. VGG19 retains the same design philosophy, with max pooling applied after blocks of convolutions and fully connected layers at the end.

DenseNet121: DenseNet121 is a densely connected convolutional network in which each layer is directly connected to each other in a feedforward manner. Instead of learning redundant feature maps, DenseNet concatenates feature maps from all preceding layers, which improves network efficiency and allows for better feature reuse. The architecture consists of dense blocks separated by transition layers that downslope feature maps using convolution and pooling. DenseNet121 has 121 layers, with relatively fewer parameters than traditional networks like VGG, making it more computationally efficient.

DenseNet169: DenseNet169 follows the same architectural design as DenseNet121 but has 169 layers, thereby offering deeper feature extraction. The dense connectivity pattern helps mitigate the vanishing gradient problem; thus, deep networks can be trained more easily. The increased depth allows DenseNet169 to capture more complex features while maintaining higher efficiency in terms of parameter count and computational cost compared to similarly deep networks.

DenseNet201: DenseNet201 further extends the DenseNet architecture to 201 layers, providing deeper feature learning capabilities. Like its counterparts, DenseNet201 employs dense connections and transition layers between dense blocks. The deeper architecture allows for more complex hierarchical feature extraction, making it suitable for tasks requiring fine-grained classification, although there is a trade-off in increased computational load compared to smaller DenseNet models.

MobileNet: MobileNet is designed for mobile and embedded applications, focusing on efficiency and speed. This method uses depthwise separable convolutions, where a standard convolution is factorized into a depthwise convolution followed by a pointwise convolution, drastically reducing the number of parameters and computational cost. The proposed structure allows MobileNet to maintain high accuracy while being computationally lightweight, which makes it ideal for low-power devices and real-time applications.

MobileNetV2: MobileNetV2 improves upon MobileNet by introducing inverted residuals and linear bottlenecks. The inverted residuals use a shortcut connection between thin bottleneck layers. In contrast, the linear bottleneck layer ensures that the activation function does not destroy significant information during the bottleneck stage. MobileNetV2 retains the use of depthwise separable convolutions for computational efficiency; however, it improves feature representation, especially for complex tasks, by reducing information loss via the linear bottleneck structure. This resulted in better performance on resource-limited devices.

3. WORKING ENVIRONMENT AND EXPERIMENTAL RESULTS

The working environment for this study was implemented using Python as the primary programing language, and the Keras library was employed for building and training the deep learning models. Keras, a high-level API, was selected for its ease of use and integration with TensorFlow, making it ideal for conducting experiments with various deep learning models. The hardware setup included a system with GPU acceleration to facilitate faster model training and testing, allowing for efficient experimentation across multiple models, optimizers, and image sizes (Keras Team, 2023a). The optimization functions are SGD, RMSprop, Adam, Adadelta, Adagrad, Adamax, Nadam, and Ftrl (Keras Team 2023b)(Filter 2022). The development and testing were conducted in a controlled environment to ensure the reproducibility of results and to measure each model's performance accurately.

The dataset used in the experiments was created by merging two existing datasets, which resulted in 6901 images of Turkish banknotes. These images were classified into six categories: 5, 10, 20, 50, 100, and 200 TL. The dataset was highly diverse and contained images of banknotes in various orientations and conditions, such as flat, angled, curved, and bent. The images were preprocessed by resizing them to four different sizes (50×50, 100×100, 150×150, and 200×200), which allowed the models to be trained and tested on varying image dimensions. The dataset was split into training and testing sets with ratios of 80% and 20%, respectively. Table 1 presents the dataset details. This preprocessing step ensured that the models could handle different image resolutions and identified the optimal size for maximum classification accuracy.

Table 1. *Dataset description, number of samples in dataset, and train-test split for each class label in dataset*

Class	Train set	Test set	Total
5 TL	920	230	1150
10 TL	920	230	1150
20 TL	920	231	1151
50 TL	920	230	1150
100 TL	920	230	1150
200 TL	920	230	1150
		Total	**6901**

The experiments used several pretrained deep learning models, including VGG16, VGG19, DenseNet121, DenseNet169, DenseNet201, MobileNet, and MobileNetV2. Each model was fine-tuned using various optimizers, such as SGD, RMSprop, Adam, Adadelta, Adagrad, Adamax, Nadam, and Ftrl. These optimizers were applied with their default parameter values to assess their impact on model performance. Additionally, different image sizes were tested to determine the optimal resolution for efficient model training. The classification task involved training CNNs with a combination of predefined and custom layers, including two hidden layers of 1024 neurons with ReLU activation and a final output layer with a SoftMax activation function that matches the number of banknote classes.

The training process was conducted for 100 epochs with an early stopping mechanism based on validation accuracy to avoid overfitting. If validation accuracy did not improve after a certain number of epochs, training was stopped to obtain the best model. The validation accuracy served as a control point, and only the best performing model was retained in each experiment. Various metrics can be used to measure the success of the classification process. This study considered accuracy as an evaluation measure, which is the proportion of the total data that is in the correct class. It is the most widely used metric, especially when the dataset is balanced (Foody, 2023). In addition, this study measured the confusion matrix against each model exploited. Accuracy was the primary evaluation metric, and it was monitored throughout the training process. Accuracy and loss values for both the training and validation datasets were plotted in each experiment to assess the performance of the models at each epoch.

DenseNet201 achieved the highest accuracy of 98.84% with an image size of 200×200 and the SGD optimizer, which was completed in 12 epochs. The reason for keeping the epoch value at 12 is that overlearning occurred at this value. The model's efficiency increased significantly after the third epoch and was maintained after the eighth epoch. The performance of DenseNet201 decreased slightly with smaller image sizes; however, it still performed well with an accuracy of 97.54% for both 150×150 and 100×100 image sizes using the Adamax optimizer. However, for the smallest image size of 50, the performance decreased to 91.82% with the Adam optimizer, indicating that the model benefits from larger image inputs that can extract more detailed features. Table 2 lists the performances of the DenseNet201 model when trained on varying image sizes and optimizers. Figure 3 shows the accuracy and loss curves of the DenseNet201 model, and Figure 4 shows the confusion matrix for each.

Table 2. *Accuracy performance analysis of DenseNet201 trained on varying input sizes and optimizers*

Image Size	Optimizer	Accuracy Rate	Epoch number
200×200	SGD	0.9884	12
150×150	Adamax	0.9754	16
100×100	Adamax	0.9754	14
50×50	Adam	0.9182	17

DenseNet169 also performed well, with its best accuracy of 98.62%, which was achieved using an image size of 200×200 and the SGD optimizer, completing in 17 epochs. With a slightly smaller image size of 150×150, it maintained strong performance, achieving 97.39% accuracy with the RMSprop optimizer. Like DenseNet201, its accuracy decreased with smaller image sizes, achieving 97.47% at 100×100 and 90.15% at 50×50, indicating that DenseNet169, like its deeper counterpart, benefits from larger images and performs better with optimizers like SGD and RMSprop. Table 3 lists the performances of the DenseNet169 model when trained on varying image sizes and optimizers. Figure 5 shows the accuracy and loss curves of the DenseNet169 model, and Figure 6 shows the confusion matrix for each.

Table 3. *Accuracy performance analysis of DenseNet169 trained on varying input sizes and optimizers*

Image Size	Optimizer	Accuracy Rate	Epoch number
200×200	SGD	0.9862	17
150×150	RMSprop	0.9739	19
100×100	Adamax	0.9747	11
50×50	RMSprop	0.9015	20

DenseNet121 reached its highest accuracy of 97.68% with an image size of 150×150 and the Adamax optimizer, which completed in 11 epochs. For an image size of 200×200, it performed slightly lower with 97.38% using the RMSprop optimizer; however, this performance was still strong. For smaller image sizes of 100×100 and 50×50, the model's accuracy decreased to 96.31% and 90.30%, respectively, confirming that, like other DenseNet models, DenseNet121 performs better with larger image sizes but can still maintain decent performance on mid-sized images. Table 4 lists the performances of the DenseNet121 model when trained on varying image sizes and optimizers. Figure 7 shows the accuracy and loss curves of the DenseNet121 model, and Figure 8 shows the confusion matrix for each.

