ILKOM Jurnal Ilmiah Vol. 17, No. 1, April 2025, pp. 34-43
Accredited 2nd by KEMDIKTISAINTEK No. 10/C/C3/DT.05.00/2025; E-ISSN 2548-7779 | P-ISSN 2087-1716

Research Article Open Access (CC–BY-SA)

Implementation Of Deep Learning Using Convolutional Neural Network Method In A Rupiah Banknote Detection System For Those With Low Vision

Dinul Akhiyar [a,1,*]; Tukino [b,2]; Sarjon Defit [c,3]
[a] *Universitas Putra Indonesia YPTK Padang, Jl. Raya Lubuk Begalung, Kota Padang, Sumatera Barat 25145*
[b] *Institut Teknologi dan Bisnis Indobaru Nasional, Batam, 29444, Indonesia*
[c] *University Putra Indonesia YPTK, Padang, Jl. Raya Lubuk Begalung, Kota Padang, Sumatera Barat 25145*
[1] *dinul_akhiyar@upiyptk.ac.id;* [2] *tukino@indobarunasional.ac.id;* [3] *sarjon_defit@upiyptk.ac.id*
** Corresponding author*

Article history: Received July 02, 2024; Revised July 05, 2024; Accepted April 04, 2024; Available online April 20, 2025

Abstract

The application of deep learning in various sectors continues to grow due to its ability to provide efficient and effective solutions to complex problems. One significant implementation is in object detection, such as identifying Indonesian rupiah banknotes. This innovation aims to assist individuals with visual impairments in using money more effectively. At present, visually impaired individuals rely on conventional methods, such as identifying banknotes by touch, folding them in specific ways, or seeking assistance from others. However, these methods are often time-consuming, prone to error, and lack practicality in everyday situations. In this project, a system was developed using the Convolutional Neural Network (CNN) architecture combined with the YOLO (You Only Look Once) algorithm. YOLO is renowned for its speed and accuracy in real-time object detection, making it an ideal choice for detecting banknotes in moving images. The training dataset included 1,260 images, and the model underwent 7,000 iterations during training. As a result, the system achieved a high mean Average Precision (mAP) score of 97.65%, demonstrating its robustness and precision. For validation, 140 test images were utilized, which yielded an impressive mAP value of 97.5%. To further evaluate the system's reliability, tests were conducted under varying conditions, such as banknotes with creases, folds, or different lighting scenarios. These tests resulted in an mAP score of 88%, showcasing the system's adaptability to real-world conditions. This system provides significant benefits for individuals with visual impairments by offering a practical, efficient, and accurate solution for recognizing banknotes. With this technology, visually impaired users can interact with currency independently, reducing their reliance on others and traditional, less practical methods. This innovation not only enhances their autonomy but also fosters inclusivity in financial transactions. By integrating this system into mobile applications or wearable devices, its accessibility and usability can be further improved, paving the way for a broader societal impact.

Keywords: CNN; Deep Learning; Detection; Moving Images; Rupiah Banknotes; YOLO.

Introduction

Artificial intelligence or often called Artificial Intelligence (AI) is developing rapidly in the current modern era. Artificial intelligence is a part of computing that examines how machines (computers) can be made to behave and think like humans [1]. One of the most popular approaches is through machine learning (ML). As the name suggests, this approach includes designing and building algorithms to develop computer capabilities based on data that has been input. Therefore machine learning requires data to be used in the process training. Machine learning makes it possible to process a lot of data and learn patterns in the data so that future predictions can be made. One of the most widely used parts of machine learning is deep learning.

In recent years, deep learning has shown extraordinary performance. This is largely influenced by more powerful computing factors, large data sets and techniques for training deeper networks [2]. One of the algorithms used in deep learning is artificial neural network (ANN) [3]. This network has a flexible nature and can adapt independently to solve problems complex problems such as pattern recognition and classification [4]. ANN's ability to solve complex problems has been proven from various kinds of research, such as data analysis, pattern recognition, control systems, detection of medical phenomena, and so on [5]. Apart from the ANN algorithm, deep learning also has other algorithms such as convolutional neural networks (CNN). In contrast to ANN, the CNN algorithm for linear operations

uses a convolution process and the weights are no longer in one dimension, but in a minimum of four dimensions which are convolution kernels.

The development of the CNN algorithm model, which provides many conveniences in the world of deep learning, has resulted in several models that have been trained, called pre-trained models. Pre-trained models are usually trained on very large datasets such as Imagenet, so the quality of the model is very good. Some examples of pre-trained models that are popularly used in deep learning include Alexnet, R-CNN, Faster R-CNN, Single Shot Detector (SSD), and You Only Look Once (YOLO).

