OPEN ACCESS
Indo-JC Indonesia Journal of Computing
ISSN 2460-9056
socj.telkomuniversity.ac.id/indojc

Ind. Journal on Computing
Vol. 5, Issue. 2, September 2020. pp. 67-80
doi:10.21108/indojc.2020.5.2.456

LBP Advantages over CNN Face Detection Method on Facial Recognition System in NOVA Robot

Luqman Bramantyo Rahmadi, Kemas Muslim Lhaksmana*, Donny Rhomanzah

Faculty of Informatics, Telkom University
Jl. Telekomunikasi No. 1, Bandung, Indonesia 40257
* kemasmuslim@telkomuniversity.ac.id

Abstract

Network-optimized virtual assistant (NOVA) is a robot developed by Bandung Techno Park (BTP) that can interact with humans for various purposes, such as a receptionist robot. NOVA robot is still in development and one of the main focuses is adding face recognition features so that the robot can actively greet and interact with humans. Therefore, we propose a face recognition and tracking system based on neural networks. This system is developed using the Google FaceNet feature extraction method. Previously, face detection in NOVA robot was implemented by employing the multi-task cascaded convolutional networks (MTCNN) method, whereas face tracking on the system was realized by using the modification of the MOSSE object tracking method. However, we found that the implementation of MTCNN in NOVA robot cannot run better than 30 fps. Therefore, this paper aims to solve this issue by investigating conventional face detection methods that could outperform MTCNN in this regard. Tests conducted on the ChokePoint dataset demonstrates that the system with LBP can achieve 30.44 fps framerate with a precision of 95% and recall of 83%. The test results show that LBP is not only better than MTCNN in identifying faces but also more efficient to compute.

Keywords: LBP, MTCNN, HAAR, NOVA, MOSSE

Abstrak

Network-optimized virtual assistant (NOVA) merupakan robot yang dikembangkan oleh Bandung Techno Park (BTP) yang dapat berinteraksi dengan manusia untuk berbagai keperluan, seperti robot resepsionis. Robot NOVA masih dalam tahap perkembangan dan salah satu fokus utamanya adalah menambahkan fitur pengenalan wajah sehingga robot dapat secara aktif menyapa dan berinteraksi dengan manusia. Oleh karena itu, kami memgajukan suatu sistem pengenalan dan pelacakan wajah yang berbasis *neural networks*. Sistem tersebut dikembangkan dengan menggunakan metode ekstraksi fitur Google FaceNet. Sebelumnya, deteksi wajah pada robot NOVA diimplementasikan dengan menggunakan metode *multi-task cascaded convolutional networks* (MTCNN), sedangkan pelacakan wajah diterapkan dengan menggunakan modifikasi dari metode pelacakan objek MOSSE. Namun, kami menemukan bahwa implementasi MTCNN pada robot NOVA tidak bisa berjalan lebih baik dari 30 fps. Oleh karena itu, paper ini bertujuan untuk mengatasi permasalahan tersebut dengan menyelidiki metode deteksi wajah konvensional yang dapat mengungguli MTCNN. Pengujian yang dilakukan pada dataset ChokePoint mendemonstrasikan bahwa sistem dengan LBP dapat mencapai *framerate* 30,44 fps dengan *precision* 95% dan *recall* 83%. Hasil pengujian menunjukkan bahwa LBP tidak hanya lebih baik dari MTCNN dalam mengidentifikasi wajah namun juga lebih efisien dalam komputasinya.

Kata Kunci: LBP, MTCNN, HAAR, NOVA, MOSSE

Received on xxx, accepted on xxx, published on xxx

I. Introduction

COMPUTER vision is one of the fields in information technology that can be utilized to obtain information from digital images or videos. Computer vision, alongside image processing, can be utilized for various needs such as edge detection for detecting the position of the sealring and edge on the sensor chips in ALICE (A Large Ion Collider Experiment) projects [10] to determine the quality of chip cutting, and person detection for detecting soccer players in the field to produce useful insights [15]. Computer vision has been widely applied in various fields with the aim to simplify human work and increase productivity. Robotics is one of them, and one of the products in the field of robotics is network-optimized virtual assistant (NOVA) robot. NOVA is a robot developed by Bandung Techno Park (BTP), which is one of the prominent science techno parks in Indonesia. NOVA can communicate with humans verbally and connected through a network. This robot can process a question and then provide information relevant to the question. This robot can also hear and speak in various languages. The NOVA robot is still in development, and one of the main focuses is adding NOVA Vision, which is a face recognition feature for NOVA.

