International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 15, No.1 (Feb-24)

https://dx.doi.org/10.12785/ijcds/150140

Lightweight Neural Network Design for Real-time Human Tracking with Visual Modality

Christopher Limawan[1] **and Benfano Soewito**[1]

[1]*BINUS Graduate Program – Master of Computer Science, Bina Nusantara, Jakarta, Indonesia*

Received 24 Apr. 2023, Revised 5 Jan. 2024, Accepted 21 Jan. 2024, Published 1 Feb. 2024

Abstract: We proposed a lightweight neural network architecture model that focuses on efficient computation when doing face detection in real-time with limited system resources, in image processing time to be exact. The concept is to implement layers and neurons as minimum as possible to reduce processing time and computing resource used in either convolutional layer or fully connected layer. This research is conducted because current neural network technology does not consider real-time detection scenario needs, such as tracking an object using camera. The result of proposed neural network implementation is an application that captures video from camera and generates boundary box that contains human face, with 0.117 seconds processing time each data, 96.735% accuracy, and 0.1219 error rate. The proposed model used 169.3 MB RAM and taking up to 1.186% CPU processing. Although does not have best accuracy and error rate, the proposed method does have faster processing time and lower usage of system resources in human face detection. This will allow computers with lower specification to use this model for face detection, especially in human tracking. This research also provides the concept of convolutional neural network, object tracking, and face detection that would be the base of create the proposed lightweight neural network.

Keywords: Neural network, Face detection, Object tracking

1. INTRODUCTION

Coronavirus Disease 2019 (COVID-19) pandemic, which happened around year 2020 to 2021, forced each person to change methods in doing their usual activities regarding how each person should do their activities [1]. Before the pandemic starts, each person performed their activities and met with participants, friends, and colleagues by ”face to face” method in a special room prepared. When the pandemic starts, each person still could perform their activities, but cannot meet with the participants, friends, even colleagues by ”face to face” method because the disease is highly infectious and contagious. Despite of dangers of the pandemic, there are several meets which could not be postponed in a long time and need participants because it does need support from another participant(s), such as financial, administration, managerial support.

Because of this matter, there are some activities that supported by online meeting to cover up the distance between each of participants, including education sector. For starters, in 2021, teaching and learning activities in school has been done in online meeting, imitating what teacher and students could have done in onsite classroom. Open video to show face of the participant, open microphone to speak, and sharing screen or whiteboard are some of the activities that teacher and students have done in online meeting for teaching and learning activities which imitated teaching and learning activities in onsite classroom. Although different activities from usual activities that have been done in classroom, the questionnaire that have been gathered shows high satisfaction of using online meeting to imitate those activities in classroom [2]. The questionnaire result shows that despite of different activities has been done in online classroom, students and teachers just need minor adjustments in terms of teaching and learning activities and the activities would be done without major problems.

In teaching and learning implementation in online classroom, current technology has shown that current trend of camera, which is pan-tilt-zoom camera, have an interesting approach. Teachers could use the camera to show the prepared onsite whiteboard for write any details to support the theory or the teacher’s claim. Teachers that used the camera can move the camera either to the teacher’s position or to the whiteboard’s position. Unfortunately, the camera cannot follow the teacher’s position, caused the teacher must move the camera manually. To resolve this matter, a system to track teacher’s movement based on specific modality is needed. Create and implement neural network model to track teacher’s movement based on human face modality is one of the solutions to achieve this goal, but the model itself should be usable in real-time, because the system will

E-mail address: christopher.limawan@binus.ac.id, bsoewito@binus.edu

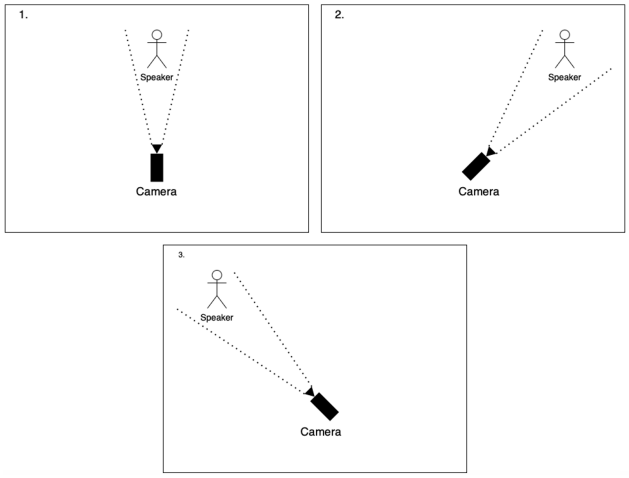

Figure 1. Human tracking process scenario using camera

continuously using the model to detect human face and interact with the camera as to tell the camera where should the camera move.

There are several studies that have similar goal and we referred those studies as our study reference to design the optimal neural network model. Research from [3] studied about speaker tracking using mouth detection to track face and Global Coherence Field (GCF) to track the source of the sound, while added discriminative and generative model in the algorithm to detect mouth. Meanwhile, [4] studied about tracking person using PAFIM method and microphone arrays, also implemented it in a drone. [5] implemented expectation-maximization for speaker position detection, also implemented in Popeye robot head, while [6] using multi-modal perception attention network for speaker tracking based on audio and visual modality. [7] tracked an object using Convolutional Neural Network based on audio and visual modality, that will be implemented in Robotics.

Based on mentioned studies, the problem is there are no research that focuses on creating a lightweight neural network, because the focus from those studies is to get a good accuracy. Meanwhile, the goal is to do a real-time object tracking, also the camera or the object that provides "vision" to the system does have additional algorithm related to object tracking, such as the camera follows the object. It means that other than good accuracy, the model does need faster processing time.

This research of lightweight neural network focuses on develop specific neural network architecture in purpose of faster processing time when detecting position of specific human. To understand the reason how faster processing time is important to object detection, have a look at Figure 1.

Let us assume that the camera has been programmed to follow the human's position. It means if the human moves outside camera's field of view and to the right, the camera will follow the speaker's movement to the right (from image number 1 to image number 2 in Figure 1). When the speaker moves to the left, the camera will follow the speaker's movement to the left (from image number 2 to image number 3 in Figure 1).

To make the camera follows human's movement, the system should extract features in the captured frame, then classify it whether the target is the human or not. To know whether those features in captured frame contains the speaker, we can implement either machine learning algorithm or neural network for classification. After classifying it, the system can decide the human's position, whether the position is at the edge of camera's field of view or not. When the position is at the edge, the camera will adjust the field of view, make the human's position to be center of camera's field of view.