Certain fields such as sports, industry and medicine have made extensive use of deep learning technology in digital image processing, especially in the object detection section. There are several digital image processing studies related to object detection that have been carried out previously. In 2021 there was research on object tracking to detect and count the number of vehicles automatically using the Kalman filter method and Gaussian mixture model [6]. Similar research was also carried out in 2020 but used the convolutioinal neural network method [7]. Meanwhile, in 2021 research was carried out regarding detecting nominal money which was implemented on the Raspberry Pi [8]. Research in the same year was carried out in creating an application for detecting nominal money using the feature matching method [9]. Research on human object detection has also been carried out, for example in 2020 the convolutional neural network and long short term memory methods were applied to recognize human activity on CCTV in the shrimp pond area [10]. In the same year, research was also carried out in the form of room monitoring for CNN-based human detection with a push notification feature [11].

So far, low vision sufferers have used conventional methods such as arranging the nominal banknotes and folding the money to differentiate the nominal money. However, both methods still have several weaknesses, namely in terms of memory, the physical condition of the money and the absence of factors that determine honesty when buying and selling goods and services.

This research aims to implement deep learning in a rupiah banknote detection application to help people with low vision in Indonesia based on the Android operating system. The algorithm used for money recognition is YOLOv4-tiny. This deep learning system will be collaborated into an Android-based application as the main support for observing the program's performance in detecting objects in various conditions.

This research aims to implement deep learning in a rupiah banknote detection application to help people with low vision in Indonesia based on the Android operating system. The algorithm used for money recognition is YOLOv4-tiny. This deep learning system will be collaborated into an Android-based application as the main support for observing the program's performance in detecting objects in various conditions.

Method

A. Convolutional Neural Networks

Convolutional Neural Networks(CNN) is a development of Multilayer Perceptron (MLP) which is designed to process two-dimensional data [12]. CNN is a type of Deep Neural Network because the network level is multi-layered. CNN was first developed by researchers from NHK Broadcasting Science Research Laboratories, Kunihiko Fukushima and the concept was finalized by Yaan LeChun in his research [13].

In CNN you can find a hidden layer, where in each hidden layer there are several neurons that are connected to each other. The final layer connected to the hidden layer is called the output layer which is tasked with representing the final result of the convolution process. Based on **Figure 1**, it is generally known that the CNN layer is divided into 2 parts, namely the Feature Extraction Layer and the Classification Layer [14], [15].

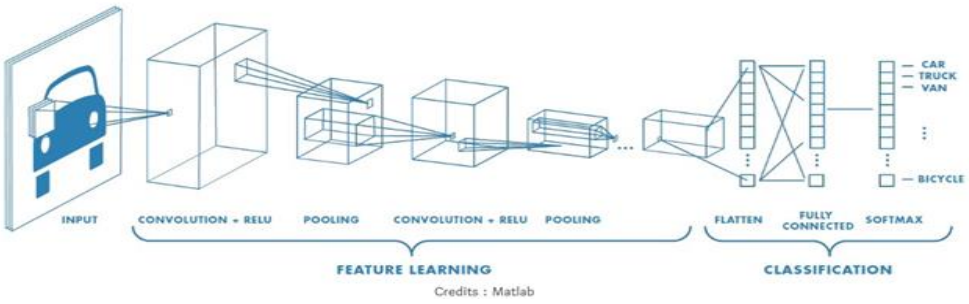

Figure 1. General Description of The Convolution Process

B. YOLO

YOLO (You Only Look Once) is an algorithm that is used to detect an object in real-time. Technically, artificial neural networks are the approach used by YOLO to be able to detect an object. The YOLO process begins by dividing the image into several regions, then predicting each bounding box and the probability for each region. The YOLO architecture consists of 27 CNN layers, namely 24 layers consisting of a convolutional layer and a pooling layer with a kernel size of 2×2, followed by 2 fully connected layers and a final decision layer with a kernel size of 1x1 as in **Figure 2**. YOLO divides the input image into a grid. size S×S where Each cell of the grid will predict the bounding box and produce a value for each class. Each bounding box consists of five predicted values, namely x coordinate center, y coordinate center, cell width (w), cell height (h) and confidence value [16].

Convolutional layersusing filters as parameters that will be updated in the learning process. The filter is initialized with a certain value, then from the grid calculation on the image with the filter value the output value is produced in array form. Then by shifting (convolution) the filter at every possible position in the image to produce an activation map [17].

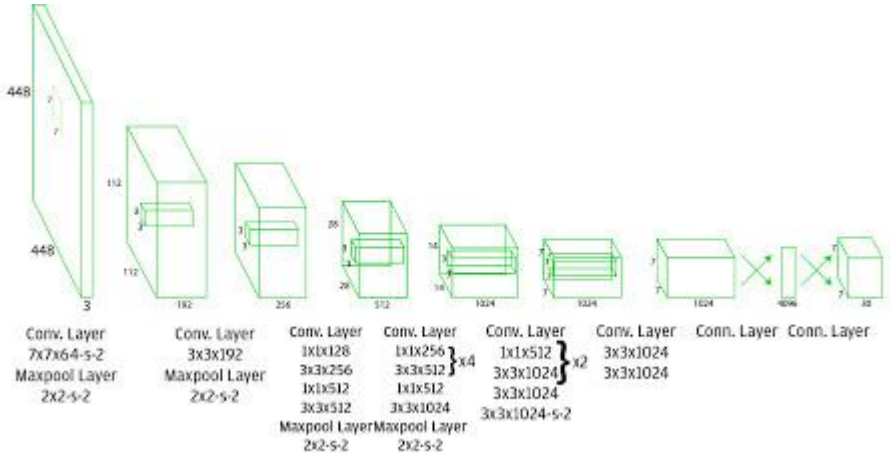

Figure 2. YOLO Architecture

In dividing the input image into a grid size of S×S, the central part which is the midpoint of the grid cell is responsible for the object detection process. Each grid cell predicts a bounding box and a confidence value for each box. This confidence value states how much confidence the system has in the bounding box which contains the core information of the object and also how accurate the grid cell is in predicting the bounding box [18], [19]. Mathematically, the confidence value can be expressed as follows:

