Library Attendance System using YOLOv5 Faces Recognition

Mardiana
Informatics Engineering
University of Lampung
Bandar Lampung, Indonesia
mardiana@eng.unila.ac.id

Meizano Ardhi Muhammad
Informatics Engineering
University of Lampung
Bandar Lampung, Indonesia
meizano@eng.unila.ac.id

Yessi Mulyani
Informatics Engineering
University of Lampung
Bandar Lampung, Indonesia
yessi.mulyani@eng.unila.ac.id

***Abstract*— Recognizing a large number of faces at the same time is an algorithmic and computational challenge. The integration of a facial recognition system with an existing automation system in a library is also a big challenge because of the many sub-systems that operate in it. The aim is to develop a prototype of a library attendance system to assist library management related to facial recognition of users who visit the library. This study uses image processing focuses on object detection using the YOLOv5 algorithm. The library attendance system integrates 3 sub-systems: API service, face recognition using YOLOv5, and visitor identification system. The results obtained are that the library attendance system can function properly, can read the API service, and display information on the results of face detection therefore the system can be used by the existing library automation system.**

Keywords*— *library attendance system, face recognition, image processing, machine learning, YOLOv5, visitor identification system

I. INTRODUCTION

Various kinds of research in the field of image processing, especially those related to face detection and face recognition, have been carried out. This research can be applied to help overcome problems that exist in libraries. Most of the librarian's works in libraries has been assisted by library automation systems [1]. However, visual tasks, such as recognizing the presence of users, still have problems. The presence of visitors is known from the use of RFID and barcode cards at the library entrance. The use of the card poses security problems in the form of card misuse by others. Computational technology and algorithms which able to recognize many objects need to be used to answer these problems. The existing library automation system needs to be integrated with other systems that function for facial recognition.

Several studies have been conducted to detect objects using YOLO. YOLO is one of the Convolutional Neural Network (CNN) algorithms [2] which was developed into several other algorithms, including R-CNN, Fast R-CNN, Faster R-CNN, and YOLO. Based on this research, YOLO has advantages that can be used to detect objects. As in studies that aim to detect objects in the form of road signs, the results show the YOLO algorithm can detect with an accuracy of 74% [3]. In addition, research that aims to detect spherical objects on humanoid robot soccer can detect with an accuracy of 60% [4].

The YOLO (You Only Look Once) algorithm is an algorithm developed to detect an object in real-time [5]. The detection system uses a repurpose classifier or localizer to detect. YOLO already has many versions, starting with the YOLO version, YOLOv2, YOLOv3 until the latest one is YOLOv5. However, the application of the YOLOv5 algorithm for the library attendance system has not been done much. So, it's an important thing to do.

The prototype library attendance system developed in this study will use the YOLOv5 algorithm because it is considered better than the previous version. With the technology, the system not only recognizes the user's face but can also store the recognized user data in a database. This will make it easier for librarians to process attendance data for recognized or unrecognized visitors.

II. METHOD

There are three-part of development based on three sub-systems in the library attendance system: API Service, Face recognition using YOLOv5, visitor identification system. System model illustrated in Fig. 1.

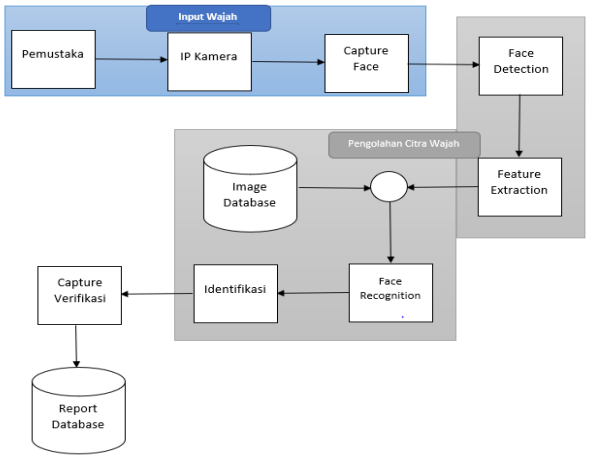

Fig. 1. Concept Model Library attendance system

The stages of development are as follow:

1. Development of API Service for Library Attendance System
2. Building face recognition using YOLOv5
3. Development of Visitor Identification System
4. Comprehensive System Testing

Development of API Service and Visitor Identification System in stages 1 and 3 is carried out using general software development lifecycle.