The main problem lies in the classifying process, as the classifying process could take too much processing time from feature extraction until classifying the captured frame. This will cause the system has a noticeable delay and adjustment of camera's field of view angle will have a noticeable delay. This is the main reason why faster processing time is important, to make camera's field of view angle updated in real-time. Although accuracy is important to human detection, faster processing time is the most important in real-time human tracking.

Another problem in recent studies is the studies does not consider the computing resource that will be used as to classify the object. Let us assume the neural network will be implemented in small computer that receives input from camera. If the computer uses low to standard computing resource, this will cause slow processing image when received image from camera. If user need to classify and track object without proper quality of computing resource, this will cause slow execution on next algorithm, such as following and tracking person with malicious intent. Those are the main reasons why our focus of this research is creating a lightweight neural network architecture model.

The contributions of this research will be as follows: (i) neural network architecture design that will be lightweight to do human tracking for low to standard computing system resource, with visual modality using face detection and (ii) implementations of neural network to pan-tilt-zoom camera, combined with human tracking algorithm using naïve method.

2. RELATED WORKS

In this part, we will look on what have researchers implemented for each works, related to neural networks and object tracking in the past.

A. *Convolutional Neural Network*

Different than usual neural network, convolutional neural network's purpose is to do feature extraction by using arithmetic calculation and specific formula [8]. After doing

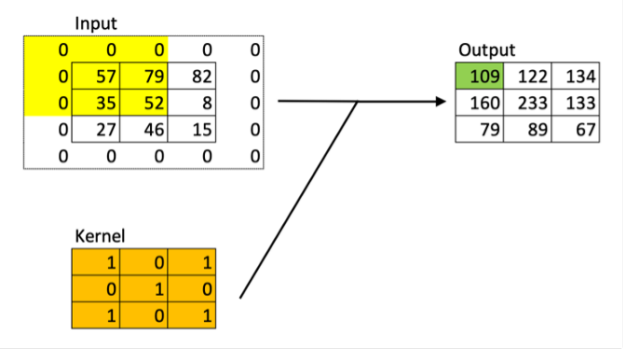

Figure 2. Convolution layer process to calculate new feature map based on calculation from array of values in input and specified kernel

Figure 3. Pooling layer process to calculate new feature map based on maximum value of specific area in feature map

feature extraction, the value will be processed in fully connected neural network. The result of fully connected neural network will be used in various requirements, for example multi-class classification, multi-label classification, and regression. Convolutional neural network also comes from feed-forward neural network. The architecture of convolutional neural network consists of: (i) convolution layer, (ii) pooling layer, and (iii) fully connected layer.

Convolution layer's target is to get a feature map from current feature map in input data such as images or texts [9]. To get the new feature map, convolution layer will calculate the result of dot product from current feature map and prepared kernel, as explained in Figure 2. The purpose of current feature map is for normalize the weight with the value that wrapped around it.

After convolution layer, specific areas of the feature map will be simplified to get the value representation for specific area [9]. There are 2 methods for pooling the specific area, which is get the max value (max pooling, have a look at Figure 3) or get the average value of specific area (global average value, have a look at Figure 4). Specification of width and height of area that will pooled are specified in the model.

Depends on the architecture of model, process of convolution and pooling layer could be executed several times with different specifications [9]. After those process, the values in feature map will processed in fully connected layer to do usual calculation in artificial neural network, also the values will be processed in activation function to get the normalized value based on the formula and rules in

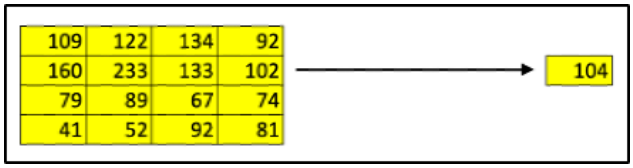

Figure 4. Pooling layer process to calculate new feature map based on average value of specific area in feature map

the specified activation function. From there, the output of fully connected layer is the value representing the class or label of input.

An example of convolutional neural network implementation for object tracking is in a form of attention network. Study from [6] uses attention network for speaker tracking, with audio and visual modality as the input. As for simple technique as convolutional neural network itself, study from [7] uses convolutional neural network of their own neural network architecture to do object tracking. Similar to [6], [7] uses audio and visual modality as input to locate the object. Meanwhile [10] splits the process of face detection in 3 stages using convolutional neural network, consists of proposal network (P-net), refinement network (R-net), and output network (O-net). [11] introduces max-feature-map operation in their convolutional neural network to normalize large dataset that contains noisy labels. Meanwhile, [12] used depthwise separable to enable faster process and calculation in convolution layer, then the model with depthwise separable attached in the edge device that contains camera. In the same topic of research, [13] added Kernelized Correlation Filter (KCF) to adjust parameter used in neural network that contains depthwise separable, which makes faster detection because of delay when the camera did not detect any face in its field of view.

Compared to recent studies above, majority of the studies do not evaluate their model based on processing time to classify specific feature, except study from [12] and [13]. Processing time does matter when the model used for real-time detection for decreasing delay time when detect specific object. For that reason, our study will evaluate the neural network model based on processing time when classify face in given image.

B. Object Tracking

In computer vision, we can get what type the object shows in image using image classification [7]. But when we want to know if the target object detected in image existed, we can use object detection. In computer vision, object detection is implemented in several machine learning model, such as Viola Jones and HOG. Object detection is not limited to machine learning model only because there are several neural network models such as Faster R-CNN and Single Shot Detector [14].

Object detection do open some new techniques in computer vision related to image, one of them is object

tracking [7]. Object detection can only mark the position of the object, but when the object and the position of mark are moved simultaneously and feels like the mark position follows the object, that is object tracking. Usually, this technique is applied in video-based dataset. To apply the object tracking, the methodology is similar to object detection, with addition of update the object mark position for each frame in video. So, machine learning model and neural network model mentioned above, which used in object detection, are capable to do object tracking.

One of current research related to object tracking implemented in development of You Only Look Once (YOLO) [15], whereas the model itself focuses on fast object detection (decreased the layer number of convolutional layers). On the other side, Fast R-CNN [16] generates several regions of interest using convolutional layer and will be pooled in pooling layer to classify each class within generated region of interests. Inspired from Fast R-CNN, region proposal network (RPN) created to create Faster R-CNN [17] and to make the model usable for real-time object detection. Compared to YOLO and Fast R-CNN, Single shot multi-box detector (SSD) [18] uses several convolutional layer results to detect specific object in specific boundary box in a single processing of neural network, which makes it more accurate than YOLO and faster than Fast R-CNN.