$$Confidence\ Score = P_r(object) \times IOU_{pred}^{truth} \quad (1)$$

Where P_r = probabilitas class object and IOU = Intersection Over Union

IOU (Intersection Over Union) is a method used to evaluate the accuracy of object detection in a data set. IOU requires two areas for the intersection and union processes, namely the ground-truth bounding box area and the predicted bounding box [20], [21].

The IOU equation can be stated as follows:

$$IOU = \frac{slice}{combined} \quad (2)$$

C. System Specifications

The system design in this final research project consists of three main stages, namely determining the specifications of the computer as a device used for the process trainingdata, designing a money detection program written using the Python programming language version 3.7 using Google Colab, and determining smartphone specifications [22]. The detection process is divided into fourteen different classes based on the nominal value of the banknotes in rupiah. The smartphone is the final device that will be the testing ground for the results of the training model that has been carried out on the computer device [23].

The visual information that is entered into the system is in the form of moving images (video) taken by the Samsung Galaxy S7 Edge smartphone camera in real-time [24]. The Opencv library is used to be able to launch the camera on

a smartphone. The dataset used in the training process was obtained by taking photos of each nominal rupiah banknote using a Samsung Galaxy S7 Edge, Asus Zenfone Max Pro M1 and iPhone 8+ smartphone camera as well as images from Google Images [25].

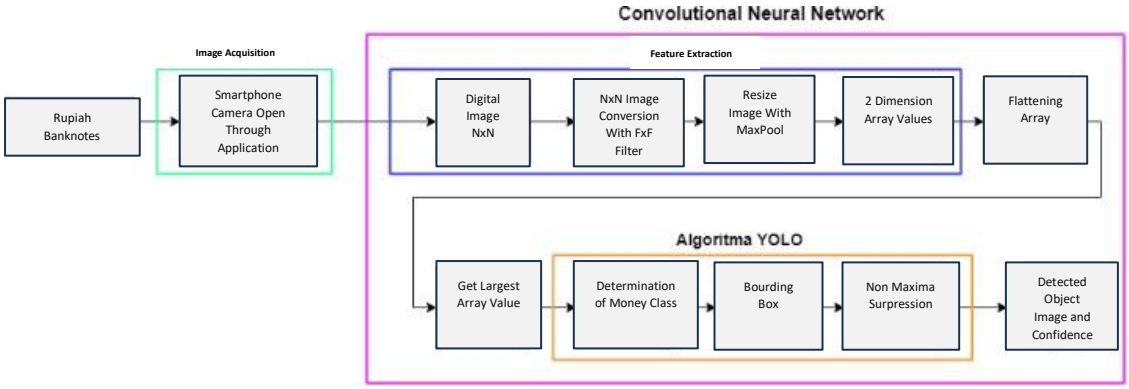

Figure 3. Overall System Block Diagram

The money image captured by the smartphone camera will be processed in an image processing system based on the training model that has been input into the program. The initial process in the system after image acquisition is to run a CNN by convolving the image size with a filter of size fxf so that the output is produced in the form of a 2-dimensional array value. Max pooling is used in programs that aim to reduce input from the convolution layer [26]. Then the flattening process is carried out, the results of the flattening process are processed on the fully connected layer by running the YOLO algorithm to get an output in the form of an object image that has been classified with a bounding box [27]. The overall system design can be described by the system block diagram in **Figure 3**.

D. Training Data Collection

The data used in this research is a collection of images of 2021 issue rupiah banknotes consisting of 7 types of nominal money. Each money has 2 classes because it has a front and a back side so the total number is 14 classes. Data collection for these fourteen money classes was obtained from personal photo data and Google Images [28].

Training data is collected by collecting images according to their class. Each nominal amount of money has training data of 200 images consisting of 100 images of the front side and 100 images of the back side. The total training data collected from 7 nominal banknotes amounted to 1,400 images. The collected data is then divided into 3 categories, namely training, validation and testing data with weights of 70%, 20% and 10% respectively [29].

E. Data Training

Training data is carried out for the process of extracting features from each class to be classified. The training process is divided into two parts, namely the annotation and training processes. **Figure 4** is an example of an image in the annotation process using the labelimg application [30].