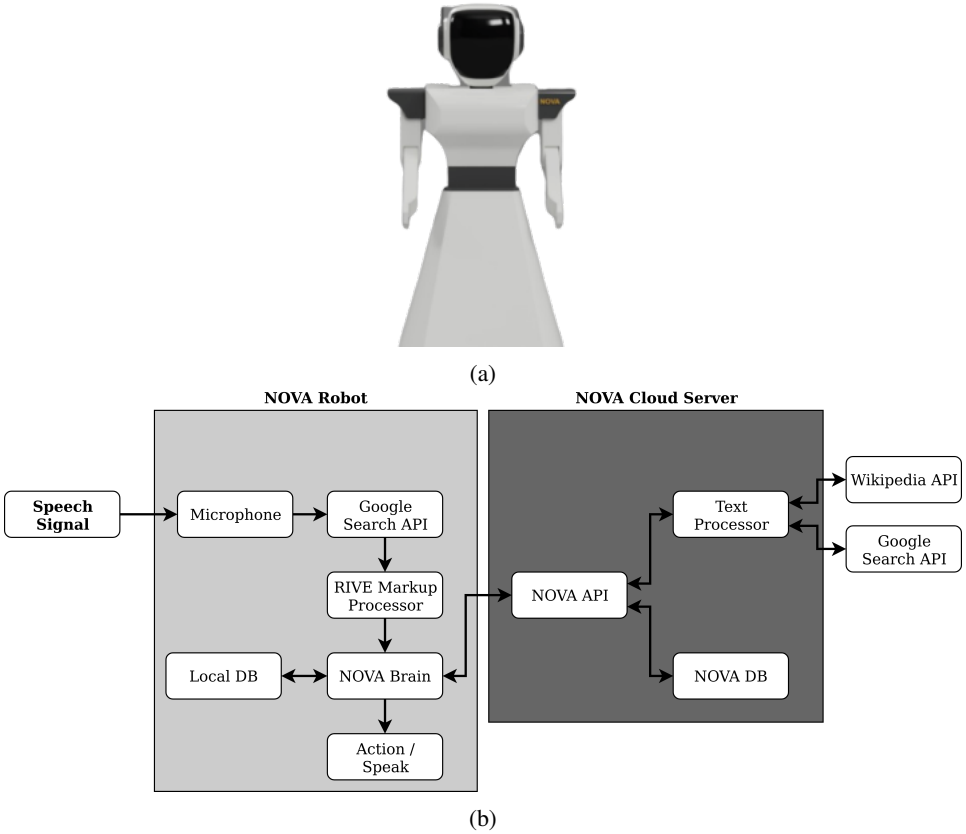


Fig. 1: NOVA robot design and workflow as shown in Fig. (a) and (b), respectively.

The circumstances have inspired us to create a face recognition system to complement the functionality and awareness of NOVA. We built a system that can detect the faces of people recorded by the camera, recognize the identity of the detected person and trace the person's face. With the addition of these features, NOVA will be capable to actively greet and move to respond to humans so that it becomes a more interactive virtual assistant.

In the realm of face recognition, the system must first be able to detect faces in the image. The face detection method used multi-task cascaded convolutional networks (MTCNN) [17]. After the face is detected, a feature extraction will be performed using the FaceNet method by Google based on neural

networks [11] to produce an embedding of the face image. The classification method used support vector machine (SVM). Finally, the system tracks faces based on the modification of the MOSSE object tracking algorithm [5].

However, we found that the implementation of MTCNN in NOVA robot cannot run better than 30 fps. Therefore, this paper aims to solve this issue by investigating conventional face detection methods that could outperform MTCNN in this regard. The conventional face detection methods that we chose is HAAR cascade and local binary pattern (LBP) cascade. HAAR and LBP cascade were chosen because they implement cascaded classifier structure, the same as MTCNN. Evaluations will be carried out using precision, recall, and F1-score evaluation metrics to measure system performance. In addition, we also compare the trade-off between inference time and performance of the two methods. It is expected that these tests can produce useful insights into the implementation of the system that we built.