Related to object tracking, usually the modality to decide whether the input contains specific object is based on image or frame in video. But in [6] and [7] studies, they implemented audio modality to track object besides frame in video. Different with recently mentioned research, [4] using sound modality only in drone to track object with sound source tracking method. Meanwhile, [19] used sound and visual modality to track human face. In different topic, [20] implemented cross entropy to neural network, in purpose of extracting road construction and path from satellite.

Different with several studies, recent studies have set their goal to find optimal accuracy and error loss. In our study, our focus is to design neural network architecture model with optimized amount of layer to get optimal processing time result, because each layer does calculation to specify feature that exists in given image, which makes longer processing time with bigger amount of layer. Our neural network architecture goal is to detect face feature in specific image because we want to do benchmark whether the model is optimal for real-time usage based on accuracy, error rate, processing time, and resource taken in system.

C. Face Detection

In computer vision, face detection can decide whether the image contains human face or not [21]. This will be the basis before doing object tracking because object tracking will need specific object that need to be detected in specific location of image. To detect human face in image, we need either skin color, motion, gray scale, and edges that will represent the existence of human face in image. One of those features or more will be the input of the specific model that will be used to recognize the existence of human face in image. In face detection based on skin color, it uses either red, green, and blue (RGB) format values or another format related to colors such as hue, saturation, and intensity value (HSV) to recognize skin color on human face. Meanwhile, face detection using motion will check between 1 frame before and 1 frame after, which will be the basis of moved objects that will be declared as human face. In face detection using gray scale, it will convert current image or frame in video to gray intensity each pixel to detect specific combinations of gray scale area as face. Lastly, in face detection using edges, it will detect edges that is visible from images or frame in video, which will be processed whether the combination of edges contains human face.

In current progression of research that related to face detection works, [22] proposed DSFD model, which using 2 results contained low-level features and high-level features to combine the result and detect human face from combined result. Meanwhile, [23] proposed DPSSD, which focuses on detect human face, especially tiny human faces in image.

3. PROPOSED METHODOLOGY

Based on experimenter hypotheses, not all of convolutional neural network designs are applicable in edge devices, especially with low specification. The reason behind this experimenter hypotheses is we need light processing, also accurate detection if we want to implement the detection system in edge device. Based on the hypotheses, the experiment will focus to develop lightweight neural network that can be used for real-time object tracking in edge device, which needs update of specific object's position each second.

A. Experimentation Procedures

To create optimized neural network architecture model which has light processing and accurate detection, we have defined our experimentation procedures in Figure 5. First, we gathered a collection of images for training and testing images in face detection. The classification method is face detection because the model would detect the position of face features, which will enable pan-tilt-zoom camera to follow the face's position based on face features' position. Collection of images will be gathered as a dataset. Before the dataset used for training the neural network architecture model, the images in dataset are preprocessed to create another combination of images, which would make the model recognized the face features despite of various positions and image quality.

After preprocess the images in dataset, we designed the optimized neural network architecture model which has light processing and accurate detection based on recent studies as our study references. In this step, our goal is to detect face features based on prepared dataset. But, creating an optimized neural network architecture model would only detect whether the image contains face or not. Because of that reason, our next step is to implement

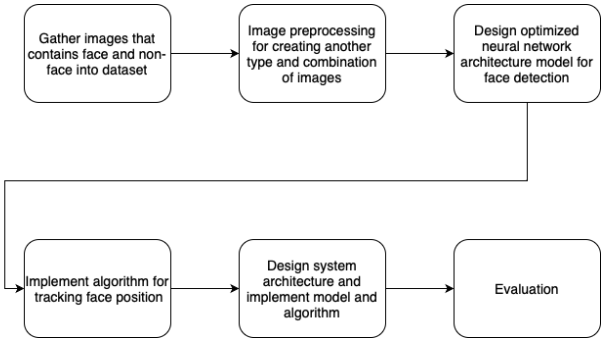

Figure 5. Experimentation procedures to create lightweight neural network architecture model

an algorithm to decide the position of the face. After algorithm implementation, the model and algorithm will be implemented in prepared system design and will be evaluated.

B. Dataset

The dataset itself is split into 2 categories, human face and no human face. The dataset of human face is originated from Labeled Faces in the Wild (LFW) dataset, which contains 13.233 images that consist of cropped face for each image in each sequence. Other than LFW dataset, experimenter added images of human face from Audio-Visual 16.3 corpus, using seq08, seq11, and seq12 recording sequences. Experimenter cropped all human faces in those recording sequences manually, resulting 2.547 images added in dataset and marked as "human face" image category. Meanwhile, "no human face" image category originated from Kaggle, Natural Images dataset. The dataset contains 6.899 images consists of all type of images, included images with human face. The experimenter excludes all images with human face in Natural Images dataset, which resulting 5.913 images that does not contain human face. Total of images used for train proposed model is 18.982 "human face" images and 11.826 "no human face" images. Then, experimenter splits the dataset into 80% training images and 20% testing images.

C. Image Preprocessing

Before train model in proposed neural network architecture to detect "human face" in given images, for each given image in dataset will be resized into 224 x 224 pixels. Then, those given images will be cropped into 128 x 128 pixels size to train the model about human face that shows half of the human face or three quarters of human face. After that, those given images will be flipped horizontally to train the model about human face that faces left and right side.

D. Lightweight Neural Network Architecture

The lightweight neural network is a neural network, which consists of 7 layers of generic convolution neural network and 1 prediction layer consists of 2 layers of fully connected layer. The sequences of the neural network are: (i) 2-dimension convolution layer to 48 neurons with 3x3 kernel, added with pooling layer with max pooling 2x2 kernel, (ii) 2-dimension convolution layer to 64 neurons with 3x3 kernel, added with pooling layer with max pooling 2x2 kernel, (iii) 2-dimension convolution layer to 64 neurons with 3x3 kernel, added with pooling layer with max pooling 2x2 kernel, (iv) 2-dimension convolution layer to 64 neurons with 3x3 kernel, added with pooling layer with max pooling 2x2 kernel, (v) 2-dimension convolution layer to 64 neurons with 3x3 kernel, added with pooling layer with max pooling 2x2 kernel, (vi) 2-dimension convolution layer to 32 neurons with 3x3 kernel, added with pooling layer with max pooling 2x2 kernel, and (vii) 2-dimension convolution layer to 32 neurons with 3x3 kernel, added with pooling layer with max pooling 1x1 kernel. The last layer of prediction layer consists of fully connected layer consists of 64 neurons, output will be in 1 neuron, consists of class probability of detected object that declares whether the image contains human face or no human face. This neural network architecture is inspired from [7] and [11] research of neural network architecture.