Building face recognition using YOLOv5 in stage 2 requires more careful consideration. YOLO (You Only Look Once) is a real-time object detection algorithm with a high level of accuracy. The YOLO algorithm uses a convolution neural network for object detection. YOLO uses an artificial neural network (ANN) approach to detect objects in an image. This network divides the image into several regions and predicts each boundary city. YOLO is very good at predicting images and classifying images. YOLO can also detect multiple images at once[6]. YOLO v5 has 4 models for training data: YOLO v5-s, YOLO v5-m, YOLO v5-l, and YOLO v5-x [7]. The four models have differences in network architecture or the number of layers and the number of parameters. The YOLO v5 architecture has 3 parts: the Backbone Model, the Neck Model, and the Head Model.

1) Backbone Model

The backbone model is used to extract important features from a given input image. YOLO v5 uses the CSP (Cross Stage Partial) Network as the backbone to extract informative features from the input image.

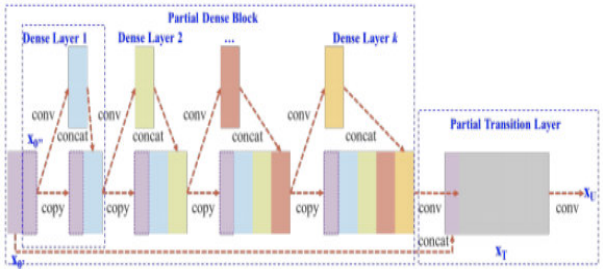

Fig. 2. DenseNet CSP [8]

Fig. 2. is a feature extraction process using DenseNet CSP. CSP (Cross Stage Partial) is based on the same principle from DenseNet except that the CSP input is separated into 2 portions, with some of it being forwarded through dense blocks which will perform convolution and part of it being sent directly to the next stage without being processed. Then the part of the block that performs the convolution will be combined (concat). Before entering the neck using FPN, additional blocks are added, namely the SPP (Spatial Pyramid Pooling) layer

2) Neck Model

Neck style on YOLOv5 using PANet which adopts the Feature Pyramids Network (FPN) structure. The PANet model is used to help the model generalize well for scaling objects. This is very helpful in identifying the same object with different sizes and scales. Fig. 3 is a PANet image for the neck, this PANet is used for feature classification.

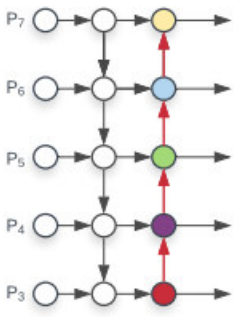

Fig. 3. PANET [9]

3) Head Model

Head model used to perform the final detection, this model applies anchor boxes to the features and produces a final output vector with class probabilities, objectivity scores, and bounding boxes. YOLOv5 uses leaky ReLU and Sigmoid as activation functions. The Leaky ReLU activation function is used in the middle layer or hidden layers while Sigmoid is used in the final detection layer. The activation function is a function to calculate the number of input weights and biases in the artificial neural network so that it can activate and deactivate neurons.

The loss value in YOLOV5 is calculated based on objectivity scores, class probability scores, and bounding box regression scores.

The stages for building face recognition using YOLOv5 can be explained as follows:

1) Building a Dataset

The first process is the process of building a dataset before training, with the steps in Fig. 4:

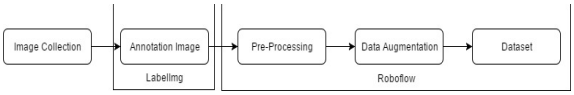

Fig. 4. Flowchart of Building a Dataset

To build the dataset, an Image collection is needed in the form of photos in JPG/Img format, then the photos are annotated or labeled with each photo using LabelImg. The result of the annotation is in the form of a file in XML format. Then the datasets with XML files are combined for image pre-processing.

2) Training Dataset

The second process is a training process using a custom YOLO v5-s model and the results of the training or Yolov5 weight model are used for the detection process, as shown in Fig. 5 :

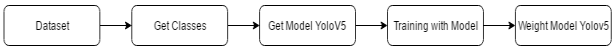

Fig. 5. Training Dataset Flowchart

The dataset is read and a class is formed which will be used to create a Yolov5-s custom detection model. Next, this file is used for training the dataset. After completing the training, the data from the training can be tested using photos and videos.