II. LITERATURE REVIEW

Multi-task cascaded convolutional networks (MTCNN) uses three stages in determining the location of faces in an image. The three stages are proposal network (P-Net), refine network (R-Net) and output network (O-Net), respectively. The proposal network produces candidate bounding boxes, the refine network performs regression and non-maximum suppression (NMS) to select candidate bounding boxes, then the output network produces the final bounding box and facial landmarks. The performance result is quite convincing with an average accuracy of 85% on the WIDER FACE dataset, and 95% on the FFDB dataset [17]. However, as with neural network-based methods in general, this method has the disadvantage on computing loads that tend to be heavier than conventional methods and are more efficient when running on GPUs than CPUs. The method can run at 99 frames per second (fps) on the GPU but only gets 16 fps on the CPU [17].

Google FaceNet is a face recognition and clustering method [11]. The method uses the Inception network architecture [13] with the addition of Triplet Loss to minimize the L2 distance between faces with the same identity, and maximize the distance between different faces. Evaluation of this method in the LFW dataset yields an accuracy of 99.63% and 95.12% in the Youtube Faces dataset. The model produces an embedding size of 128 dimensions, which is a representation of the features of the input face. Like MTCNN, FaceNet which is also based on neural networks has a much higher computational requirement compared to conventional feature extraction methods in general. With an input size of 160×160 pixels, the model requires 1.6 Giga FLOPS (floating point operations per second) to process it into embedding [11].

A support vector machine (SVM) is a statistical learning method that implements hyperplane to separate data by calculating the distance between them. Classification using SVM can be applied in various problems such as medical decision support, time series methods, and also face authentication [7]. SVM can also handle linear and non-linear data by applying a kernel to the learning process. Data generated from the FaceNet feature extraction process produces linear data in the form of a 128-dimensional array, so that the classification model can be trained with SVM.

Another work [5] proposes a new method for tracking objects that are robust to changes in scale but still efficient in computation. The proposed method is based on the minimum output sum of squared error (MOSSE) filter [3] with the application of fast scale space tracking which applies different filters each for scale change and translation or object movement. This method can work quickly and efficiently on the CPU, making it suitable for applications on embedded systems such as robots.

The use of HAAR-like features in object detection is one of the first methods for detecting objects [14]. The method uses a multilevel classifier where checking is not continued to the next level when no candidate object is found at one level. Meanwhile, another method named local binary pattern (LBP) can also be trained to recognize human faces by using texture and shape descriptors [1]. This method compares the value of one pixel with its neighboring pixels in binary. This method can run in real-time because all calculations are integers [1], compared to HAAR features that use float. This makes LBP suitable for application in embedded systems. Based on this background, this research investigates whether LBP is appropriate to solve MTCNN issue, i.e. its limitations to run beyond 30 fps in our system.

III. Research Method

A. Workflow

Fig. 2 is an outline of how the system works. The system designed is a system that can detect and recognize faces on video or camera input and then trace the detected faces. In general, there are 6 steps in this system. First, the system captures frames obtained from the camera. Next, face detection is performed, and the detected faces are pre-processed to match the input for the feature extraction step. And then, the feature extraction process results in the embedding of face images which will then used to predict the identity of the face. And the last step is tracking the face on the frame.

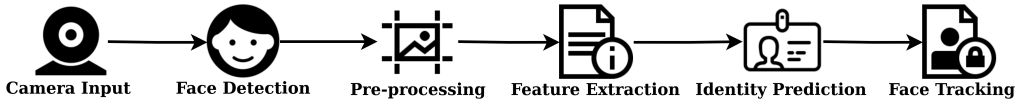


Fig. 2: System workflow.

B. Implementation Model

The implementation model of the system is described in Fig. 3. The system will be implemented on NOVA robots which have limitations on hardware so the system must be divided into two parts, namely the local computing side of the NOVA robot and the cloud computing side of the work server. The face tracking process is performed locally on the robot, while the face detection, feature extraction, and classification processes are performed on the cloud. Work servers are computers that have a dedicated GPU because CNN-based computing will run faster on the GPU to speed up the system. NOVA robots and work servers are connected through a local area network (LAN) as a medium for data and information transfer between them.