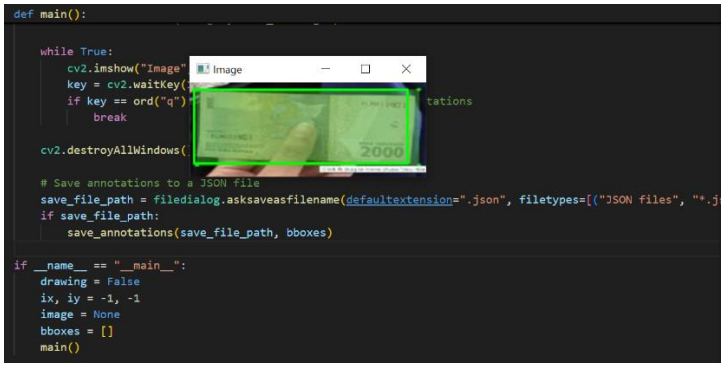

Figure 4. Training Data Annotation Process

Frameworks used for the training process is the darknet using Google Colab. The number of classes used for object classification in the training process is listed in the previously determined Obj.data file. In this file there is also a distribution of the amount of training data, validation data and testing data. The amount of each data distribution is 70% for training data (980 images), 20% validation data (280 images), and 10% testing data (140 images). Meanwhile,

to identify the number of classes, the program will read a file of type .labels which is in the training data. The number of layers, filters and CNN architecture design used are designed in the custom-yolov4-tiny-detector.cfg file which is then called from the terminal to be used in the training process. The number of iterations that will be carried out is 7000 iterations [31].

F. Testing Data

The testing process is carried out after the training process is complete with the aim of testing whether the program can recognize an object outside the training data and validation data that has been used previously. The set of images included in the testing data is placed in a different directory from the training data and validation data. The amount of testing data that will be used is 10% of the total 1,400 images.

The first stage that will be carried out is loading the same 38 layers used in the training process. Next, the program will carry out a detection process using the weight files that have been saved and produce an object classification along with its confidence level.

Results and Discussion

A. Results

After carrying out the training and testing process with a total of 1,400 images of data and program design, the next stage is to discuss and analyze matters regarding the testing to be carried out. The purpose of this testing is to determine the output results and level of accuracy of the program that has been implemented. The input data used is in the form of moving images originating from an Android cellphone camera. Testing is divided into 3 parts, namely:

a) Testing program detection performance with variations in money conditions.

b) Testing program detection performance under low light conditions.

c) Testing program performance on detection response time.

1. Testing Program Detection Performance with Varying Money Conditions

Testing program performance with variations in money conditions has 3 experimental variations, namely the whole money condition, the left side of the money condition, and the right side of the money condition. The selection of this variation was made to test whether the program could recognize objects when only a few parts of the money were included in the camera frame

a. Program Performance Testing Money Conditions All sections

In testing the performance of the program with the money condition, all parts indicate that when the program detects a money object, all parts of the money enter the camera frame. This test will be carried out on all 14 classes, with 5 images in each class. The object detection process is carried out by pointing the smartphone camera at the money object to be detected.

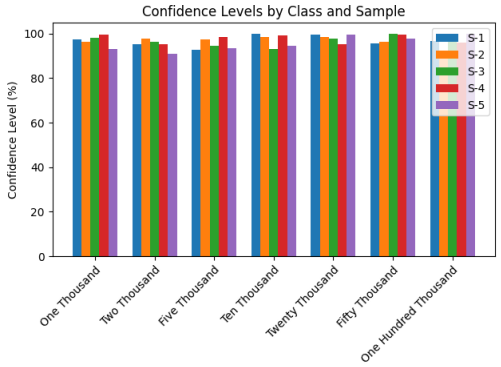

Figure 5. Program Performance Graph for Classes Thousand, 2 thousand, 5 thousand, 10 thousand, 20 thousand, 50 thousand, 100 thousand All Parts

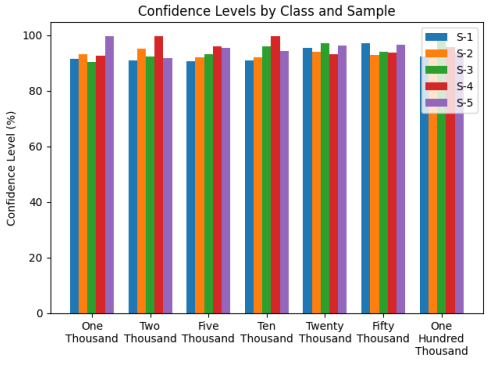

Figure 6. Graph of Class Program Performance of One Thousand, Two Thousand, Five Thousand, Ten Thousand, Twenty Thousand, Fifty Thousand, One Hundred Thousand

Based on **Figure 5** and **Figure 6**, it is known that the program's confidence value in detecting objects in 14 classes always changes with each sample. In the picture there is a parameter Sn which means the nth sample in that class,

where n is in the range 1 to 5. In this experiment the program was not successful in detecting all objects. The program has more difficulty detecting the left side of the front side than the back side of the money. We can see that several classes that represent the front side of money have unstable confidence levels and there are even 8 money objects that cannot be recognized or are predicted incorrectly by the program. Meanwhile, the classes that represent the back side of money have more even confidence levels than the front side and there is only 1 money object that cannot be recognized by the program. This can occur due to differences in training data for each class previously used, thus affecting the money detection results if only the left part enters the camera frame.