In neural network layers, we need 2-dimension convolution layer to extract image features from captured frame, which will be in form of 2-dimension image. Then, each convolution layer result will be processed in max pooling layer, as several areas of pixel will be represented as a value. The proposed generic convolutional neural network is built in 7 layers, processed in sequential order, as the features from recent layers will be processed using mathematical formula(s), producing detailed features that comes from captured frame.

Regarding the neural network usage for human face classification, the neural network usage scenario will be each batch of image(s) will be processed in raw, which still contains RGB value. The reason behind keeping the image(s) in raw values is because deciding factor of human face depends on skin color base, supported by research from [21]. Each value in RGB will be treated as input that will be processed in input layer. Then those value will be processed and calculated based on neural network architecture that has been specified earlier in Figure 6. Last layer of hidden layer (which is 64 neurons) will be processed, resulting a value in output layer, representing binary output between "human face" and "no human face" class.

Another information that will be used in lightweight neural network is using stochastic gradient descent optimizer and cross entropy loss function, also when trained the model, the model will be trained in 100 epochs. When trained the model, each batch size will be set 64 images per batch to train the model. Learning rate that will be used in this research is 0.01. Learning rate will be adjusted using cosine annealing [24] for each epoch and restarted every 10 epochs. Before model training starts, weights for each neuron in proposed method are initialized using LeCun initialization.

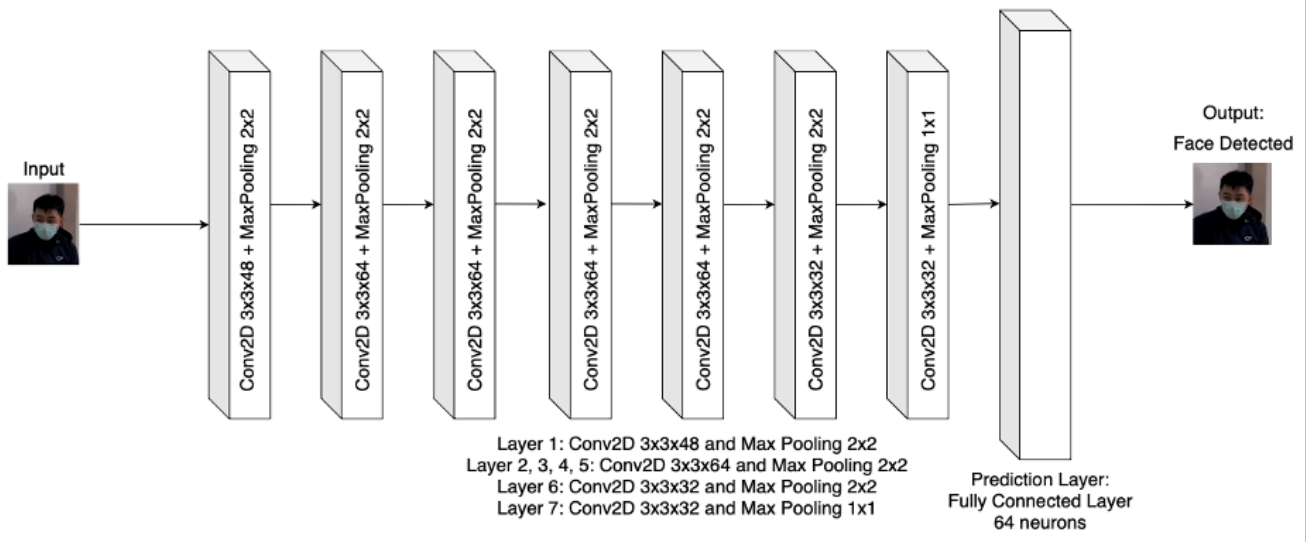

Figure 6. Proposed model of convolutional neural network architecture

E. Algorithm for Human Tracking

In general, experimenter implemented neural network in specific flow to detect face from captured frame, which comes from camera. From Figure 7, the algorithm begins from capturing current frame in camera. Then, the frame will be divided into several sub-frames because the human face in video could be in small images, which is possible for the system to not recognize the captured frame as image that contains human face. When the human face is detected within sub-frames, the system will create the boundary box, declares which part of frame that contains human face.

To implement the designed flow, experimenter chooses to use naïve method of tracking. As specified in Figure 8, after the frame captured by video, the frame will be checked using sliding window algorithm. Each specific area of frame will be processed by neural network and will be validated whether the specific area of frame contains human face. If human face is detected in specific area of frame, the algorithm will declare that in the specific area of frame contains human face. After that, the sliding window algorithm will be stopped.

To mark which frame contains human face, when splitting frames into several sub-frames, the algorithm will save both sub-frame and frame's coordinates in pixel to get the same sub-frame. Before the current captured frame is updated, the algorithm will draw boundary box based on frame's coordinates to get specific sub-frame that contains human face.

F. System Design Architecture

When implementing neural network to system, neural network will be implemented in a form of simple application that uses Figure 8 as the basis of face detection and human tracking, as illustrated in Figure 9. The application will be installed in desktop computer. To make the implementation works, camera will be attached using wire to desktop computer. The camera will be the system that captures frame in video and provide the frame to application. Then, the camera will be placed to have a field of view contains the human. When speaker moves from his/her position, the camera follows the human movement, which makes camera's field of view still contains the human figure.