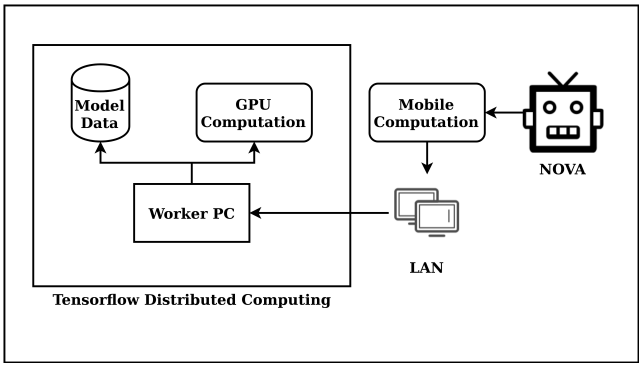


Fig. 3: Implementation model of the system.

C. Process Flow

The system process flow is the detailed workflow of the system as shown in Fig. 4. In the first stage, there were three models loaded, namely the MTCNN model, the FaceNet model, and the classifier model. The model is loaded at the beginning because it only needs to be loaded once, not on each loop. The next stage is face detection, the camera first takes the frame and then it will first be checked for interrupt. The interrupt is in the form of manual input from the user and will stop the system if it exists. An interrupt is needed so that the system does not loop indefinitely. Then it will be checked whether there is an active tracker. If there is an active tracker, then the process goes directly to the face tracking step so that in each iteration it does not have to go through the face detection and feature extraction step that has a high computational load. If there is no active tracker, face detection will be performed on the frame. If

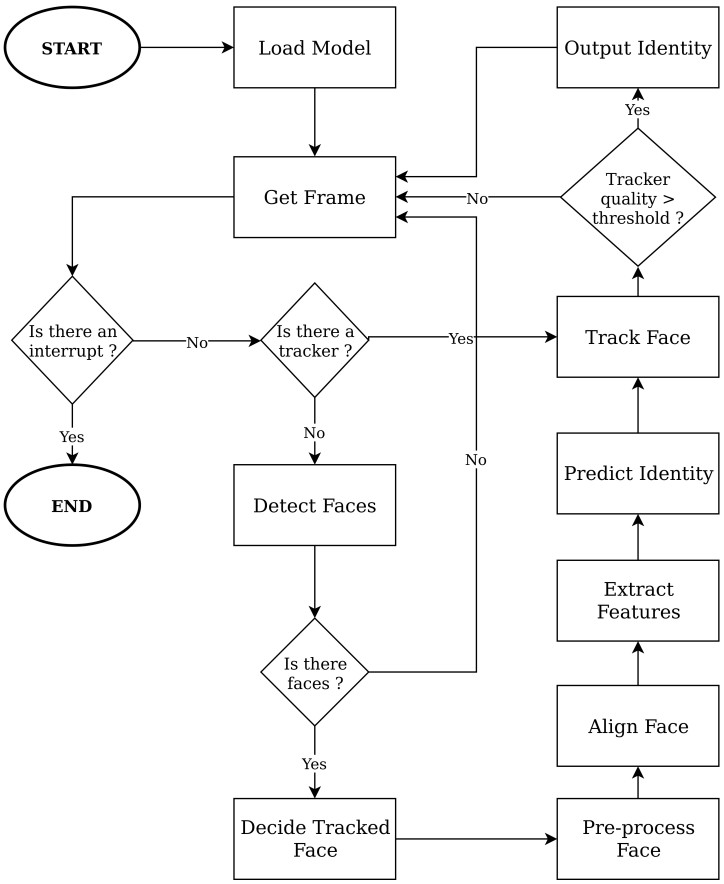


Fig. 4: Process flow of the system as a whole.

there isn't any detected face, next frame from the camera will be taken. If faces are detected, it will be determined which face has the largest bounding box, because only one face will be tracked.