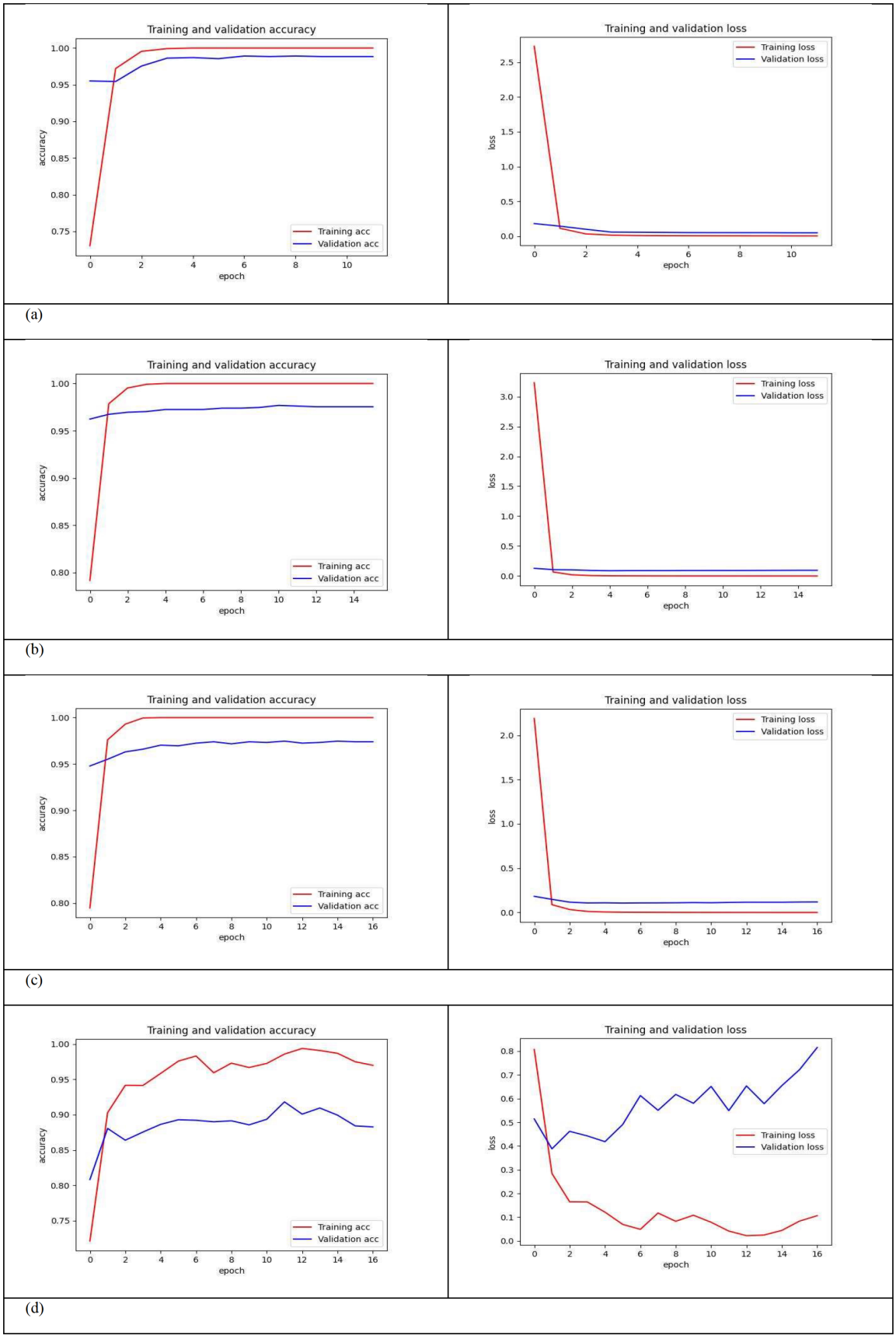

Figure 3. Accuracy and loss curves of DenseNet201 when trained over varying image sizes with several optimizers. Trained with (a) image size of 200×200 and SGD optimizer, (b) image size of 150×150 and Adamax optimizer, (c) image size of 100×100 and Adamax optimizer, (d) image size of 50×50 and Adam optimizer.

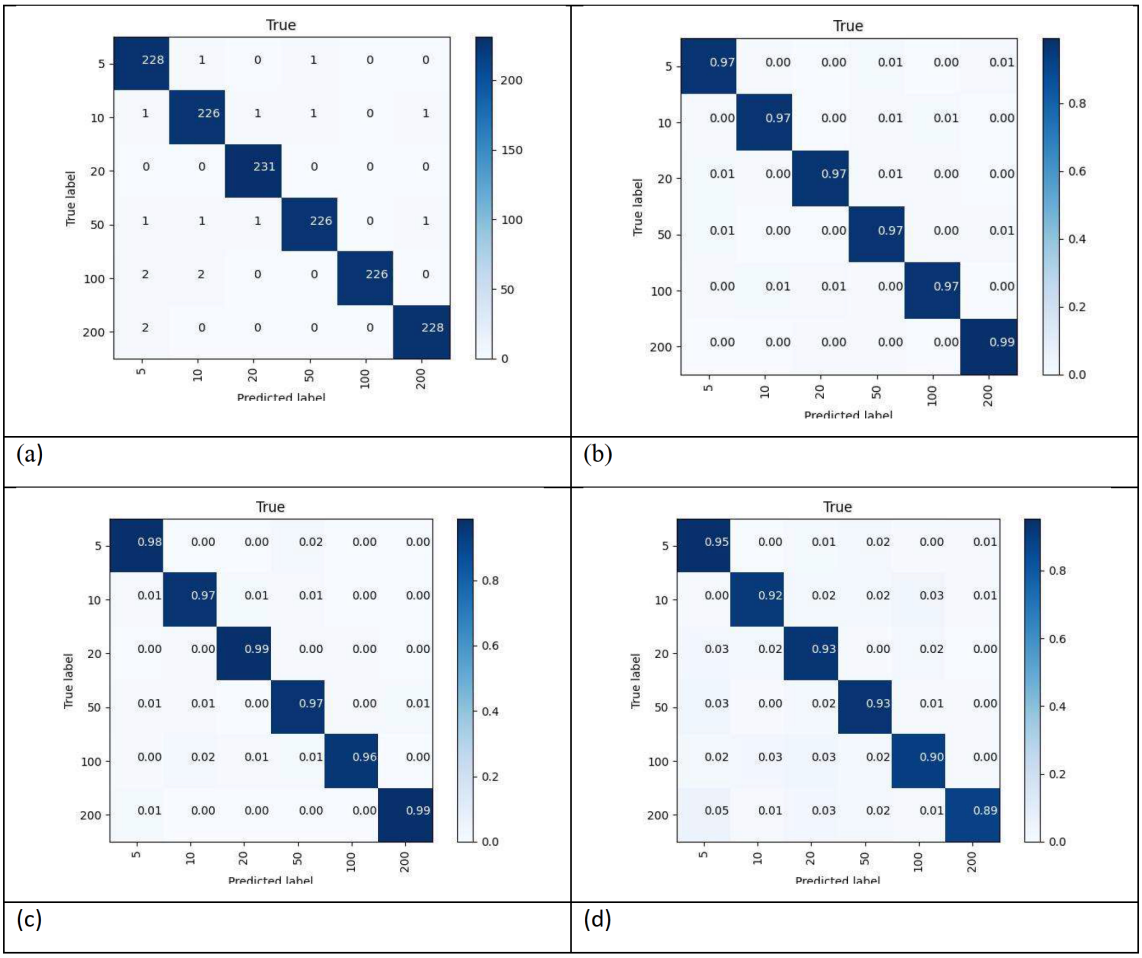

Figure 4. The confusion matrix of DenseNet201 when trained on images of various sizes with several optimizers. Trained with (a) image size of 200×200 and SGD optimizer, (b) image size of 150×150 and Adamax optimizer, (c) image size of 100×100 and Adamax optimizer, (d) image size of 50×50 and Adam optimizer.

Table 4. *Accuracy performance analysis of DenseNet121 trained on varying input sizes and optimizers*

Image Size	Optimizer	Accuracy Rate	Epoch number
200×200	RMSprop	0.9738	22
150×150	Adamax	0.9768	11
100×100	Adamax	0.9631	17
50×50	RMSprop	0.9030	23

MobileNet achieved its highest accuracy of 97.47% with an image size of 200×200 and the Adamax optimizer, completing in 20 epochs. For smaller image sizes of 150×150, 100×100, and 50×50, the model's performance gradually decreased to 96.81%, 95.51%, and a very low 19.55%, respectively, indicating that MobileNet struggles significantly with very small image sizes. MobileNet's performance suggests it is a good option for efficient classification with larger image sizes; however, it becomes less reliable when image resolution is reduced. While conducting the experiments, we also tried various other image sizes, such as 128×128 and 96×96. Table 5 lists the performance obtained by the MobileNet model when trained on varying image sizes and optimizers. Figure 9 shows the accuracy and loss curves of the MobileNet model, and Figure 10 shows the confusion matrix for each.

MobileNetV2 demonstrated moderate performance compared to the other models, with the highest accuracy of 95.51% at an image size of 150 using the Adamax optimizer. At the largest image size of 200, this value reached 95.37% with the Ftrl optimizer. However, with smaller image sizes of 100 and 50, MobileNetV2's performance decreased to 93.63% and 17.45%, respectively. This suggests that while MobileNetV2 can achieve decent performance with mid-sized images, it struggles significantly with negligible images, which is similar to MobileNet. Table 6 lists the performance obtained by the MobileNetV2 model when trained on varying image sizes and optimizers. Figure 11 shows the accuracy and loss curves for the MobileNetV2 model, and Figure 12 shows the confusion matrix for each.

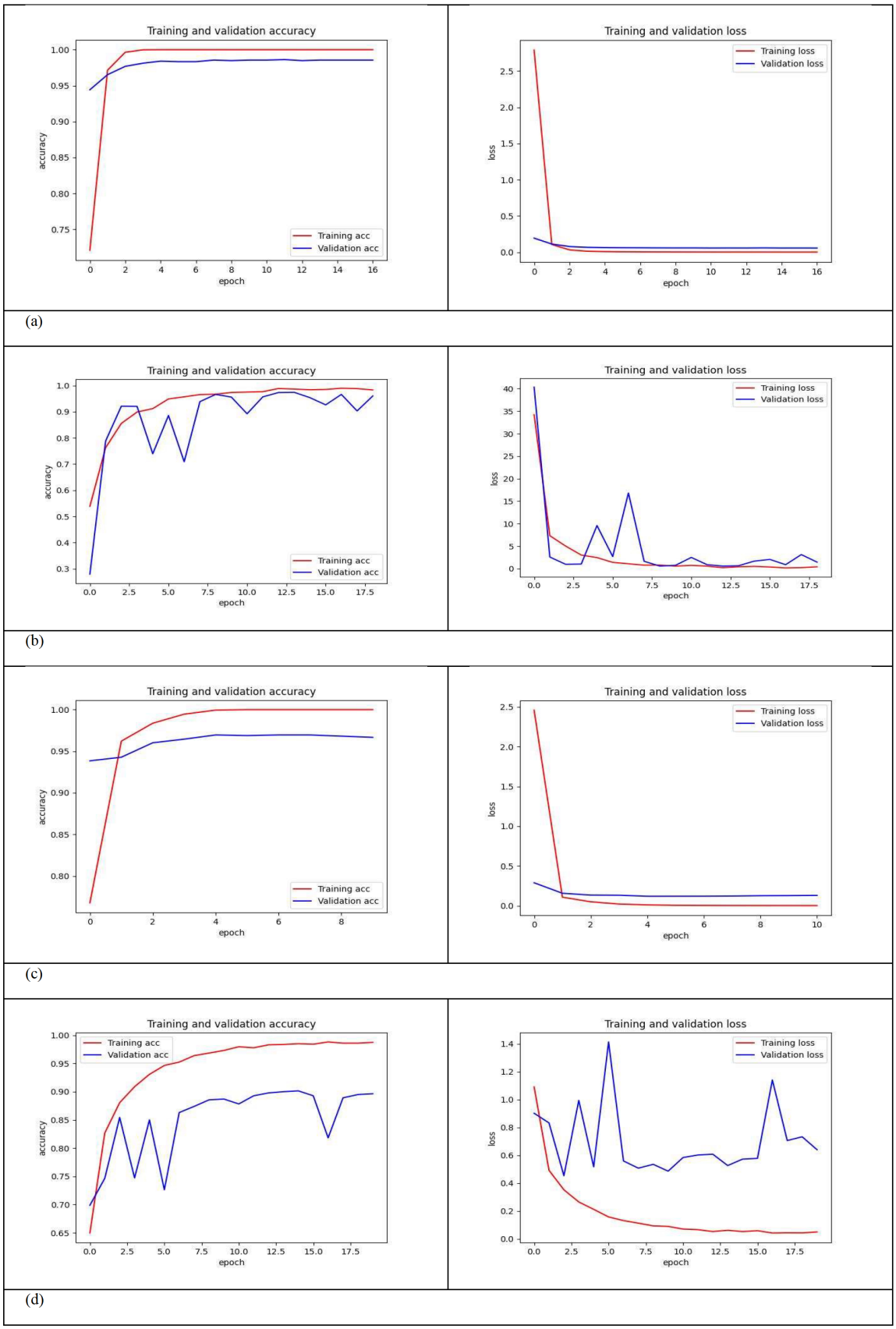

Figure 5. Accuracy and loss curves of DenseNet169 when trained over varying image sizes with several optimizers. Trained with (a) image size of 200×200 and SGD optimizer, (b) image size of 150×150 and RMSprop optimizer, (c) image size of 100×100 and Adamax optimizer, (d) image size of 50×50 and RMSprop optimizer.