Based on **Figure 5** and **Figure 6**, it is known that the program's confidence level value in detecting objects in 14 classes always changes with each sample. In the picture there is a parameter Sn which means the nth sample in that class, where n is in the range 1 to 5. In this experiment the program was not successful in detecting all objects. There was a prediction error made by the program when detecting the fourth sample in the "10 thousand" class and the fifth sample in the "one thousand" class.

b. Program Performance Testing Left Side Money Conditions

In testing the performance of the program with the left side of the money, it indicates that when the program detects the money object, only the left side of the money enters the camera frame. This test will be carried out on all 14 classes, with 5 images in each class. The object detection process is carried out by pointing the smartphone camera at the money object to be detected.

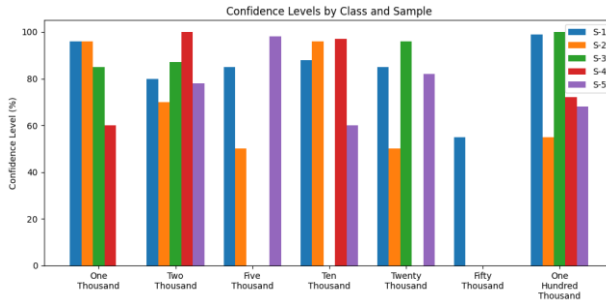

Figure 7. Graph of Program Performance for Classes One Thousand, 2 Thousand, 5 Thousand, 10 Thousand, 20 Thousand, 50 Thousand, 100 Thousand Left Part

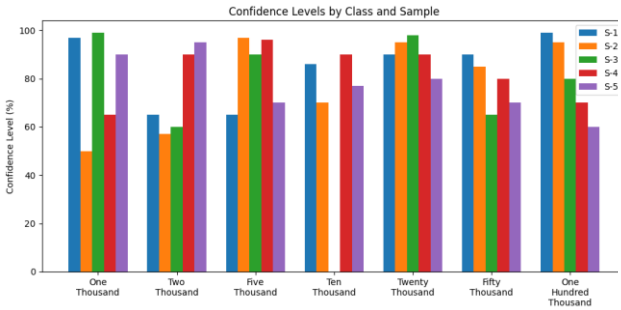

Figure 8. Graph of Program Performance for Classes One Thousand, 2 Thousand, 5 Thousand, 10 Thousand, 20 Thousand, 50 Thousand, 100 Thousand Left Part

c. Program Performance Testing Right Part Money Conditions

In testing the performance of the program with the right side of the money, it indicates that when the program detects the money object, only the right side of the money enters the camera frame. This test will be carried out on all 14 classes, with 5 images in each class. The object detection process is carried out by pointing the smartphone camera at the money object to be detected.

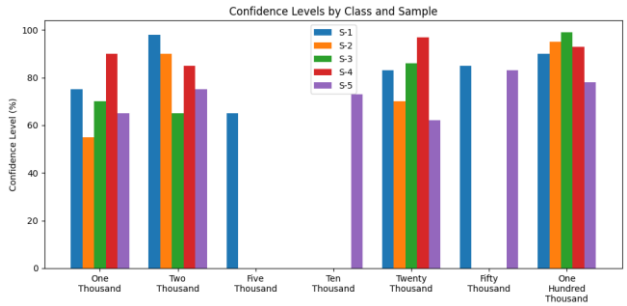

Figure 9. Program Performance Graph for Classes One Thousand, 2 Thousand, 5 Thousand, 10 Thousand, 20 Thousand, 50 Thousand, 100 Thousand Right Part

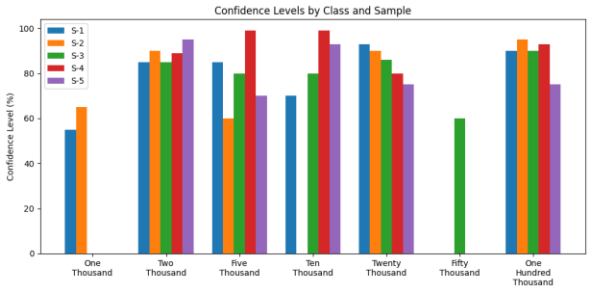

Figure 10. Program Performance Graph for Classes One Thousand, 2 Thousand, 5 Thousand, 10 Thousand, 20 Thousand, 50 Thousand, 100 Thousand Right Part

Based on **Figure 9** and **Figure 10**, it is known that the program's confidence level value in detecting objects in 14 classes always changes with each sample. In the picture there is a parameter Sn which means the nth sample in that

class, where n is in the range 1 to 5. In this experiment the program was not successful in detecting all objects. The program has more difficulty detecting the right side of the front side than the back side of the money. We can see that several classes that represent the front side of money have unstable confidence levels and there are even 11 money objects that cannot be recognized or are predicted incorrectly by the program. Meanwhile, the classes that represent the back side of money have more even confidence levels than the front side and there are 8 money objects that cannot be recognized by the program. This can occur due to differences in training data for each class previously used, thus affecting the money detection results if only the left part enters the camera frame.