The third step is pre-processing. In this step, the face will be cut from the frame according to its bounding box, then resized to 160×160 pixel size according to the input size in the FaceNet method [11]. Face alignment will also be applied to the face to normalize the face so that the location of the landmark standardized in the middle. The fourth step is extraction of features from face that have been normalized. The fifth stage is predicting the face's identity from the features that have been extracted. The sixth stage is the face tracking process, which uses bounding box from previously detected face as it's input. Tracking is carried out on each frame as long as the tracker quality exceeds the threshold. If the tracker quality falls below the threshold, the tracker will be re-initialized and face detection will be done again after capturing the frame from the camera.

D. Methods Used

1) Multi-Task Cascaded Convolutional Networks (MTCNN): MTCNN is a face detection method based on a neural network which uses 3 levels of convolutional network namely proposal network (P-Net), refine network (R-Net), and output network (O-Net) [17] . The proposal network classifies faces using cross-entropy loss as in the equation 1, where p_i is the probability produced by the network. The notation y_i^{det} denotes the ground truth label. L denotes the list of the result produced by the network.

$$L_i^{det} = -(y_i^{det} log\,(p_i) + \left(1 - y_i^{det}\right)(1 - log\,(p_i))) \tag{1}$$

Refine network regresses the bounding box generated on the P-Net by applying Euclidean loss as

described in the equation 2, where $\hat{y}_i^{box}$ is the regression target obtained from the network and y_i^{box} is the ground truth coordinate.

$$L_i^{box} = \left\| \hat{y}_i^{box} - y_i^{box} \right\|_2^2 \tag{2}$$

Output network performs facial landmark regression to produce facial landmark output by applying Euclidean loss as described in the equation 3, where $\hat{y}_i^{landmark}$ is the facial landmark's coordinate obtained from the network and $y_i^{landmark}$ is the ground truth coordinate.

$$L_i^{landmark} = \left\| \hat{y}_i^{landmark} - y_i^{landmark} \right\|_2^2 \tag{3}$$

The final outputs are bounding box coordinates and facial landmark points.

2) HAAR Cascade: HAAR cascade is a face detection method that uses HAAR-like features as a face descriptor. The way HAAR cascade works is by generating candidate features by subtracting the number of pixels in the gray box by the number of pixels in the white box for each possible size and location and optimizing the result using adaptive boosting (AdaBoost) to reduce the number of candidates.

Then the candidate is tested on the graded classifier, where each level has a different number of features, with the top level usually having very few features. The detectors has 6000 features at 38 levels [14].

3) Local Binary Pattern (LBP) Cascade: The LBP object detection method is a holistic method where features are obtained from local representations obtained by comparing one pixel with its neighboring pixels [9].

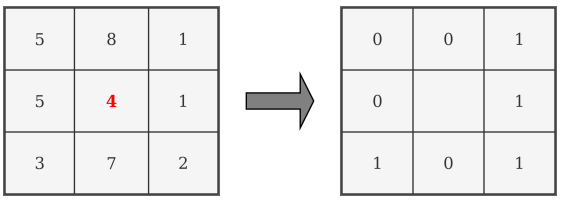


Fig. 5: Feature matrix generated by LBP.

The comparison produces a binary value with a value of 1 if the intensity of the middle pixel is more than or equal to its neighboring pixels, otherwise, it's given a value of 0. Next, LBP value will be calculated from the 8 binary values to provide a sequence of numbers on each neighboring pixel with consistent directions for all pixels in the image.

4) FaceNet: Google FaceNet [11] is a facial feature extraction method that uses Inception Resnet v1 architecture [13], with an additional loss function called Triplet Loss. Triplet Loss consists of two corresponding face thumbnails, and one different face thumbnail to separate positive and negative pairs with a measured distance.

5) Support Vector Machine (SVM): Support vector machine (SVM) is a learning algorithm used in solving classification and regression problems using hyperplane [7]. The hyperplane is a field that has $n-1$ dimensions in space that has n dimensions. Hyperplane provides restrictions that separate one class from another. However, hyperplane often adjusts its boundaries with an uneven distribution of data. Therefore, in SVM several parameters can be set. These parameters are the kernel, regularization, gamma, and degree.

6) MOSSE-based Object Tracker: Another work [5] proposes an object tracking method where improvement is made on the MOSSE object tracker [3] by introducing discriminative correlation filters for multidimensional features and exhaustive scale space tracking. The proposed Discriminative Correlation Filter has three types, namely $1D$ filter for scale estimation, $2D$ filter for translation and $3D$ filter for localization of targets on a scale and space as a whole. The three types of filters make it possible to track objects that change translation and scale in real-time.