3) Object Detection Model Testing

The third process is the object detection process using a trained model.

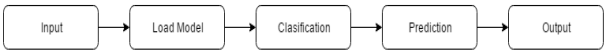

Fig. 6. Test Flowchart

Fig. 6 is the testing phase using photos and videos. Input can be photos or videos for testing. The detection process is carried out by loading the model that has been built, then classification and prediction are carried out using bounding boxes and confidence scores. The results displayed are in the form of prediction boxes, confidence values, and object classes.

Stage 4 is Comprehensive System Testing. To evaluate the integrated sub-system, comprehensive system testing is conducted by incorporating the three-part sub-system. Fig. 7. represents the comprehensive system testing phase.

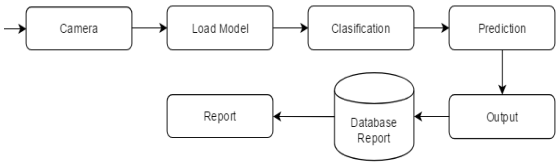

Fig. 7. Comprehensive System Testing Flowchart

First, the input is from a face captured by the camera. Then the detection process will load the model that has been built, then perform classification and prediction using bounding box and confidence score. The results displayed are in the form of prediction boxes, confidence values, and object classes.

The detection results in the form of an object class (cls) will be stored in the report database and then the data in the report database will appear on the report page. The report page is built to display data in the form of a number, user name, NPM, date of attendance, and time of attendance.

III. RESULTS AND DISCUSSION

A. API Service for Library Attendance System

To send the results of the facial recognition sub-system, a request to the web service is submitted via the HTTP protocol. Response provided by the web service contains information about the time, the number of visitors (humans), and recognized users.

The nature of the web service API provided is agnostic so that it does not require the use of a particular programming language to access it.

The Reader Attendance API service is accessed via JavaScript. The time required for each request in the 100x test is 500 milliseconds (ms).

B. Face Recognition using YOLOv5

The data used for the facial recognition sub-system is 1528 data consisting of 1420 training data, 69 validation data, and 39 test data. The percentage distribution is 92.9% training data, 4.5% valid data, and 2.6% test data.

1) Building a Dataset

Data labeling is done manually using labelImg software which marks objects with rectangles as shown in Fig. 8.

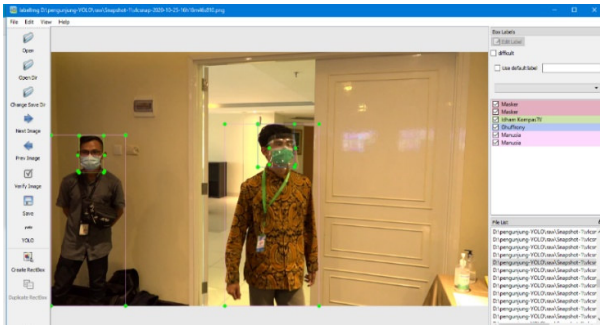

Fig. 8. Labeling using labelImg

The object class used in this research is a total of 37 objects. Objects include general visitors (humans), and identified users.

2) Training Dataset

YOLOv5 repository is used as the basic framework. all repositories and dependencies of YOLOv5 must be fulfilled. Dataset training for the face recognition sub-system was carried out on 1420 images that had been labeled. The object in the image is resized to 416x416. The training was carried out for 1000 epochs with a total of 16 batches. The basic model used is yolov5s.yaml.

The computational specifications used for training are CUDA Nvidia Tesla T4 with 15079MB of memory. The training process was completed in 3.736 hours for 1420 images labeled with 1000 epochs. The resulting weight model is 14.9MB.

3) Object Detection Model Testing

Based on the results of the training, it can be seen that the level of minimized return is achieved when the graph starts to form an elbow (elbow method) in 700 epochs at mA@0.5, mAP@0.5:0.95, Precision, and Recall. The results can be seen in Fig. 9.