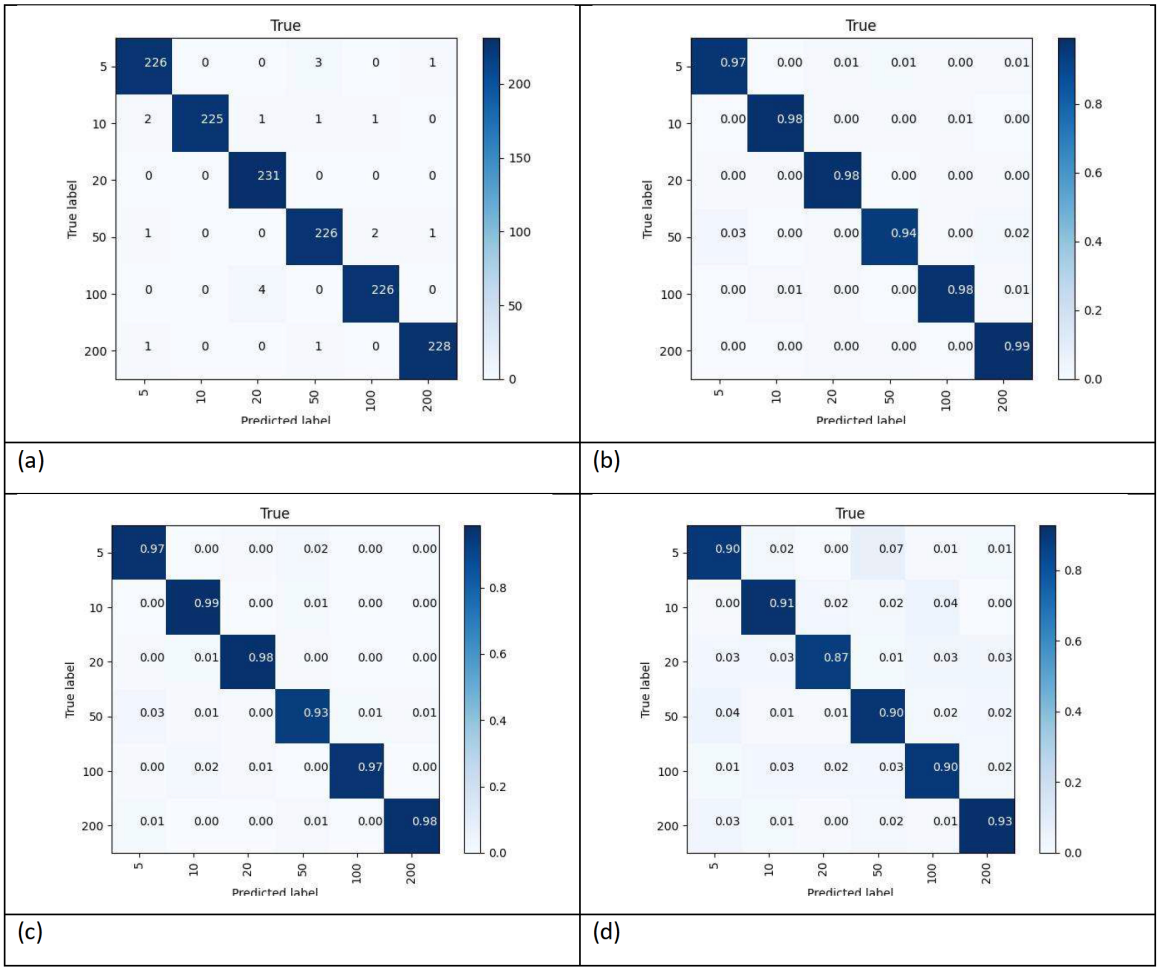

Figure 6. The confusion matrix of DenseNet169 when trained on images of various sizes with several optimizers. Trained with (a) image size of 200×200 and SGD optimizer, (b) image size of 150×150 and RMSprop optimizer, (c) image size of 100×100 and Adamax optimizer, (d) image size of 50×50 and RMSprop optimizer.

Table 5. *Accuracy performance analysis of MobileNet trained on varying input sizes and optimizers*

Image Size	Optimizer	Accuracy Rate	Epoch number
200×200	Adamax	0.9747	20
150×150	Adamax	0.9681	15
100×100	Adamax	0.9551	17
50×50	Adamax	0.1955	26

Table 6. *Accuracy performance analysis of MobileNetV2 trained on varying input sizes and optimizers*

Image Size	Optimizer	Accuracy Rate	Epoch number
200×200	Ftrl	0.9537	6
150×150	Adamax	0.9551	18
100×100	SGD	0.9363	22
50×50	Nadam	0.1745	15

VGG16 achieved the highest accuracy of 96.69% at an image size of 150×150 using the RMSprop optimizer. With an image size of 200×200, it performed similarly, achieving 96.38% with the same optimizer. For smaller image sizes of 100×100 and 50×50, the model accuracy decreased to 95.37% and 90.15%, respectively. VGG16 performed reasonably well across different image sizes, although its performance was slightly lower compared to the DenseNet models, indicating that it may not be as efficient in extracting intricate features from banknote images. Table 7 lists the

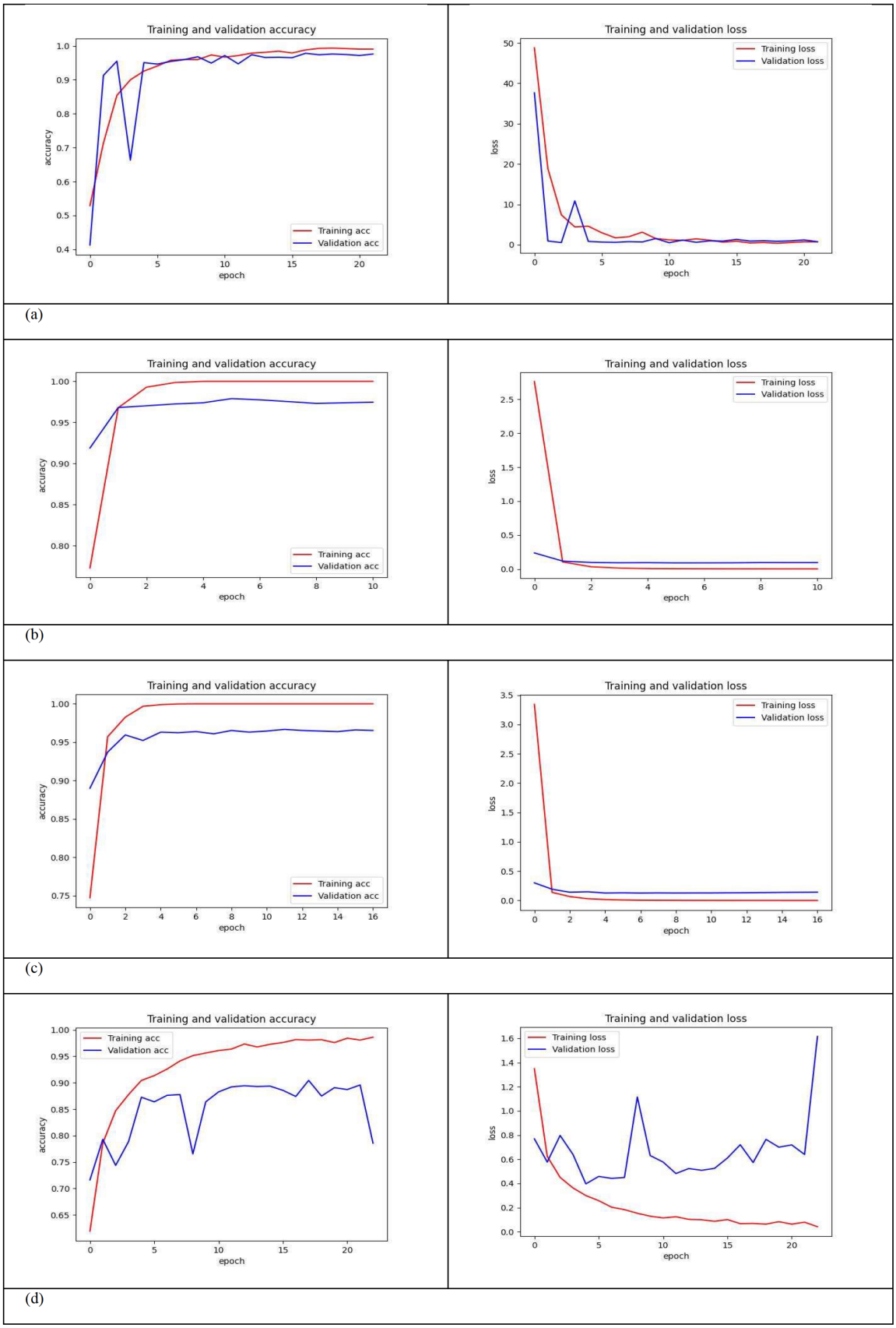

Figure 7. Accuracy and loss curves of DenseNet121 when trained over varying image sizes with several optimizers. Trained with (a) image size of 200×200 and RMSprop optimizer, (b) image size of 150×150 and Adamax optimizer, (c) image size of 100×100 and Adamax optimizer, (d) image size of 50×50 and RMSprop optimizer.