2. Testing the Program's Detection Performance of Variations in Low Light Conditions

When testing program performance in low light conditions, it indicates that when the program detects a money object, the ambient light conditions are dim. This test will be carried out on all 14 classes, with 5 images in each class. The object detection process is carried out by pointing the smartphone camera at the money object to be detected.

Based on **Figure 11** and **Figure 12**, it is known that the program's confidence value in detecting objects in 14 classes always changes with each sample. In the picture there is a parameter Sn which means the nth sample in that class, where n is in the range 1 to 5. In this experiment the program was not successful in detecting all objects. The program has more difficulty detecting the back side than the front side of money. We can see that several classes that represent the back side of money have unstable confidence levels and there are even 14 money objects that cannot be recognized or are predicted incorrectly by the program. Meanwhile, the classes that represent the front side of money have more even confidence levels than the back side and there are 4 money objects that cannot be recognized by the program. This can happen because the light conditions are dim so it is difficult for the program to recognize the money object to be detected.

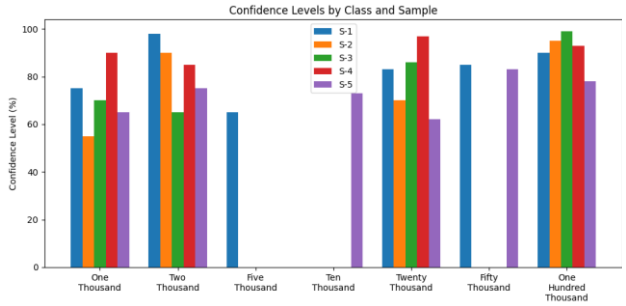

Figure 11. Program Performance Graph Class One Thousand, 2 Thousand, 5 Thousand, 10 Thousand, 20 Thousand, 50 Thousand, 100 Thousand Low Light

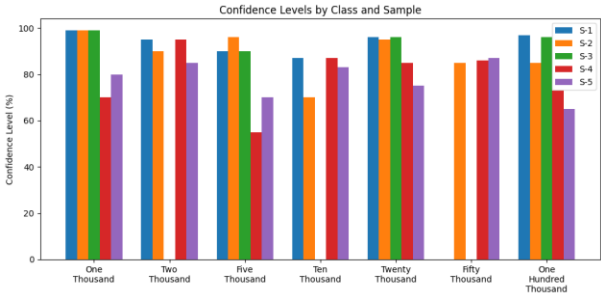

Figure 12. Program Performance Graph for Classes One Thousand, 2 Thousand, 5 Thousand, 10 Thousand, 20 Thousand, 50 Thousand, 100 Thousand Right Part usand, 100 Thousand Right Part

3. Testing program performance on detection response time

In testing program performance on detection response time, the aim is to determine the speed of the program in detecting money objects. The response time calculation starts when the object has entered the camera frame and stops when the program can detect the nominal amount of money. This test will be carried out on all 14 classes, with 5 images in each class. The data used in testing program performance regarding detection response time is program performance testing data with cash conditions in all parts.

After testing and data analysis, it can be seen that the average program response time for all classes is between 0.97 – 1.82 seconds. The fastest average response time is in the "fifty thousand" class with a value of 0.99 seconds, while the longest response time is in the "2 thousand" class with a value of 1.82 seconds. Then from this data we can calculate the average overall response time of the program for detecting money objects with 14 classes of 1.28 seconds.

4. Calculation of mAP Value for Overall Testing

The program performance evaluation process can be calculated using the values obtained from the precision and recall of each class. The precision and recall calculation process uses three confusion matrix parameters, namely true positive (TP), false positive (FP) and false negative (FN). The precision value is obtained by comparing the total true positive (TP) parameters with the amount of data that is predicted to be positive. Meanwhile, the recall value is obtained by comparing the total true positive (TP) parameters with the amount of data that is actually positive. The level of accuracy can be calculated using the mean average precision (mAP) method. The mAP value is usually used

as an indicator of accuracy performance in object detection such as the faster R-CNN algorithm, SSD, and including YOLO.

Next, to calculate the mean average precision (mAP) value, an 11-point interpolation technique was used. This technique will use the highest value of precision if there is a similar value. Interpolated precision is the average precision measured at 11 recall levels with the same distance, namely from 0.0, 0.1, 0.2, 0.3 to 1.0. **Table 1** shows the relationship between recall and precision values after carrying out the 11 point interpolation technique.