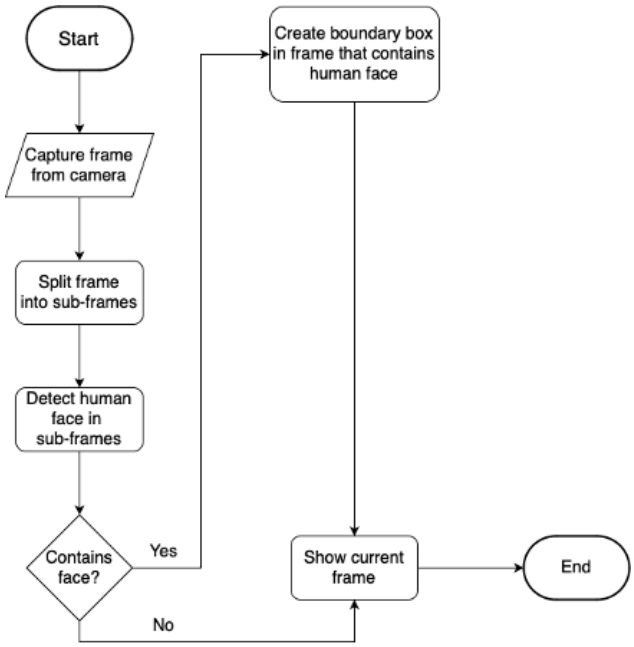

Figure 7. Flowchart of algorithm and neural network implementation to detect face from camera

```
function trackHumanFace()
slide_x = 256
slide_y = 256
while True do
frame = videoCapture()
sliced_frames = sliceFrames(slide_x, slide_y)
foreach sliced_frame ∈ sliced_frames do
        start_time = getCurrentTime()
        result = detectFace(sliced_frame)
        end_time = getCurrentTime()
        if result contains face
time_diff = end_time – start_time
print time_diff
coordinates = getFrameCoordinates(sliced_frame)
            x = coordinates[0]
            y = coordinates[1]
            max_x = coordinates[2]
            max_y = coordinates[3]
createBoundaryBox(frame, x, y, max_x, max_y)
            break
        end if
end foreach
showImage(frame)
end while
```

Figure 8. Algorithm of human tracking in video

G. Evaluation

Before evaluating proposed method performance to models from another research, proposed method performance will be compared to another defined models with modified neural network architecture. Creation of modified neural network architecture for performance comparison were created based on convolutional layer quantity modification. Metrics that used for performance comparison are loss value, accuracy, and time needed to do face detection.

After doing performance comparison between proposed method and modified models, experimenter evaluates the lightweight neural network to models from another research. Experimenter used loss value, accuracy, time needed to do face detection, RAM used, and CPU used. The proposed method will be compared to LightCNN with 29-layers [11], LightCNN v2 with 29-layers [11], LightCNN with 9-layers [11], YOLO v3 [25], and SSD [18], in accuracy, error rate in loss function, data processing time performance, RAM used, and CPU used.

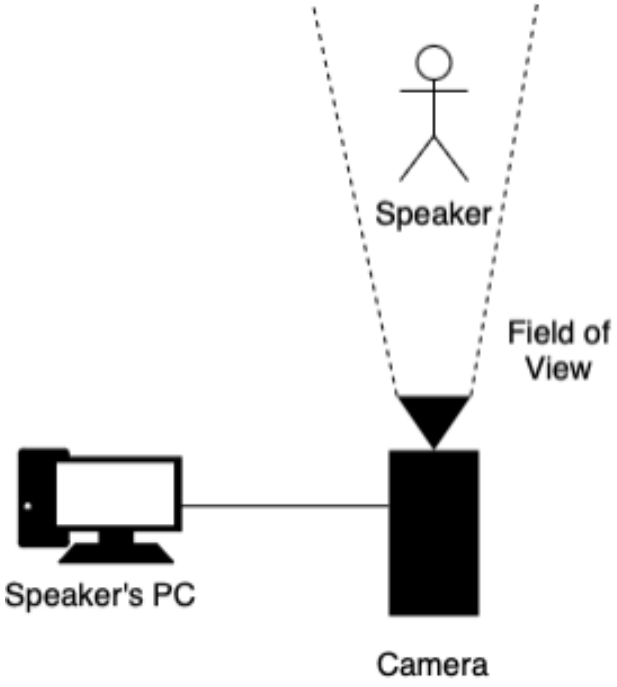

Figure 9. Physical architecture of human tracking implementation using camera

To get current accuracy of face detection, the method is the same with classify whether the frame contains face, as described in Equation 1. The principle is to get total of true positive and true negative results, then divided by total of tests conducted (total of true positive, true negative, false positive, and false negative). True positive (TP) means the model detected a face in image and the image is registered as image that contains face, whether true negative (TN) means the model does not detect a face in image and the image is not registered as image that contains face.

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

To calculate error rate using loss function, loss function that have been used for this research is cross entropy loss function. The reason behind using cross entropy loss function is because the detection model is just detecting between 2 categories, which is human face and not human face. In Equation 2, the distribution of numbers in p and q could be checked using loss function. The higher the error rate value, then the model cannot differentiate between inputs.

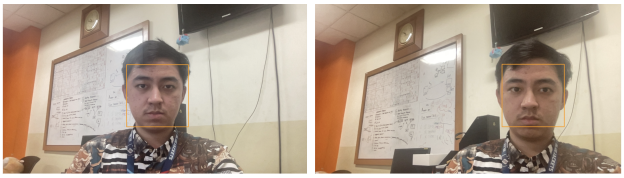

Figure 10. Proposed method implementation result, captured from camera and processed using proposed model

TABLE I. Performance experiment results of proposed method

Experiment No.	Data Processing Time (in seconds)
1	0.0427
2	0.0387
3	0.0657
4	0.0636
5	0.063

$$Loss = -\Sigma_k p_k \log q_k \quad (2)$$

To calculate data processing time performance, start and end time of face detection will be marked. If the captured frame does not contain face, start time and end time calculation will be ignored, since there is no related object as benchmark's main target, as specified in Figure 8. Otherwise, the start time and end time will be calculated. In Equation 3, the difference between end and start is the same as simple subtraction between end and start time.

$$\Delta t = t_{end} - t_{start} \quad (3)$$

4. RESULTS AND DISCUSSION

A. Implementation Results

From implementation results in using proposed neural network architecture, orange-colored border is the mark where the face is located, as shown in Figure 10. When in movement of human face, the neural network had been sent a new frame from captured video and checked whether the position of human face is changed or not. After running the neural network several times, based on Table I, data processing time between frames does not take more than 0.1 seconds.

B. Accuracy Statistics between Training and Validation Phase

From Figure 11, experimenter checked the performance of neural network between training and validation dataset using accuracy. As we can see in the accuracy statistics, the proposed model accuracy shows high decreasing accuracy in 38th epoch. But after 38th epoch, the decreasing accuracy is still there when trained the data, but the decreasing accuracy numbers are minimized. Last accuracy value of training phase is 97.14% and validation phase is 96.32%.