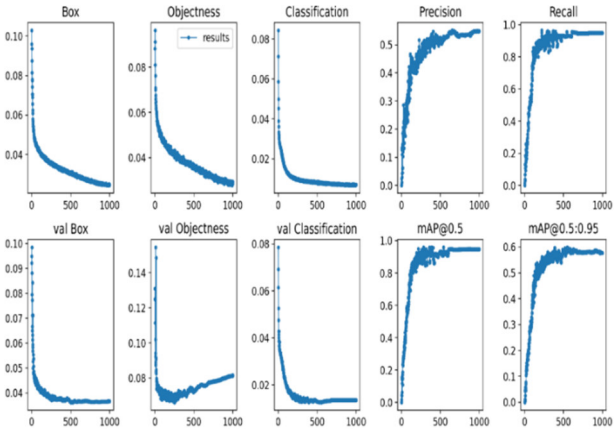

Fig. 9. The results of the Face recognition sub-system dataset training

Based on the results of the training as can be seen in Fig. 10, the highest value for precision is 0.5507 at 650 epochs. Thus, training for more than 650 epochs has experienced diminishing returns.

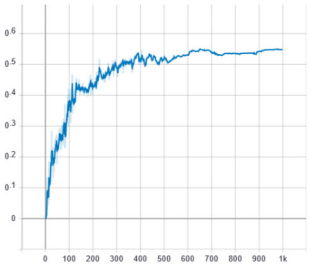

Fig. 10. Precision Face recognition sub-system

Based on the results of the training as can be seen in Fig. 11, the highest score for Recall is 0.9546 at 607 epochs. Thus, training for more than 607 epochs has experienced diminishing returns.

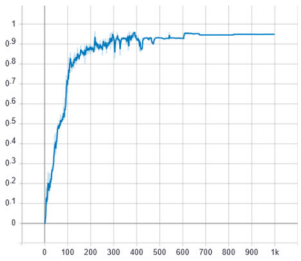

Fig. 11. Recall Face recognition sub-system

The method used to measure the Mean Average Precision (mAP) on multiple object detection for the face recognition sub-system is mAP@0.5 and mAP@0.5:0.95. The face recognition sub-system on mAP@0.5 has a high level of precision. The highest value as can be seen in Fig. 12, reached was 0.9884 in the 631st epoch and after that, the change was very small in the range of 0.02.

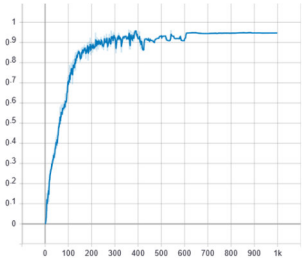

Fig. 12. mAP@0.5 Face recognition sub-system

The face recognition sub-system at mAP@0.5:0.95 has a lower precision level than mAP@0.5 because the prediction threshold is increased by 0.05 from 0.5 to 0.95. The highest value as can be seen in Fig. 13, reached was 0.5986 at the 546th epoch which then tended to decline and stabilize at the range of 0.5800.

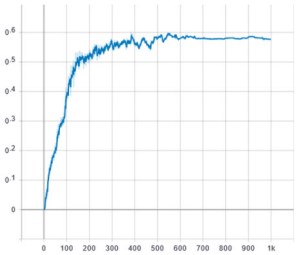

Fig. 13. mAP@0.5 :0.95 Face recognition sub-system

The mAP value indicates the average precision value is far above the threshold of 0.4 so that the resulting model is feasible to use.

Some of the Ground Truth of the Face recognition sub-system can be seen in Fig. 14.

Fig. 14. Ground Truth Face recognition sub-system

Some of the Ground Truth Augmented Training Data obtained from Machine Learning can be seen in Fig.15.

Fig. 15. Ground Truth Augmented Training Data Face recognition sub-system

Ground Truth compared to the results of object detection and found that the accuracy is quite good for large objects but on small objects, there is a tendency for the wrong detection or not to be recognized at all. The confidence threshold for the test is 0.4.

C. Visitor Identification System

The Attendance System library pulls information from the Face recognition sub-system provided via a web service. The information available through the API is retrieved using JavaScript which is then displayed on the visitor identification sub-system. The information that has been processed and presented on the visitor identification sub-system can be seen in Fig. 16.

Fig. 16. Visitor Identification System Output based on Web Service API

D. *Library Attendance System*

Library Attendance System testing is carried out on two media models, namely images and videos. Tests were carried out using a CUDA NVIDIA Tesla P100-PCIe-16GB with 16280MB of memory. The model used is the training model on the face recognition sub-system.

1) Image Object Detection

There are 39 images tested with more than one object in each image. Object detection on 39 images was carried out for 5.564 seconds, including the preparation process. The average object detection per image is 0.015 seconds. Some of the object detection results can be seen in Fig. 17.