Table 1. Recall and precision values after 11 point interpolation

Recall	Interpolated Precision
0.0	0.85
0.1	0.85
0.2	0.85
0.3	0.85
0.4	0.85
0.5	0.85
0.6	0.85
0.7	1.00
0.8	0.94
0.9	0.78
1.0	1.00

Based on **Table 1**, it can be seen the relationship between each recall and precision values after carrying out the 11 point interpolation technique. The recall value in the range 0.0 to 0.6 has the same precision value, namely 0.85, because in the overall test the lowest recall value was 0.6, so the range of recall values below 0.6 will have the same precision value. The same. Next, the process of calculating the mean average precision (mAP) can be carried out because all the precision and recall values have been obtained. The calculation of the mean average precision (mAP) value is shown as follows:

$$\frac{1}{11} = [(7 \times 0{,}85) + (1 \times 1) + (1 \times 0{,}94) + (1 \times 0{,}78) + (1 \times 1)] = 0{,}88$$

Based on the calculation of the mean average precision (mAP) value using the formula above, it can be seen that the program's mAP value after testing variations in money conditions (whole part, right part, left part) and variations in low light conditions produces a value of 88%.

B. Discussion

This discussion aims to evaluate the performance of the banknote detection system implemented using deep learning based on the Convolutional Neural Network (CNN) method with the YOLOv4-tiny algorithm. The test results show that the system achieves a relatively high level of accuracy, but there are still several challenges that need to be addressed to optimize its performance, particularly in certain conditions.

1. Banknote Condition Variability: The program's performance test under varying banknote conditions reveals variability in detection confidence across different banknote classes. The system particularly struggles to detect the left side of the banknote's front, leading to some classes being undetected. This suggests the need for further training with more balanced data, especially for the front side of the banknotes. Adjustments to the algorithm are also necessary to improve detection accuracy.
2. Low Light Conditions: The program shows a significant decline in performance when detecting banknotes under low light conditions, particularly on the back side of the banknotes. Poor lighting affects the program's ability to correctly recognize objects. Therefore, improvements in the detection algorithm for low-light conditions, such as enhancing image processing techniques or using more sensitive hardware, should be considered.
3. Detection Response Time: The response time test shows that the system takes varying amounts of time to detect banknotes across different classes. The average detection time ranges from 0.97 to 1.82 seconds. Although this time is within an acceptable range, the variation across classes suggests opportunities for further optimization. Developers can evaluate bottlenecks in the detection process and implement optimizations to improve detection speed.

4. Mean Average Precision (mAP) Calculation: The mAP calculation is used as a metric to evaluate the overall accuracy of the program in detecting banknotes. An mAP value of 88% demonstrates that the program is fairly effective in detecting objects under various test conditions. However, to further improve performance, developers may consider strategies such as adjusting model parameters, increasing the diversity of training data, or implementing more advanced object detection techniques.
5. Recommendations for Improvement:
   a) Class Balancing: Disparities in detection performance across different classes highlight the importance of balanced training data to ensure equitable learning across all categories.
   b) Algorithm Refinement: Refining the algorithm to enhance robustness and adaptability to diverse environmental conditions, such as low light, can lead to more reliable detection results.
   c) Continuous Evaluation: Regular evaluation and iterative development based on feedback and real-world testing are essential to ensure continuous improvement and optimal functionality of the program.

In conclusion, while the program shows promising capabilities in detecting banknotes, the results from performance testing emphasize the need for ongoing refinement to enhance accuracy, reliability, and responsiveness under various conditions.

Conclusion

In calculating the mAP value of the system based on the entire test which includes testing with various variations of money conditions (entire part, left part and right part) as well as testing in various dim light conditions, it reaches a mAP value of 88%. The average response time for the program to detect money is 1.28 seconds. From the results of this research, it is recommended to use more training data and training iterations to obtain a higher level of accuracy. Apart from that, the use of a camera as well as the distance and angle of image capture as well as an optimal image annotation process can increase accuracy.

References

[1] Mukhopadhyay S, Hossain S, Ghosal SK, Sarkar R. Secured image steganography based on Catalan transform. *Multimed Tools Appl*. 2021; 80(9): 14495-14520.

[2] Ghosal SK, Mukhopadhyay S, Hossain S, Sarkar R. Application of Lah transform for security and privacy of data through information hiding in telecommunication. *Trans Emerg Telecommun Technol*. 2021; 32(2):e3984.

[3] Wang, T., et al.: Video matting via consistency-regularized graph neural networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 4902–4911 (2021.

[4] Wu, Z., et al.: A comprehensive survey on graph neural networks. *IEEE Transact. Neural Networks Learn. Syst.* 32(1), 4–24 (2020). https://doi.org/10.1109/tnnls.2020.2978386.

[5] F. Han, J. Yao, H. Zhu, and C. Wang, "Underwater Image Processing and Object Detection Based on Deep CNN Method," J. Sensors, vol. 2022, p. 20, 2022.

[6] M. Elleuch, R. Maalej, and M. Kherallah, "A New design based-SVM of the CNN classifier architecture with dropout for offline Arabic handwritten recognition," Procedia Comput. Sci., vol. 80, pp. 1712–1723, 2022

[7] Protogerou, A., et al.: A graph neural network method for distributed anomaly detection in IoT. *Evolving Syst.* 12(1), 19–36 (2021). https://doi.org/10.1007/s12530-020-09347-0.

[8] P. Ma, "Recognition of handwritten digit using convolutional neural network," Proc. - 2020 Int. Conf. Comput. Data Sci. CDS 2020, vol. 19, no. 2, pp. 183–190, 2023.