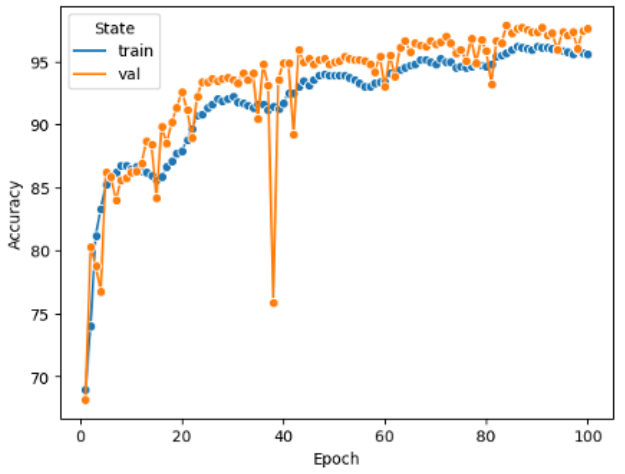

Figure 11. Training and validation accuracy statistics

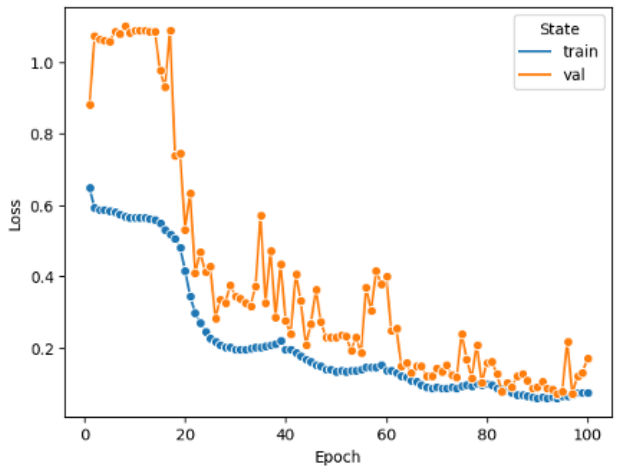

Figure 12. Training and validation error rate value from loss function statistics

C. Loss Value Statistics between Training and Validation Phase

From Figure 12, experimenter checked the performance of neural network between training and validation dataset using loss value. As we can see in the loss value statistics, validation loss values show high loss value with value more than 0.4 until 60th epoch. But after 60th epoch, validation loss values kept in between 0 to 0.3. Last error rate value of training phase is 0.0737 and validation phase is 0.1701.

D. Data Processing Time Statistics in Training Phase

From Figure 13, experimenter checked the performance of neural network between training and validation dataset from time taken for processing data. From the statistics, we can see that there are still fluctuations of validation's data processing time from 80th epoch and more, but the process kept in between 0.11 to 0.13 seconds.

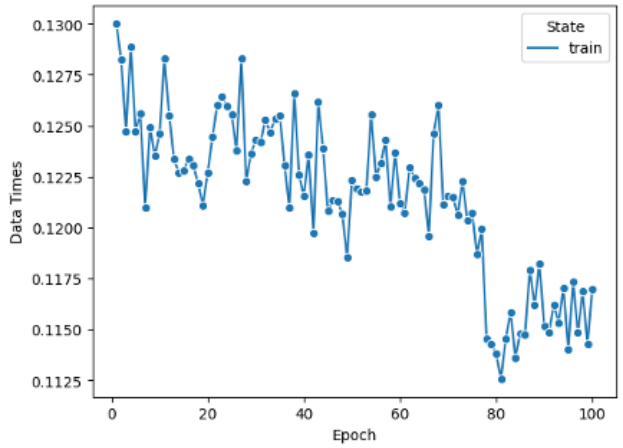

Figure 13. Training and validation data processing time statistics

E. Evaluation Compared to Another Model with Modified Quantity of Convolution Layers

In evaluation phase between proposed method and another model with modified quantity of convolutional layers, experimenter modified quantity of convolutional layers from Figure 6 and did performance comparison between proposed method. Models that have modified and used for performance comparison are specified in Table II. Each defined models in Table II will be processed in 2 fully connected layers in purpose to classify image between "human face" and "no human face". In Table II, there are 5 neural network models that will be tested for performance comparison, which are model with (i) 3 convolution layers, (ii) 4 convolution layers, (iii) 5 convolution layers, (iv) 6 convolution layers, and (v) 8 convolution layers.

The performance comparison each model in Table III shows that model number (iii) and proposed method have the same number of accuracies, but different loss value and data processing time. Model number (iii) showed decreased loss value but slower data processing time. Meanwhile, proposed method showed higher loss value and faster data processing time.

Another fact worth mentioning is data processing time for model number (v) is higher than proposed method. If we analyze data processing time result from model number (i) to (iv) and proposed method, increasing convolution layer means decreasing data processing time, which would conclude model number (v) would be shown faster data processing time.

F. Evaluation Compared to Another Research

In evaluation phase between proposed method and another research, experimenter used pan-tilt-zoom camera and attached the camera to the computer that detects human face. From Table IV, experimenter compared the performance between proposed method and other research. Based on the results, even though does not have better accuracy and loss value, but the method has a notably fast data processing time. In accuracy metric, LightCNN with 29-layers has better accuracy when trained with dataset of Natural Images, Labeled Faces in the Wild, and Audio-Visual 16.3 corpus. Although accuracies of LightCNN models are higher than proposed method and error rate from loss function are lower than proposed method, proposed method did have faster data processing time when using sliding window method to detect the location of face.

From Table IV, experimenter recorded another result from system performance approach, which was comparing system performance usage between proposed method and another studies. Based on the results, proposed method used fewest memory and CPU resources than other studies. While other studies used memory resources more than 200 MB, the proposed method can suppress used memory resources to below than 200 MB. CPU usage of proposed method did show fewer usage than other studies, with small difference of 0.02% usage.