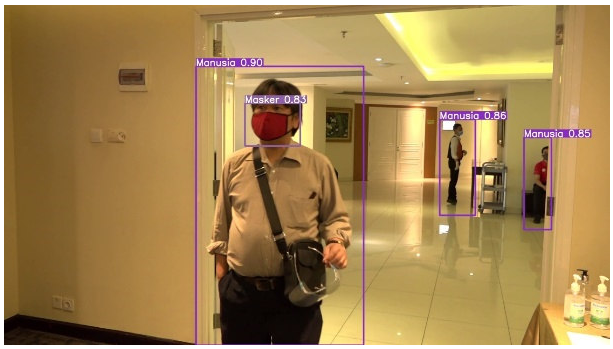

Fig. 17. Object Detection Results in Images

2) Video Object Detection

The video used for testing had a running time of 15 minutes 45 seconds. Detection of objects on 15 minutes 45 seconds of video with 23.623 image frames were carried out for 878.967 seconds, including the preparation process. The average object detection per image is 0.013 seconds. The number of frames per second (frames per second) is 76.92 fps which is faster than the standard 60 fps for video. Some of the object detection results can be seen in Fig. 18.

Fig. 18. Object Detection Results on Video

3) Library Attendance System Validation

Library Attendance System will validate the attendance for a confidence score of 0.9, the detection results will appear on the report page. The report page will display the data once even though the object is detected many times in one day. This is because the coding to save the database has been given a parameter not to save more than 1 cls at a time. The example of the report page can be seen in Fig.19.

Fig. 19. Library Attendance System Report

IV. CONCLUSION AND FUTURE WORKS

Library Attendance System is able to integrate three-part of sub-system which include a face recognition system based on YOLOv5. YOLOv5 face recognition subsystem which can detect multiple objects in images and videos has an excellent level of performance in terms of the average object detection time standard per image. A large number of face recognition systems with the YOLOv5 algorithm at the training stage took 3.736 hours for 1420 images with a total of 1000 epochs on a CUDA Nvidia Tesla T4 with 15079MB of memory. The mAP value indicates the average precision value is far above the threshold of 0.4 so that the resulting model is feasible to use. Detection of multiple objects with the YOLOv5 algorithm in the image has a good performance level, which is 0.14267 seconds. Detection of multiple objects with the YOLOv5 algorithm on video has a good level of performance, which is 76.92 fps. Further development for Library Attendance System, such as physical access control systems or security warning systems, all of which aim to assist library management, can be developed based on this study.

REFERENCES

[1] P. B. Babu and M. Krishnamurthy, *Library automation to resource discovery: a review of emerging challenges*. The Electronic Library, 31(4), 433–451, 2013.

[2] C. Hu, Y. Wang, G. Yu, Z. Wang, and A. Lei, "Embedding CNN-Based Fast Obstacles Detection for Autonomous Vehicles," Intell. Connect. Veh. Symp., pp. 1–11, 2018.

[3] C. Liu, Y. Tao, J. Liang, K. Li, and Y. Chen, "Object Detection Based on YOLO Network," 2018 IEEE 4th Inf. Technol. Mechatronics Eng. Conf., no. Itoec, pp. 799–803, 2018.

[4] E. Rudiawan, R. Analia, D. S. P, and H. Soebakti, "The Deep learning Development for Real-Time Ball and Goal Detection of Barelang-FC," 2017 Conf. Int. Electron. Symp. Eng. Technol. Appl., pp. 146–151, 2017.

[5] J. Redmon, "You Only Look Once: Unifed, Real-Time Object Detection", *Ann. Med. Psychol.* (Paris)., vol. 134 II, no. 1, pp. 59–61, 2016.

[6] M. S. Chauhan, A. Singh, M. Khemka, A. Prateek, and R. Sen, "Embedded CNN based vehicle classification and counting in non-laned road traffic," 2019.

[7] Glenn Jocher, "YOLOv5 in PyTorch," *Github*, 2020. [Online]. Available: https://github.com/ultralytics/yolov5. [Accessed: 08-Nov-2020].

[8] C. Y. Wang, H. Y. M. Liao, I. H. Yeh, Y. H. Wu, P. Y. Chen, and J. W. Hsieh, "CSPNET: A new backbone that can enhance learning capability of CNN," *arXiv*, 2019.

[9] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 10778–10787, 2020.