E. Dataset

The dataset that will be used to measure system performance is the ChokePoint dataset [16]. ChokePoint dataset is granted by National ICT Australia Limited (NICTA) to be used for academic research. We chose the ChokePoint dataset because it resembles the system implementation scenario of a real-world surveillance recorded through a camera. The ChokePoint dataset is a video dataset consisting of 48 video sequences and each sequence consists of a series of 800×600 frames with a frame rate of 30 fps. The dataset consists of 25 subjects in portal 1 (P1) and 29 subjects in portal 2 (P2). In each sequence, there is only one subject per frame and the face images contained in each sequence have been categorized according to their respective identities. Each frame in the sequence is stored in the JPEG file extension while each face image is stored in the PGM file extension.

F. Evaluation Metric

Evaluation metrics are used as an exact measurement in system testing. The evaluation metrics used are accuracy, precision, recall, and F1-Score obtained from the confusion matrix. Face detection performance is measured by Intersection Over Union (IoU) metric.

1) Confusion Matrix: In table I, True Positive (TP) is the number of positive classes that are correctly predicted. True Negative (TN) is the number of negative classes correctly predicted. False Positive (FP) is the number of positive classes that are wrongly predicted. False Negative (FN) is the number of negative classes that are wrongly predicted.

	Actual Positive	Actual Negative
Predicted Positive	TP	FP
Predicted Negative	FN	TN

Table I: Confusion matrix.

Accuracy is the ratio of total correct predictions to the number of predictions made. Precision is the ratio of total true positive predictions to total positive predictions. Recall is the ratio of total true positive predictions to total true predictions. While F1-Score is the harmonic mean of Precision and Recall.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$F1Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{7}$$

2) Intersection Over Union: Intersection Over Union (IoU) is a measure of compatibility of the object detection algorithm. $IoU(p, gt)$ represents the ratio of parts of predicted bounding boxes (B_p) that intersect with ground truth bounding boxes (B_{gt}). Bounding boxes are x, y, height, and width coordinates that form a box that points to the location of an object in the image. The greater the IoU value, the better the performance of an algorithm in detecting objects.

$$IoU(p, gt) = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})} \tag{8}$$

According to the guidelines of the PASCAL visual object challenge [6], a detection is considered true if the IoU exceeds 0.5 (50%).

IV. Results and Discussion

1) Dataset Baseline: Referring to the testing protocol for ChokePoint dataset [16], the dataset is divided into two groups, where the two groups will take turns as a development set and evaluation set. Threshold and other parameters will be tuned in the development set, and then will be tested in the evaluation set. And then, the evaluation results are taken from the average evaluation of the two groups.

	Total Frames	Total Sequences
G1	16665	8 (4 P1 & 4 P2)
G2	20652	8 (4 P1 & 4 P2)

Table II: Distribution of the dataset. G1 is group 1 and G2 is group 2. Each group consist of sequences recorded from two portal, namely portal 1 (P1) and portal 2 (P2)

The ground truth in the ChokePoint dataset only consists of the identity and eye and mouth coordinates of the frame, while the IoU metric requires bounding boxes as input. Therefore we took the ground truth bounding box for the ChokePoint from another work [2], then we combine it with the ground truth identity of the ChokePoint dataset.

2) Model Training and Validation: The classification models are trained with SVM using face datasets that have been taken from each video sequence of the ChokePoint dataset. Faces are categorized according to their respective identities and sequences. Next, we further separate the dataset according to the group in table II. And then, for each group, we train the model using the cross-validation method.

We use the SVM module from the scikit-learn library with the Tensorflow-GPU backend and the Python programming language. Tensorflow-GPU is used as a backend for FaceNet feature extraction process. The parameters we set for the SVM are kernel and gamma. We use the polynomial kernel and we set the gamma values to auto to train the models. Then we test the performance of the model on testing data.