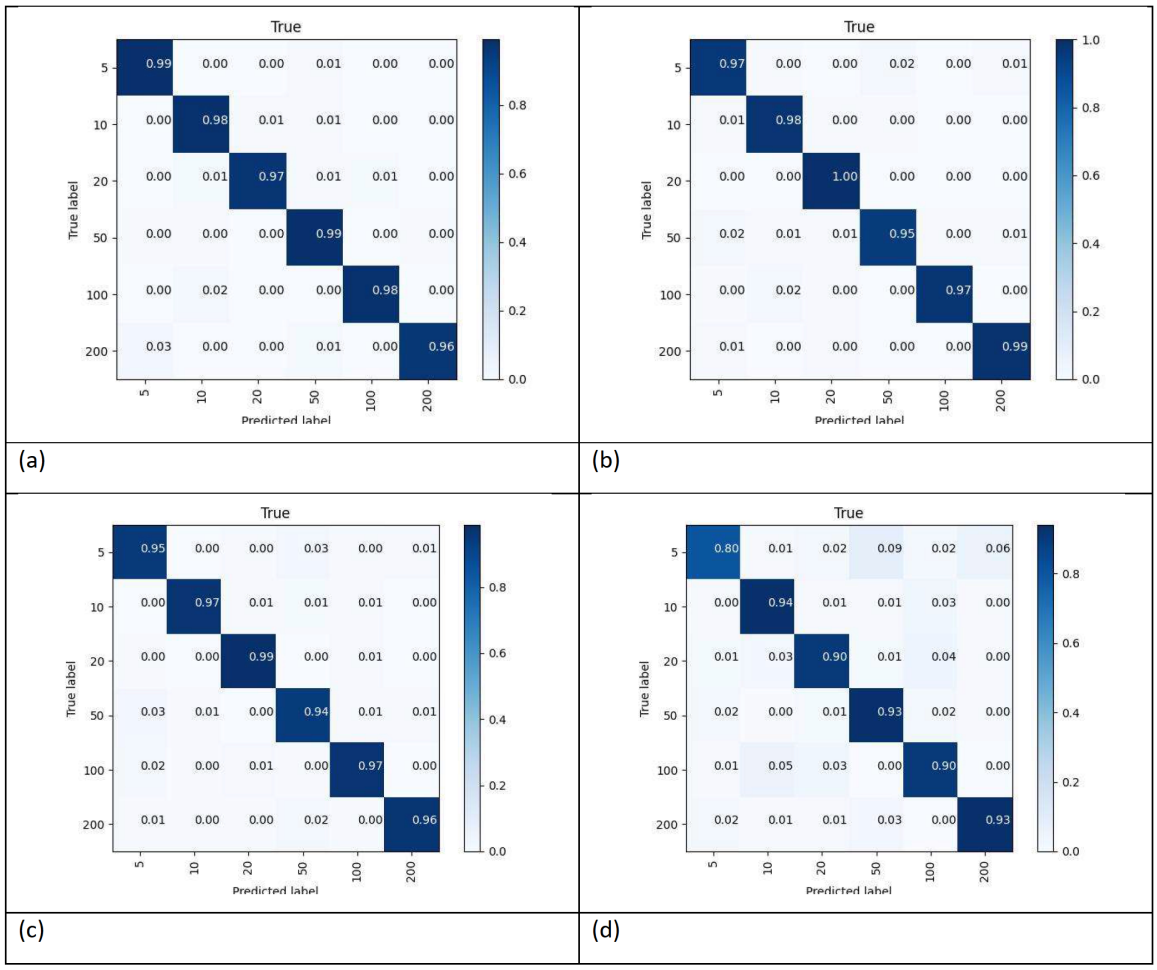

Figure 8. The confusion matrix of DenseNet121 when trained on images of various sizes with several optimizers. Trained with (a) image size of 200×200 and RMSprop optimizer, (b) image size of 150×150 and Adamax optimizer, (c) image size of 100×100 and Adamax optimizer, (d) image size of 50×50 and RMSprop optimizer.

performances of the VGG19 model when trained on varying image sizes and optimizers. Figure 13 shows the accuracy and loss curves for the VGG19 model, and Figure 14 shows the confusion matrix for each.

Table 7. *Analysis of accuracy of VGG16 trained over varying input sizes and optimizers*

Image Size	Optimizer	Accuracy Rate	Epoch number
200×200	RMSprop	0.9638	24
150×150	RMSprop	0.9669	20
100×100	Adamax	0.9537	13
50×50	RMSprop	0.9015	24

VGG19 achieved its highest accuracy of 96.89% with an image size of 200×200 using the Adamax optimizer, and it completed in 21 epochs. For smaller image sizes of 150×150 and 100×100, it achieved 96.02% and 94.57%, respectively. At the smallest image size of 50×50, the performance dropped to 89.28%, suggesting that VGG19 also benefits from larger image sizes but struggles more than VGG16 with smaller images. Table 8 lists the performance obtained by the MobileNetV2 model when trained on varying image sizes and optimizers. Figure 15 shows the accuracy and loss curves for the VGG19 model, and Figure 16 shows the confusion matrix for each.

The experimental results revealed that image size plays a significant role in determining model efficiency. For smaller images (50×50), no model or optimizer achieved satisfactory results, indicating that the image resolution was too low for the models to extract meaningful features. Some models performed reasonably well for image sizes of 100×100 and 150×150, especially when faster training times were prioritized. However, the best performance was observed with an image size of 200×200, particularly when combined with the DenseNet201 model and the SGD optimizer. This

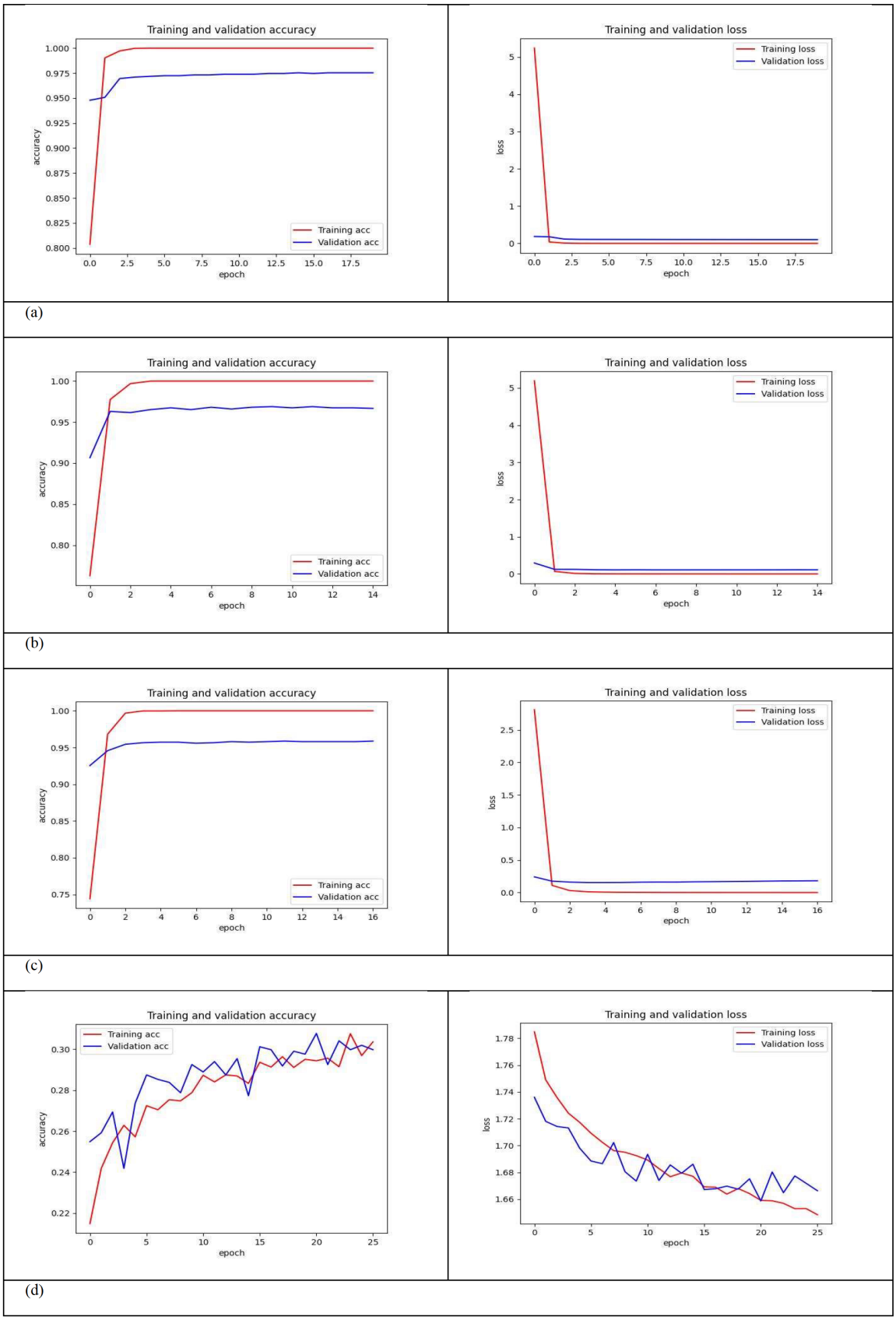

Figure 9. Accuracy and loss curves of MobileNet when trained over varying image sizes with several optimizers. Trained with (a) image size of 200×200 and Adamax optimizer, (b) image size of 150×150 and Adamax optimizer, (c) image size of 100×100 and Adamax optimizer, (d) image size of 50×50 and Adamax optimizer.