[9] S. Indolia, A. K. Goswami, S. P. Mishra, and P. Asopa, "Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach," Procedia Comput. Sci., vol. 132, pp. 679–688, 2022.

[10] P. Wang, X. Zhang, and Y. Hao, "A Method Combining CNN and ELM for Feature Extraction and Classification of SAR Image," J. Sensors, vol. 2021, 2021.

[11] A. Patil and M. Rane, "Convolutional Neural Networks: An Overview and Its Applications in Pattern Recognition," Smart Innov. Syst. Technol., vol. 195, pp. 21–30, 2023.

[12] W. Lee, K. Ko, Z. Geem, K. Sim, and E. Engineering, "Method that Determining the Hyperparameter of CNN

using HS algorithm," J. Korean Inst. Intell. Syst., vol. 27, no. 1, pp. 22–28, 2022.

[13] Zagrouba R, Kardi A. Comparative study of energy efficient routing techniques in wireless sensor networks. *Information.* 2021; 12(1): 42.

[14] T. Tukino., M. Pratiwi and S. Defit, "Deep Learning Based Technical Classification of Badminton Pose with Convolutional Neural Networks," ILKOM Jurnal Ilmiah, vol. 16 no. 1, pp. 76-86, 2024. https://doi.org/10.33096/ilkom.v16i1.1951.76-86

[15] Pandey, K., Patel, S.: Deep learning with convolutional neural networks: from theory to practice. In: 2023 7th International Conference on Trends in Electronics and Informatics (ICOEI), pp. 1217–1224. IEEE (2023).

[16] P. Lakhani, B. Sundaram, "Deep learning at chest radiography: automated classification of pulmonary tuberculosis by using convolutional neural networks". Radiology 284:574–582, 2022.

[17] F. Lu, F. Wu, P. Hu, Z. Peng, D. Kong D, "Automatic 3D liver location and segmentation via convolutional neural network and graph cut". Int J Comput Assist Radiol Surg 12:171–182, 2023.

[18] F. Milletari, N. Navab, S-A. Ahmadi, "V-net: fully convolutional neural networks for volumetric medical image segmentation". In: Proceedings of the 2023 Fourth International Conference on 3D Vision (3DV). https://doi.org/10.1109/3DV. 2021.79, 2023.

[19] Fenza, G., et al.: Cognitive name-face association through context-aware graph neural network. *Neural Comput. Appl.* 34(13), 10279–10293 (2022). https://doi.org/10.1007/s00521-021-06617-z.

[20] Li T, Xu Y, Luo J, He J, Lin S. A method of amino acid terahertz spectrum recognition based on the convolutional neural network and bidirectional gated recurrent network model. *Sci Program*. 2021; 2021:2097257.

[21] Heaton, J., (2022). AIFH, Volume 3: Deep Learning and Neural Networks. Washington DC: Heaton Research, Inc.

[22] Hijazi, S., Kumar, R., & Rowen, C. (2022). Using convolutional neural networks for image recognition. Cadence Design Systems Inc.: San Jose, CA,USA.

[23] F. Han, J. Yao, H. Zhu, and C. Wang, "Underwater Image Processing and Object Detection Based on Deep CNN Method," J. Sensors, vol. 2022, p. 20, 2022.

[24] M. Elleuch, R. Maalej, and M. Kherallah, "A New design based-SVM of the CNN classifier architecture with dropout for offline Arabic handwritten recognition," Procedia Comput. Sci., vol. 80, pp. 1712–1723, 2021.

[25] B. Sekeroglu and I. Ozsahin, "Detection of COVID-19 from Chest X-Ray Images Using Convolutional Neural Networks," SLAS Technol., vol. 25, no. 6, pp. 553–565, 2022.

[26] P. Ma, "Recognition of handwritten digit using convolutional neural network," Proc. - 2022 Int. Conf. Comput. Data Sci. CDS 2022, vol. 19, no. 2, pp. 183–190, 2022.

[27] S. Indolia, A. K. Goswami, S. P. Mishra, and P. Asopa, "Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach," Procedia Comput. Sci., vol. 132, pp. 679–688, 2021.

[28] P. Wang, X. Zhang, and Y. Hao, "A Method Combining CNN and ELM for Feature Extraction and Classification of SAR Image," J. Sensors, vol. 2022, 2022.

[29] A. Patil and M. Rane, "Convolutional Neural Networks: An Overview and Its Applications in Pattern Recognition," Smart Innov. Syst. Technol., vol. 195, pp. 21–30, 2021.

[30] W. Lee, K. Ko, Z. Geem, K. Sim, and E. Engineering, "Method that Determining the Hyperparameter of CNN using HS algorithm," J. Korean Inst. Intell. Syst., vol. 27, no. 1, pp. 22–28, 2021.

[31] M. Sadeghi et al., "PERSIANN-CNN: Precipitation estimation from remotely sensed information using artificial neural networks–convolutional neural networks," J. Hydrometeorol., vol. 20, no. 12, pp. 2273–2289, 2022.