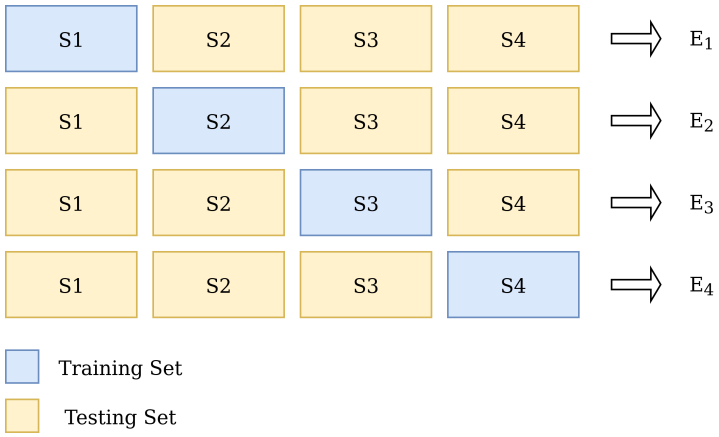


Fig. 6: Illustration of the model validation method, S is a video sequence on each portal in the group. Each sequence will alternately be used as a training set, and the other sequences will become testing set to measure the performance of the model as described by E_i.

We evaluate the results of cross-validation as a reference for which model has the best performance, and we use the best models as a classification model for testing the system.

3) Evaluation: Evaluations were carried out on the face detection method applied to the face recognition system that we built. There are three methods that we evaluate, namely MTCNN, HAAR cascade, and LBP cascade. We use the cascade classifier module in the OpenCV library, Tensorflow-GPU backend, and Python programming language. We use Tensorflow-GPU backend to perform neural network calculations

	Best Sequence	Accuracy
G1P1	P1E_S1_C1	99.6 %
G1P2	P2L_S1_C1	99.6 %
G2P1	P1E_S4_C1	99.7 %
G2P2	P2L_S3_C3	99.3 %

Table III: The results of the model validation+ show the sequences that produce the model with the best accuracy for each group (G) and portal (P).

on the MTCNN face detection method and FaceNet feature extraction. All three methods will be run on the same computer as the control.

The face detection model used for MTCNN is a publicly available pre-trained model [17]. The model was trained with the VGGFace2 dataset [4] which contained 3.31 million images from 9131 subjects. While the model for HAAR cascade uses the publicly available model in the OpenCV library namely haarcascade_frontalface_alt.xml, and the model for LBP cascade also uses the OpenCV model namely lbpcascade_frontalface.xml. The FaceNet feature extraction model also taken from a publicly available pre-trained model [11]. The face classification models is as stated in the III table. Evaluation of the model in group 1 (G1) will be done using a video sequence dataset in group 2 (G2), and vice versa.

The experiment results are described in a confusion matrix with the help of IoU. True Positive ($TP(t)$) is the number of predictions that have an IoU ($IoU(p, gt)$) exceeding the threshold t and a correct identity (I_{same}). False Positive ($FP(t)$) is the number of predictions that have an IoU ($IoU(p, gt)$) below the threshold t and a correct identity (I_{same}). While False Negative (FN) is the number of faces that are not detected by the system. TP and FP are described in equation 9 and 10, respectively.

$$TP(t) = \{(p, gt) \in I_{same},\ with\ IoU(p, gt) > t\} \tag{9}$$

$$FP(t) = \{(p, gt) \in I_{same},\ with\ IoU(p, gt) < t\} \tag{10}$$

We evaluate the system using macro-average metrics. Macro-average metrics take the average of measurement metrics in different data sets. The dataset in this experiment is a video sequence. Macro-averages are used when classes are in a balanced set, where in the data we use there are no video sequences where the majority of the persons present in the sequence consist of one identity [12]. The macro-average metrics are described by equation 11, 12, 13, and 14.

$$Precision_M = \frac{\sum_{i=1}^{n} \frac{tpi}{tp_i + fp_i}}{n} \tag{11}$$

$$Recall_M = \frac{\sum_{i=1}^{n} \frac{tpi}{tp_i + fn_i}}{n} \tag{12}$$

$$Average\ Accuracy = \frac{\sum_{i=1}^{n} \frac{tp_i + tn_i}{tp_i + fp_i + fn_i + tn_i}}{n} \tag{13}$$

$$F1Score_M = \frac{\left(\beta^2 + 1\right) Precision_M Recall_M}{\beta^2 Precision_M + Recall_M} \tag{14}$$

The IoU thresholds that we use are 50% (0.5) IoU and 20% (0.2) IoU. The thresholds is based on the average results of the IoU of each method described in Fig. 7.