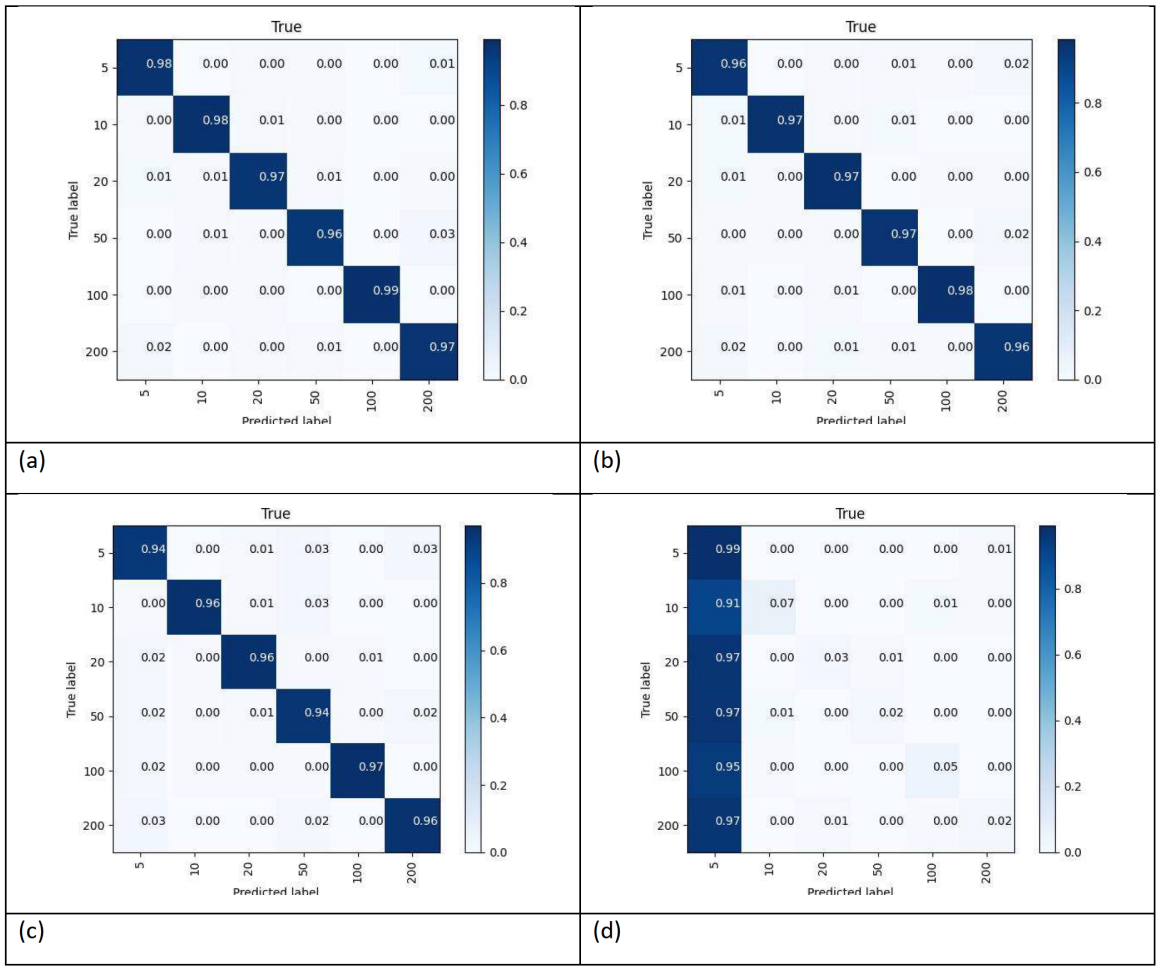

Figure 10. The confusion matrix of MobileNet when trained on images of various sizes with several optimizers. Trained with (a) image size of 200×200 and Adamax optimizer, (b) image size of 150×150 and Adamax optimizer, (c) image size of 100×100 and Adamax optimizer, (d) image size of 50×50 and Adamax optimizer.

Table 8. *Analysis of accuracy of VGG19 trained over varying input sizes and optimizers*

Image Size	Optimizer	Accuracy Rate	Epoch number
200×200	Adamax	0.9689	21
150×150	Adamax	0.9602	16
100×100	Adamax	0.9457	13
50×50	Adamax	0.8928	28

combination achieved an impressive accuracy of 98.84%, which improved efficiency significantly after the third epoch and stabilized by the eighth epoch, demonstrating that larger image sizes provide models with more detailed features for classification.

4. DISCUSSION

In this section, the results of the tests performed in the test environment are analyzed for the three variables studied, two of which are fixed variables. The data in the graphs were sorted from smallest to largest according to the appropriate cases, and the value analyzed was the accuracy value.

The image size parameter is a parameter tested in the test environment and is the first step of the feature detection process in the learning process. In the test environment, the width and height of the images were set to be the same. Although it seems to be an advantage to choose a high dimension, it can be observed that after a certain value, it is not useful and sometimes decreases the performance value.

The VGG, Mobile, and DenseNet models were selected as the types of transfer learning to be tested. Mobile

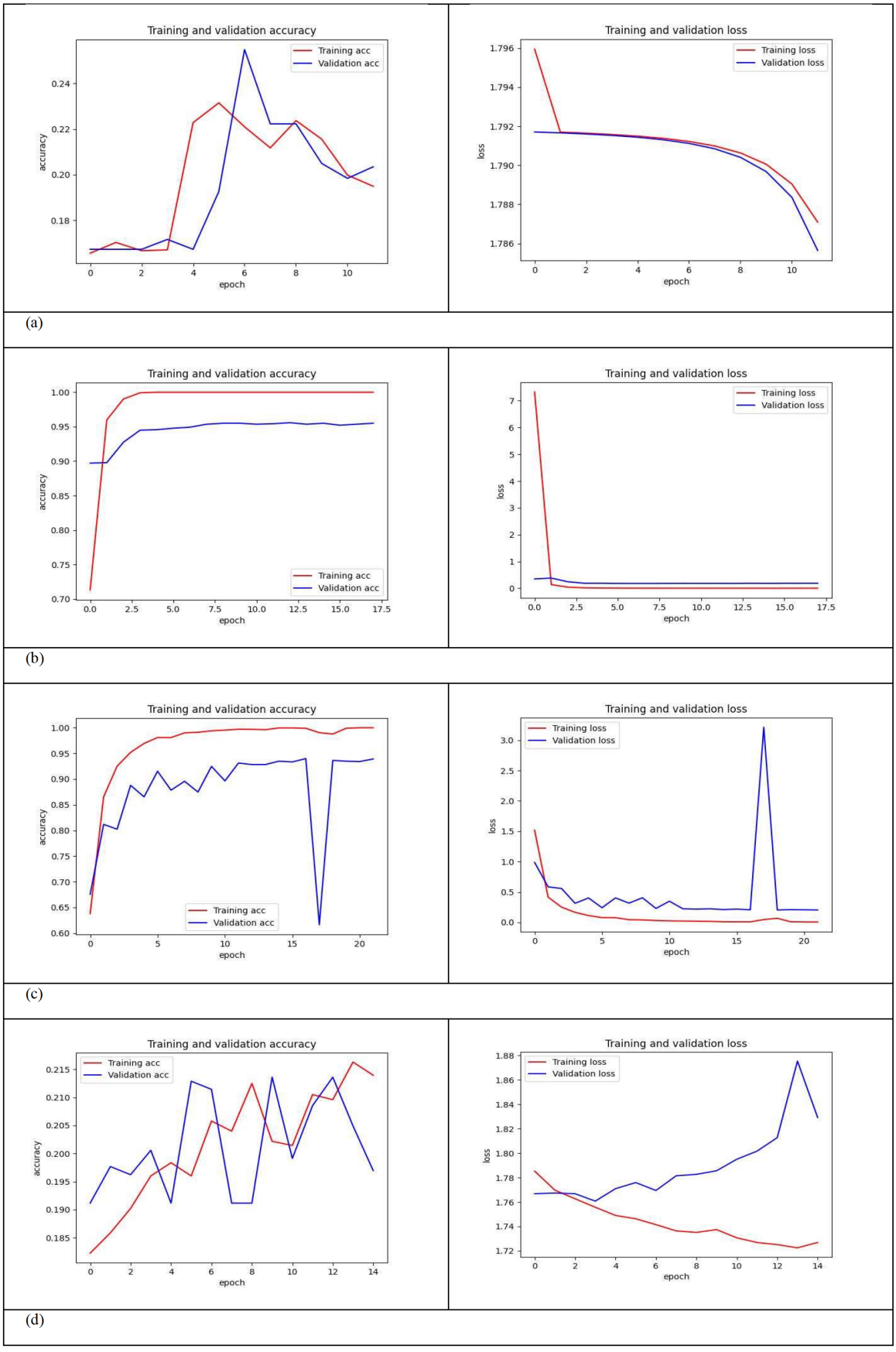

Figure 11. Accuracy and loss curves of MobileNetV2 when trained over varying image sizes with several optimizers. Trained with (a) image size of 200×200 and Ftrl optimizer, (b) image size of 150×150 and Adamax optimizer, (c) image size of 100×100 and SGD optimizer, (d) image size of 50×50 and Nadam optimizer.

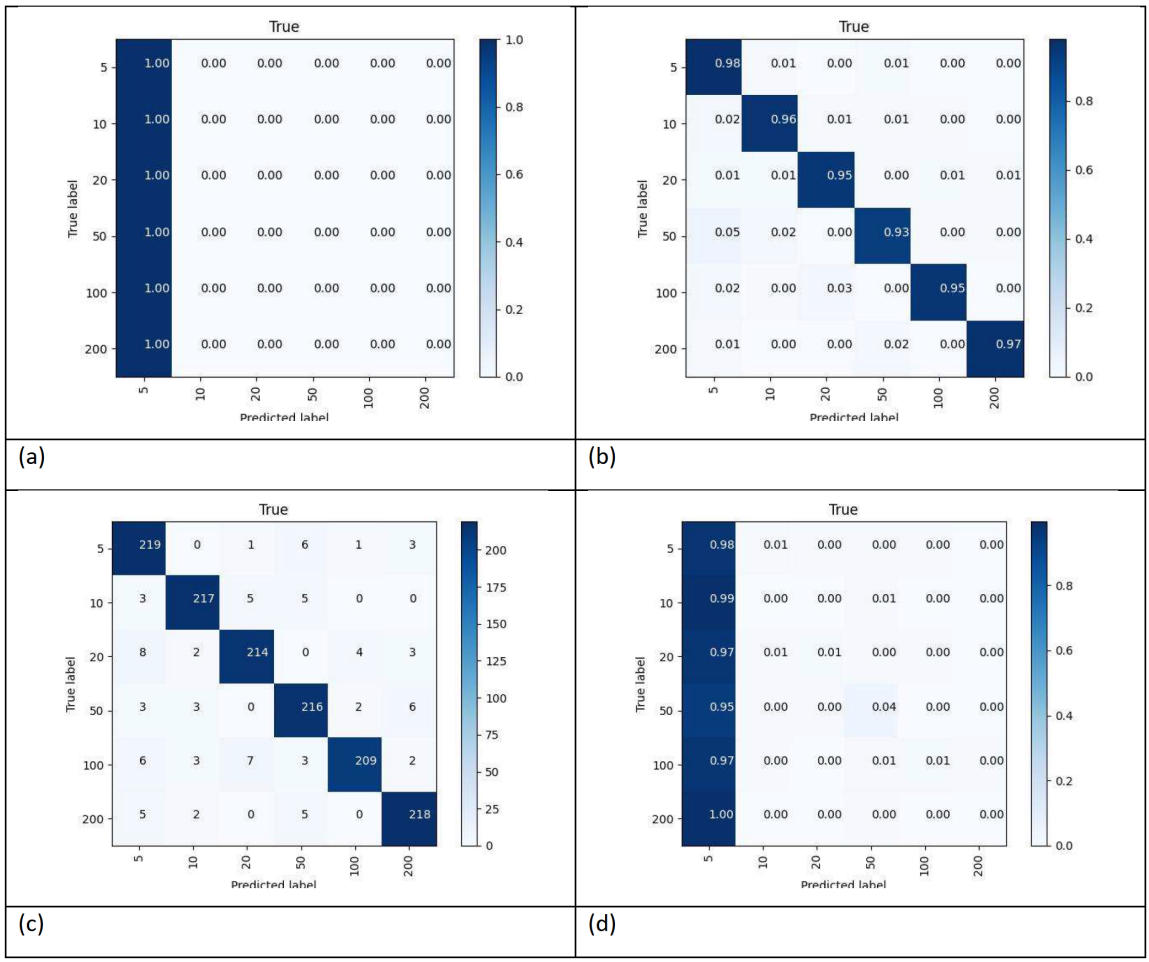

Figure 12. The confusion matrix of MobileNetV2 when trained on images of various sizes with several optimizers. Trained with (a) image size of 200×200 and Ftrl optimizer, (b) image size of 150×150 and Adamax optimizer, (c) image size of 100×100 and SGD optimizer, (d) image size of 50×50 and Nadam optimizer.