G. Discussion

From the experiment results, there are 2 things that we should be elaborated further: (i) performance comparison between proposed neural network architecture to modified models and (ii) performance comparison between proposed neural network architecture to another research. Model with modified quantity of convolution layers showed accuracy between 94% to 97%, loss value between 0.08 to 0.15, and data processing time between 0.11 to 0.13 seconds. From performance result shown in Table III, model number (iii) and proposed method showed better performance than another defined model. But to decide which model does have the best performance in performance comparison, experimenter used specific formula in Equation 4 to represent each result shown in Table III to effectiveness value of specific model.

$$M^i_{effective} = 0.3(A^i_{train} + A^i_{val}) - 0.3(L^i_{train} + L^i_{val}) - 0.4D_i \quad (4)$$

The formula calculates between accuracy in training and validation (in formula, the variable are represented using A variable), loss value in training and validation (in formula, the variable are represented using L variable), and data processing time in training phase (in formula, the variable represented using D variable). Data processing time had higher constant than the accuracy and loss value constants because the purpose of this experiment was to find faster processing data in terms of face detection. Because of that reason, 0.4 had been set as data processing time's constant.

Another fact in Equation 4 is lower value of loss value and data processing time means that the model had better performance compared to another model. Because of this reason, loss value and data processing time are processed to negative constant (-0.3 for loss value and -0.4 for data processing time). Meanwhile, higher accuracy means that the model had better performance compared to another

TABLE II. Another defined models with modified quantity of convolution layers for performance comparison

Model No.	Neural Network Model Architecture
(i)	Convolution layer 1: Conv2D 3x3x48 + MaxPooling2D 2x2 Convolution layer 2: Conv2D 3x3x64 + MaxPooling2D 2x2 Convolution layer 3: Conv2D 3x3x32 + MaxPooling2D 1x1
(ii)	Convolution layer 1: Conv2D 3x3x48 + MaxPooling2D 2x2 Convolution layer 2: Conv2D 3x3x64 + MaxPooling2D 2x2 Convolution layer 3: Conv2D 3x3x64 + MaxPooling2D 2x2 Convolution layer 4: Conv2D 3x3x32 + MaxPooling2D 1x1
(iii)	Convolution layer 1: Conv2D 3x3x48 + MaxPooling2D 2x2 Convolution layer 2: Conv2D 3x3x64 + MaxPooling2D 2x2 Convolution layer 3: Conv2D 3x3x64 + MaxPooling2D 2x2 Convolution layer 4: Conv2D 3x3x32 + MaxPooling2D 2x2 Convolution layer 5: Conv2D 3x3x32 + MaxPooling2D 1x1
(iv)	Convolution layer 1: Conv2D 3x3x48 + MaxPooling2D 2x2 Convolution layer 2: Conv2D 3x3x64 + MaxPooling2D 2x2 Convolution layer 3: Conv2D 3x3x64 + MaxPooling2D 2x2 Convolution layer 4: Conv2D 3x3x64 + MaxPooling2D 2x2 Convolution layer 5: Conv2D 3x3x32 + MaxPooling2D 2x2 Convolution layer 6: Conv2D 3x3x32 + MaxPooling2D 1x1
(v)	Convolution layer 1: Conv2D 3x3x48 + MaxPooling2D 2x2 Convolution layer 2: Conv2D 3x3x64 + MaxPooling2D 2x2 Convolution layer 3: Conv2D 3x3x64 + MaxPooling2D 2x2 Convolution layer 4: Conv2D 3x3x64 + MaxPooling2D 2x2 Convolution layer 5: Conv2D 3x3x64 + MaxPooling2D 2x2 Convolution layer 6: Conv2D 3x3x64 + MaxPooling2D 2x2 Convolution layer 7: Conv2D 3x3x32 + MaxPooling2D 2x2 Convolution layer 8: Conv2D 3x3x32 + MaxPooling2D 1x1

TABLE III. Performance experiment results of proposed method compared to another defined model in Table II

Model	Accuracy	Error Rate	Data Processing Time (in seconds)
(i)	95.455%	0.1295	0.1302
(ii)	95.815%	0.12275	0.1271
(iii)	96.735%	0.09465	0.1256
(iv)	94.67%	0.1438	0.1187
7 convolutional layers (proposed method)	**96.735%**	**0.1219**	**0.117**
(v)	95.8825%	0.14455	0.1257

TABLE IV. Performance experiment results of proposed method compared to another research related to convolutional neural network

Model	Accuracy	Error Rate	Data Processing Time (in seconds)	Memory Used	CPU Usage
Proposed method	96.735%	0.1219	**0.117**	**169.3 MB**	**1.186%**
LightCNN 9-layers [11]	99.83%	0.0084	0.1197	219.6 MB	1.207%
LightCNN 29-layers [11]	99.95%	0.0025	0.1981	271.8 MB	1.372%
LightCNN v2 29-layers [11]	99.90%	0.0051	0.3689	261.6 MB	1.598%
You Only Look Once (YOLOv3) [25]	99.95%	0.00011	0.6644	1.6 GB	2.788%
Single Shot Detector (SSD300) [18]	99.95%	0.08301	0.2108	379.2 MB	1.826%

TABLE V. Performance calculation result, represented using effective value from Equation 5

Model No.	Effective Value
(i)	-0.9796
(ii)	-0.2911
(iii)	0.6936
(iv)	-0,1039
7 convolutional layers (proposed method)	**1.4983**
(v)	-0,81735

model. Because of this reason, the accuracy result processed to positive constant (0.3 for accuracy).

Before the accuracy, loss value, and data processing time, those values will be processed using Equation 5 to normalize minimum and maximum values, transformed to be z-score value for each result. The calculation of z-score are needed because accuracy, loss value, and data processing time had different minimum and maximum value.

$$z^{i}_{score} = \frac{v_i - \bar{v}}{\sqrt{\frac{1}{n-1}\Sigma_j = 1^n (v_n - \bar{v})}} \tag{5}$$

In Table V, proposed method and model number (iii) had positive effective value, meanwhile model number (i), (ii), (iv), and (v) had negative effective value. To find model with better effective value, experimenter seek the highest effective value in the performance calculation result. Proposed method performance calculation result had been shown as the highest effective value, compared to another models.

The next problem is about performance comparison between proposed neural network architecture and another research. Based on experiment to change neural network model, the difference between given many neurons and few neurons in fully connected layer is on how "detail" the model should know the signature of specific face. It caused the model with fewer neurons in last layer shows lower accuracy than more neurons in last layer. The number of neurons and layers in neural network also affect the results of neural network phases.

In recent studies, LightCNN models are splitted into 3 models, which is 9-layers type, 29-layers type, and 29-layers type without max pooling each layer. Meanwhile, YOLOv3 used Darknet-53 model as base model, added with 53 additional layers for object detection. SSD300 used VGG16 model as base model, added with 7 layers for object detection. But proposed method used only 7 layers, added with naïve method to split captured frame into sub-frames.