models were selected because they are fast and have fewer layers, VGG models have more layers, and Dense models have a balanced number of layers but run slower than other models. In this study, another tested parameter was the optimizer function. These functions affect learning speed and weight. For this reason, they also affect the performance measurement value.

From Figure 17, it can be observed that there is no efficiency with any transfer model or optimizer function when images with a width and height of 50 are used in the test environment. For the other image sizes, low efficiency was observed in some cases, but high efficiency was observed in most cases. A size of 100 is good if the training process must be rapid, and a size of 200 is good if performance is important. Increasing the size is not an important parameter alone; however, it is advantageous when used in conjunction with other parameters in the test environment.

In Figure 18, the effect of the examined model on the performance accuracy value was not significantly different from that of the appropriate parameters. The most efficient models were DenseNet201 and DenseNet121, and the least efficient models were MobileNetV2. It can be seen that the selection of the appropriate model is not the only condition that provides sufficient efficiency; the appropriate optimizer function and appropriate image size should also be determined. It can be observed that appropriate accuracy values can occur under appropriate conditions for all models.

Figure 19 shows that Ftrl is the least efficient optimizer function. With the other functions, efficient results can be obtained in an appropriate test environment. The efficiency criterion can be reached when the parameters are tested using various combinations of parameters that do not have appropriate values for the parameters alone for efficient operation in the test environment.

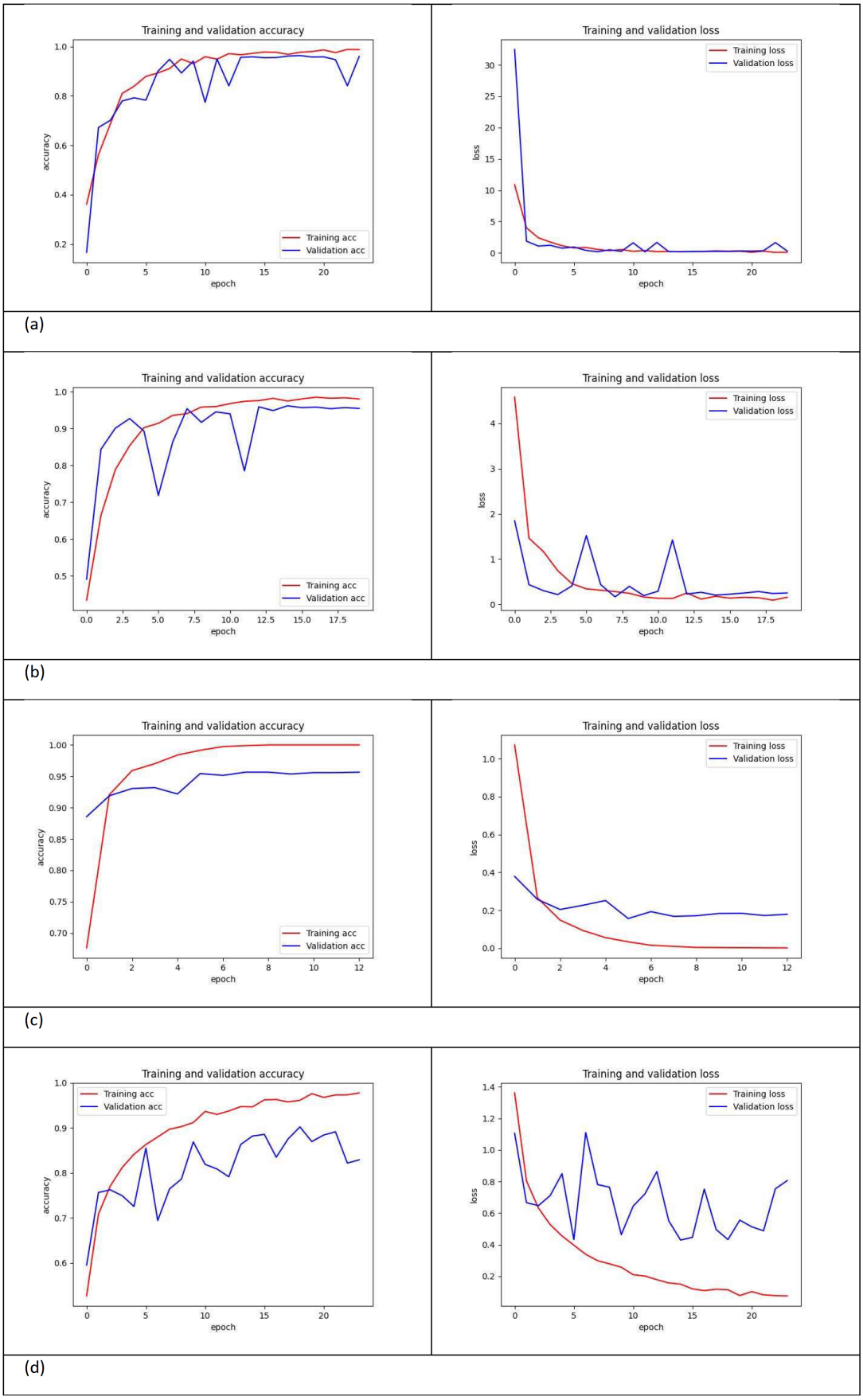

Figure 13. Accuracy and loss curves of VGG16 when training for various image sizes with several optimizers. Trained with (a) image size of 200×200 and RMSprop optimizer, (b) image size of 150×150 and RMSprop optimizer, (c) image size of 100×100 and Adamax optimizer, (d) image size of 50×50 and RMSprop optimizer.

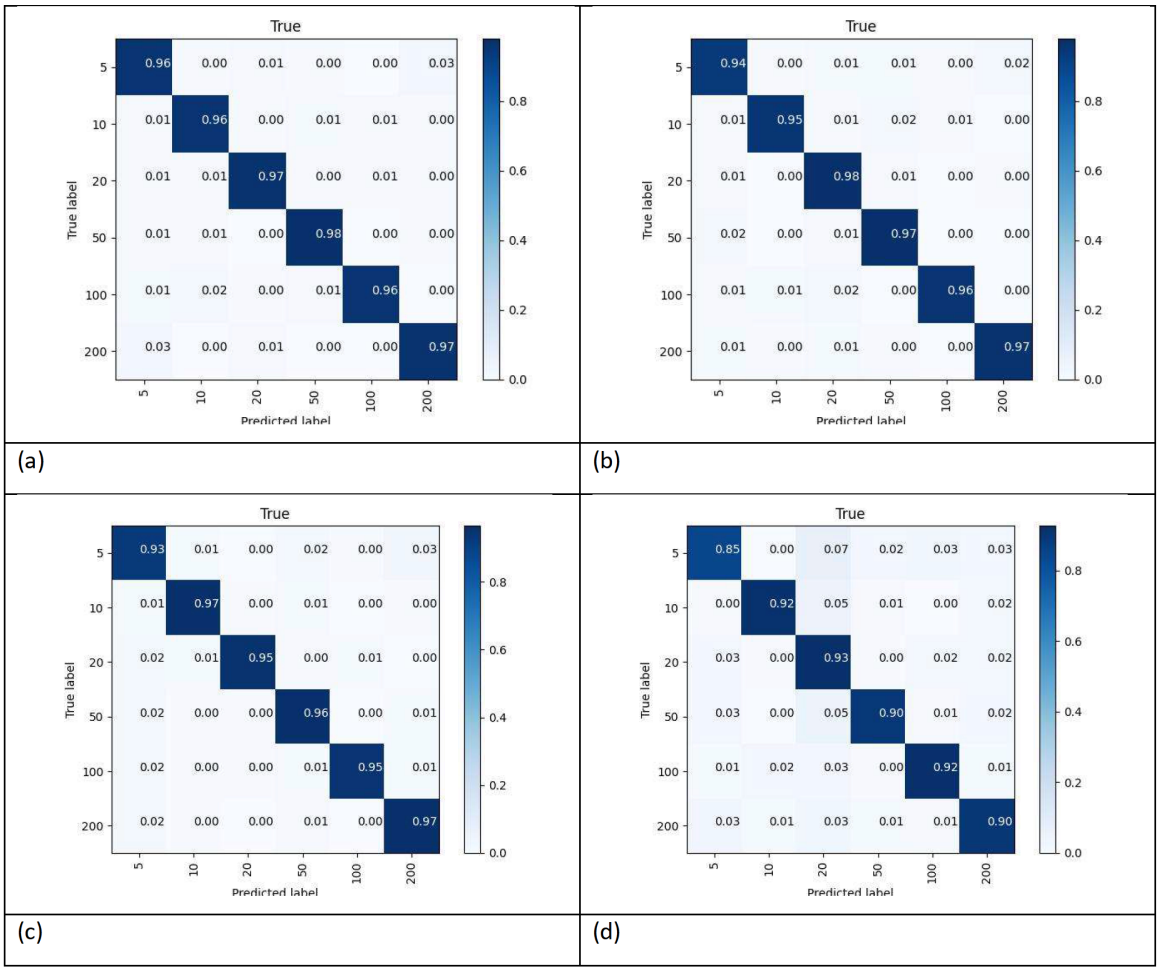

Figure 14. Accuracy and loss curves of VGG16 when training for various image sizes with several optimizers. Trained with (a) image size of 200×200 and RMSprop optimizer, (b) image size of 150×150 and RMSprop optimizer, (c) image size of 100×100 and Adamax optimizer, (d) image size of 50×50 and RMSprop optimizer.

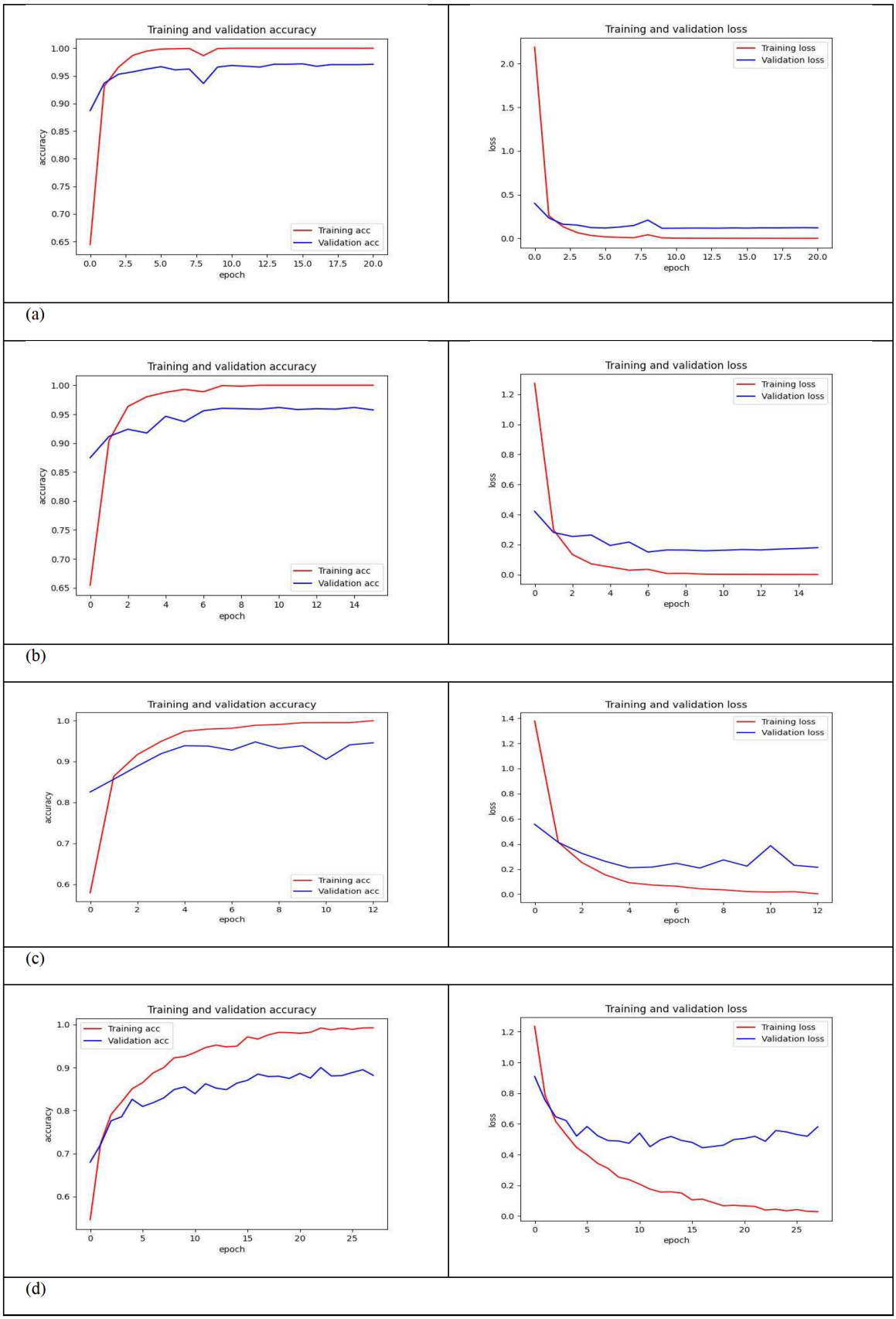

Figure 15. Accuracy and loss curves of VGG19 after training on various image sizes with several optimizers. Trained with (a) image size of 200×200 and Adamax optimizer, (b) image size of 150×150 and Adamax optimizer, (c) image size of 100×100 and Adamax optimizer, (d) image size of 50×50 and Adamax optimizer.

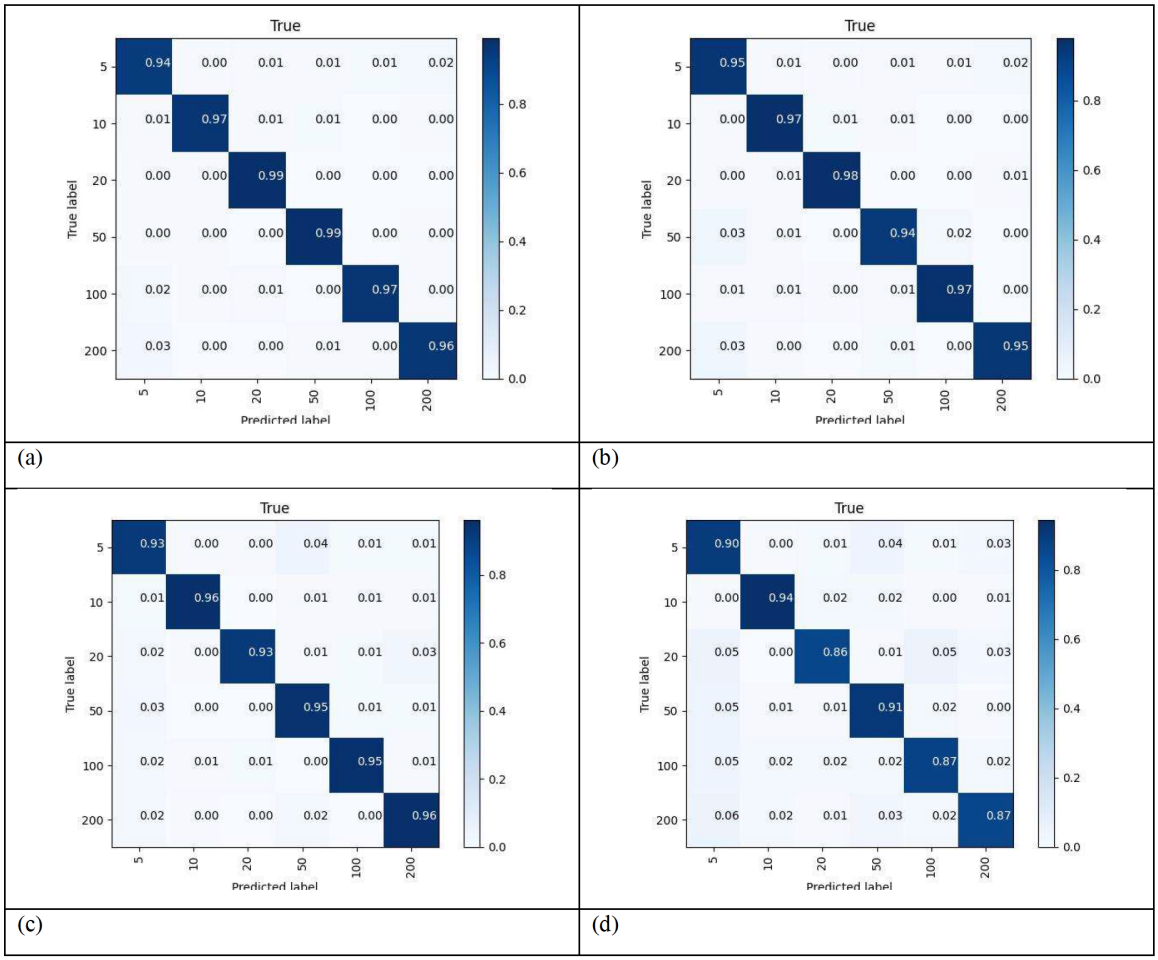

Figure 16. The confusion matrix of VGG19 when training for various image sizes with several optimizers. Trained with (a) image size of 200×200 and Adamax optimizer, (b) image size of 150×150 and Adamax optimizer, (c) image size of 100×100 and Adamax optimizer, (d) image size of 50×50 and Adamax optimizer.

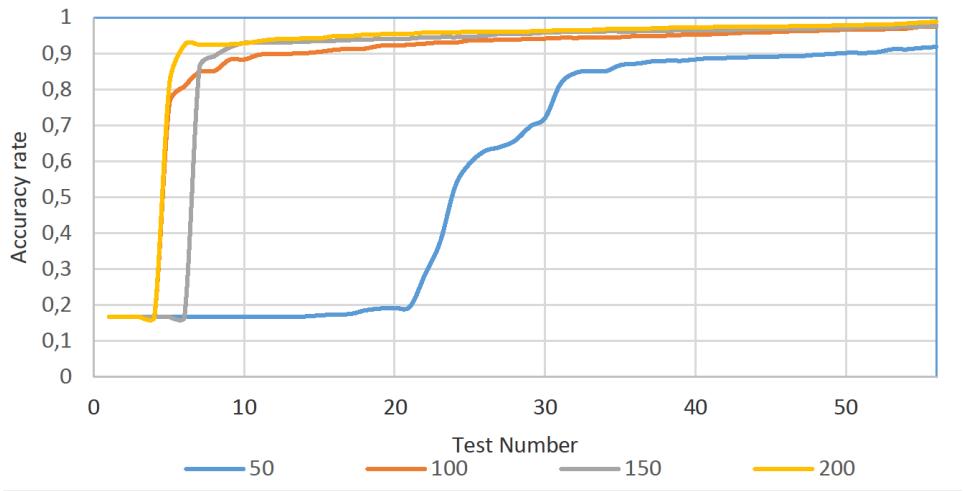

Figure 17. The image size and test performance relationship.