Although lower accuracy and higher error rate compared to another research, fewer layers and neurons mean faster data processing. This methodology proven that proposed method did have faster processing time, also fewer resources used, whereas LightCNN that have more layers (model with 9 layers and 29 layers) appear to be slower than proposed method (model with 7 layers). Compared to popular models of several studies such as YOLO and SSD300, the proposed method did have faster performance with effective use of system resources with memory usage lower than 200 MB and CPU usage lower than 1.5%. Because of the result, implementation of proposed method to edge device is possible with the minimum requirement of 512 MB memory and edge device processors.

5. CONCLUSION

In this paper, experimenter proposed a convolutional neural network that has been designed to be as lightweight as possible, with keeping accuracy higher than 95% and loss less than 0.4. The experiment's result is the proposed convolutional neural network design does have faster data processing time (0.117 seconds) and effective use of system resources (169.3 MB RAM and 1.186% CPU usage) in data processing when using proposed method. In the future, the experimenter will optimize region proposal network and implement the optimized method to this model and attached it to CCTV, which will track human in real-time.

REFERENCES

[1] T. K. Nguyen and T. H. T. Nguyen, "Acceptance and use of video conferencing for teaching in covid-19 pandemic: an empirical study in vietnam," *AsiaCALL Online Journal*, pp. 1–16, 2021.

[2] T. H. Fatani, "Student satisfaction with videoconferencing teaching quality during the covid-19 pandemic," *BMC Medical Education*, vol. 20, p. 396, 2020.

[3] X. Qian, A. Brutti, O. Lanz, M. Omologo, and A. Cavallaro, "Multi-speaker tracking from an audio–visual sensing device," *IEEE Transactions on Multimedia*, vol. 21, pp. 2576–2588, 2019.

[4] T. Yamada, K. Itoyama, K. Nishida, and K. Nakadai, "Sound source tracking using integrated direction likelihood for drones with microphone arrays," in *2021 IEEE/SICE International SII*. Iwaki, Fukushima: IEEE, 2021, pp. 394–399.

[5] Y. Ban, X. Li, X. Alameda-Pineda, L. Girin, and R. Horaud, "Accounting for room acoustics in audio-visual multi-speaker tracking," in *2018 IEEE ICASSP*. Calgary, AB: IEEE, 2018, pp. 6553–6557.

[6] Y. Li, H. Liu, and H. Tang, "Multi-modal perception attention network with self-supervised learning for audio-visual speaker tracking," in *AAAI–22*. IEEE, 2022, pp. 1456–1463.

[7] J. Wilson and M. C. Lin, "Avot: audio-visual object tracking of multiple objects for robotics," in *2020 IEEE ICRA*. Paris, France: IEEE, 2020, pp. 10 045–10 051.

[8] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: analysis, applications, and prospects," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, pp. 1–21, 2021.

[9] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights into Imaging*, vol. 9, pp. 611–629, 2018.

[10] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multi-task cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, pp. 1499–1503, 2016.

[11] X. Wu, R. He, Z. Sun, and T. Tan, "A light cnn for deep face representation with noisy labels," *IEEE Transactions on Information Forensics and Security*, vol. 13, pp. 2884–2896, 2018.

[12] S. Y. Nikouei, Y. Chen, S. Song, R. Xu, B.-Y. Choi, and T. R. Faughnan, "Real-time human detection as an edge service enabled by a lightweight cnn," in *2018 IEEE EDGE*. IEEE, 2018, pp. 125–129.

[13] S. Y. Nikouei, Y. Chen, S. Song, and T. R. Faughnan, "Kerman: A hybrid lightweight tracking algorithm to enable smart surveillance as an edge service," in *2019 16th IEEE CCNC*. IEEE, 2019, pp. 1–6.

[14] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: a survey," *Proceedings of the IEEE*, vol. 111, pp. 257–276, 2023.

[15] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *2016 IEEE CVPR*. Las Vegas, NV: IEEE, 2016, pp. 779–788.

[16] R. Girshick, "Fast r-cnn," in *2015 IEEE ICCV*. Santiago, Chile: IEEE, 2016, pp. 1440–1448.

[17] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis amp; Machine Intelligence*, vol. 39, pp. 1137–1149, 2017.

[18] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: single shot multibox detector," in *ECCV 2016*. Amsterdam, Netherlands: Springer, 2016, pp. 21–37.

[19] O. J. G.-P. D. Lathoud, G., "Av16.3: an audio-visual corpus for speaker localization and tracking," in *MLMI 2004*. Springer, 2004, pp. 182–195.

[20] B. Shan and Y. Fang, "A cross entropy based deep neural network model for road extraction from satellite images," *Entropy 2020*, vol. 22, p. 535, 2020.

[21] A. Kumar, A. Kaur, and M. Kumar, "Face detection techniques: a review," *Artif Intell Rev*, vol. 52, pp. 927–948, 2017.

[22] J. Li, Y. Wang, C. Wang, Y. Tai, J. Qian, J. Yang, C. Wang, J. Li, and F. Huang, "Dsfd: dual shot face detector," in *2019 IEEE/CVF CVPR*. IEEE, 2019, pp. 5060–5069.

[23] R. Ranjan, A. Bansal, J. Zheng, H. Xu, J. Gleason, B. Lu, A. Nanduri, J.-C. Chen, C. D. Castillo, and R. Chellappa, "A fast and accurate system for face detection, identification, and verification," *IEEE Transactions on Biometrics, Behavior, and Identity Science*, vol. 1, pp. 82–96, 2019.

[24] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," in *ICLR 2017*. IEEE, 2017.

[25] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiV*, 2018.

Christopher Limawan Christopher Limawan received B.Sc. degree in Cyber Security major from BINUS University in 2021 and received M.Sc. degree in Master of Computer Science in 2023. Algorithm optimization and cyber security are his research interests.

Benfano Soewito Benfano Soewito received B.Sc. degree in Physics major from Airlangga University, then received M.Sc. and Ph.D. degree in Electrical and Computer Engineering major from Southern Illinois University. Currently, he is a Deputy Head in Master of Computer Science department, BINUS University. He has various research interests, such as cloud computing, cyber security, and machine learning. Recently, his research focused on machine and deep learning implementation for various applications.