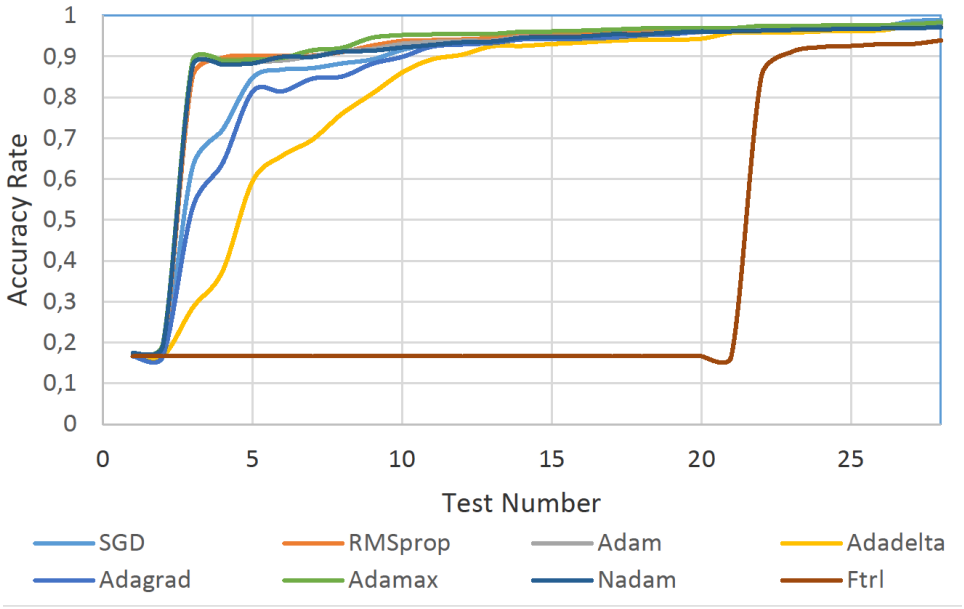

Figure 18. The accuracy comparison relationship between transfer learning models with respect to the test set.

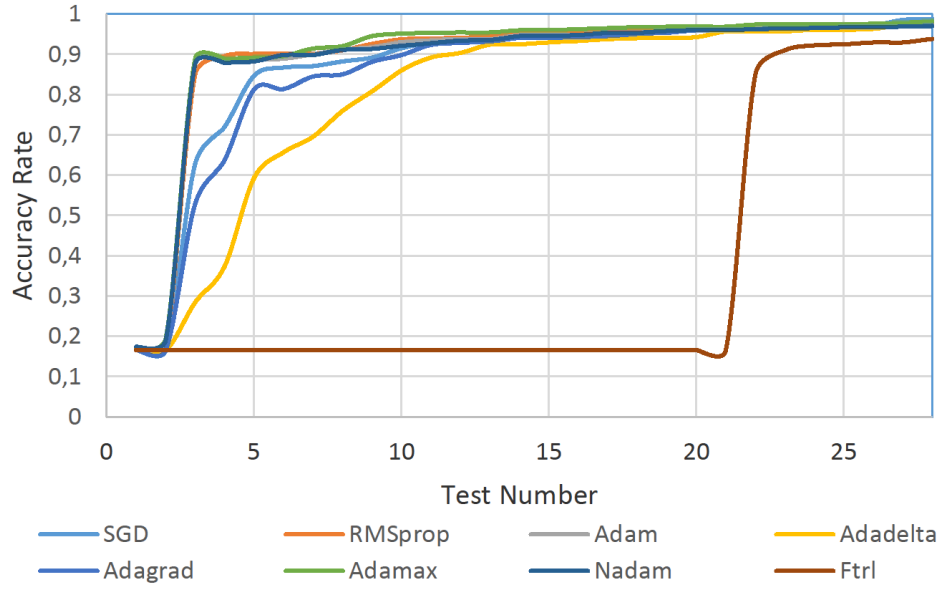

Figure 19. Figure 19 shows that Ftrl is the least efficient optimizer function. With the other functions, efficient results can be obtained in an appropriate test environment. The efficiency criterion can be reached when the parameters are tested using various combinations of parameters that do not have appropriate values for the parameters alone for efficient operation in the test environment.

5. CONCLUSION

With advancements in artificial intelligence and deep learning, research into banknote classification has provided new opportunities to realize more accurate, reliable, and scalable systems. These systems can leverage new technologies such as edge computing and Internet of Things, to improve the efficiency of real-time recognition systems. Efficient real-time classification ensures faster transactions and enhances user experience. This study investigated the performance of various deep learning models in the classification of Turkish banknotes using a comprehensive dataset of 6901 images across six denominations (5 TL, 10 TL, 20 TL, 50 TL, 100 TL, and 200 TL). The dataset includes banknotes under different conditions such as flat, angled, and bent conditions, providing a realistic testing environment. The study evaluates the performance of pre-trained models, including VGG16, VGG19, DenseNet121, DenseNet169, DenseNet201, MobileNet, and MobileNetV2, by varying key parameters, such as image size (50×50, 100×100, 150×150, 200×200) and optimizers (SGD, RMSprop, Adamax, Adam, and others). The best results were obtained using the DenseNet201 model, which achieved an accuracy of 98.84% with an image size of 200 and the SGD optimizer in just 12 epochs. DenseNet169 also performed well, achieving 98.62% accuracy under similar conditions. The study highlights the importance of selecting appropriate image sizes and optimizers because smaller image sizes significantly reduced model performance, particularly for MobileNet and MobileNetV2, which showed poor results with images sized 50×50. In addition, models using the Ftrl optimizer consistently exhibited lower performance. These findings demonstrate the effectiveness of DenseNet models for banknote classification, especially when combined with larger image sizes and optimizers like SGD. The results provide valuable insights into developing robust and efficient automated currency recognition systems, especially for real-time applications in financial sector.

In future work, it is planned to conduct tests using banknotes from different countries. Another field of study is to work with a test environment in which the effect of optimizing functions as variables can be better observed.

Peer Review: Externally peer-reviewed.

Conflict of Interest: Authors declared no conflict of interest.

Grant Support: The authors received no specific funding for this work.

ORCID IDs of the authors

Mirsat Yeşiltepe 0000-0003-4433-5606
Harun Elkıran 0000-0002-5834-6210
Jawad Rasheed 0000-0003-3761-1641

REFERENCES

Baek, S., Choi, E., Baek, Y., & Lee, C. (2018). Detection of counterfeit banknotes using multispectral images. *Digital Signal Processing, 78*, 294–304. https://doi.org/10.1016/j.dsp.2018.03.015

Baltaci, F. (2020). Turkish lira banknote dataset. Retrieved January 2, 2024, from https://www.kaggle.com/datasets/baltacifatih/turkish-lira-banknote-dataset

Baykal, G., Demir, U., Shyti, I., & Unal, G. (2018). Turkish lira banknotes classification using deep convolutional neural networks. In 2018 26th Signal Processing and Communications Applications Conference (SIU) (pp. 1–4). IEEE.

Filter, J. (2022). Split-folders - PyPI. Retrieved January 15, 2024. from https://pypi.org/project/split-folders/

Foody, G. M. (2023). Challenges in the real world use of classification accuracy metrics: From recall and precision to the Matthews correlation coefficient. *PLOS ONE, 18*(10), e0291908. https://doi.org/10.1371/journal.pone.0291908

Galeana Pérez, D., & Bayro Corrochano, E. (2018). Recognition system for euro and Mexican banknotes based on deep learning with real scene images. *Computación y Sistemas, 22*(4). https://doi.org/10.13053/cys-22-4-3079

İyikesici, B., & Erçelebi, E. (2023). An efficient deep learning architecture for Turkish lira recognition and counterfeit detection. *Turkish Journal of Electrical Engineering and Computer Sciences, 31*(3), 678–692. https://doi.org/10.55730/1300-0632.4009

Keras Team. (2023a). Keras application. Keras. Retrieved January 15, 2024, from https://keras.io/api/applications/

Keras Team. (2023b). Optimizers. Keras. Retrieved January 17, 2024, from https://keras.io/api/optimizers/

Khashman, A., Ahmed, W., & Mammadli, S. (2019). Banknote issuing country identification using image processing and neural networks. In Proceedings (pp. 746–753). https://doi.org/10.1007/978-3-030-04164-9_98

Khashman, A., & Sekeroglu, B. (2005). *Multi-banknote identification using a single neural network*. In Proceedings (pp. 123–129).

Khashman, A., Sekeroglu, B., & Dimililer, K. (2005). Deformed banknote identification using pattern averaging and neural networks. In Proceedings of the 4th WSEAS International Conference on Computational Intelligence, Man-Machine Systems and Cybernetics (pp. 233–237).

Linkon, A. H. M., Labib, M. M., Bappy, F. H., Sarker, S., Jannat, M.-E., & Islam, M. S. (2020). Deep learning approach combining lightweight CNN architecture with transfer learning: An automatic approach for the detection and recognition of Bangladeshi banknotes. In 2020 11th International Conference on Electrical and Computer Engineering (ICECE) (pp. 214–217). IEEE. https://doi.org/10.1109/icece51571.2020.9393113

Mittal, S., & Mittal, S. (2018). Indian banknote recognition using convolutional neural network. In 2018 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU) (pp. 1–6). IEEE. https://doi.org/10.1109/iot-siu.2018.8519888

Pachón, C. G., Ballesteros, D. M., & Renza, D. (2023). An efficient deep learning model using network pruning for fake banknote recognition. Expert Systems with Applications, 233, 120961. https://doi.org/10.1016/j.eswa.2023.120961

Prakash, H., Yadav, A., Ushashree, P., Jha, C., Sah, G. K., & Naik, A. (2023). *Deep learning approaches for automated detection of fake Indian banknotes*. In 2023 IEEE International Conference on Integrated Circuits and Communication Systems (ICICACS) (pp. 1–5). IEEE. https://doi.org/10.1109/icicacs57338.2023.10100265

Sahin, O. (2018). *GitHub - Ozgurshn/TurkishBanknoteDataset.* Retrieved January 2, 2024, from https://github.com/ozgurshn/TurkishBanknoteDataset.

Veeramsetty, V., Singal, G., & Badal, T. (2020). CoinNet: Platform independent application to recognize Indian currency notes using deep learning techniques. *Multimedia Tools and Applications, 79*(31–32), 22569–22594. https://doi.org/10.1007/s11042-020-09031-0

Wang, L., Zhang, Y., Lanchi, X., Zhang, X., Guang, X., Li, Z., Li, Z., Shi, G., Hu, X., & Zhang, N. (2022). Automated detection and classification of counterfeit banknotes using quantitative features captured by spectral-domain optical coherence tomography. *Science & Justice, 62*(5), 624–631.https://doi.org/10.1016/j.scijus.2022.09.004.

How cite this article

Yeşiltepe, M., Eklıran, H., & Rasheed, J. (2024). Turkish Lira Banknote Classification using Transfer Learning and Deep Learning. *Acta Infologica*, Advance Online Publication. https://doi.org/10.26650/acin.1447456