MTCNN has a very low average IoU because the method produces facial bounding boxes that are larger than the face area in general on ground truth bounding boxes. As shown in Fig. 9, MTCNN has the lowest IoU among the three methods. The difference in system performance between the two IoU thresholds can be seen in Fig. 8.

Fig. 8 shows an improvement of up to 70% between the evaluation metrics of the MTCNN system with an IoU threshold of 0.5 and 0.2, while other methods have only improved by about 10%. This shows

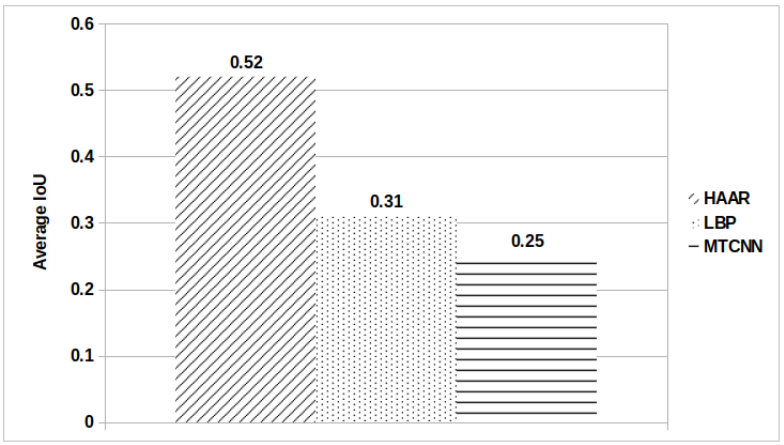
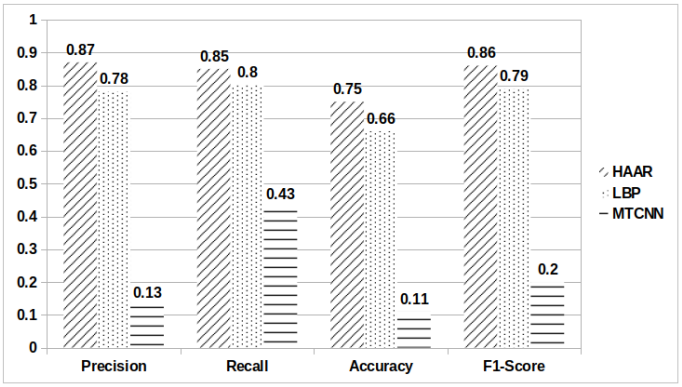
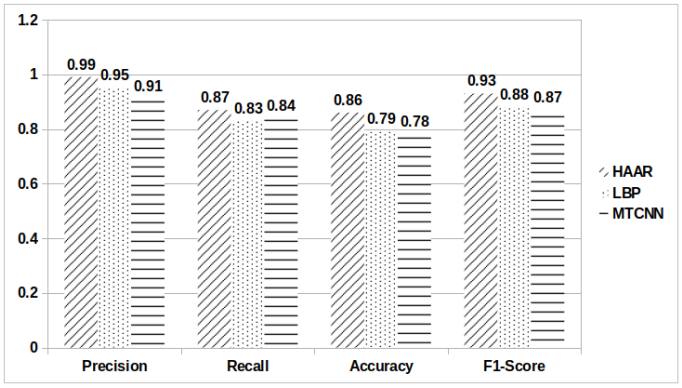


Fig. 7: Average IoU of each face detection method. HAAR has the best performance compared to LBP and MTCNN, with IoU exceeding the 0.5 threshold.



(a)



(b)

Fig. 8: System performance with each face detection method. Fig. (a) shows the performance at 50% (0.5) IoU threshold, Fig. (b) shows the performance at the 20% (0.2) IoU threshold. There is a drastic increase of 67% in F1Score for the MTCNN method with IoU threshold of 0.2.

that the IoU thresholds set in the experiment are highly dependent on the ground truth bounding box of the dataset, and the IoU standard of 0.5 [6] is not always relevant to all methods and datasets. Even though the threshold has been adjusted accordingly, the system performance with the MTCNN detection

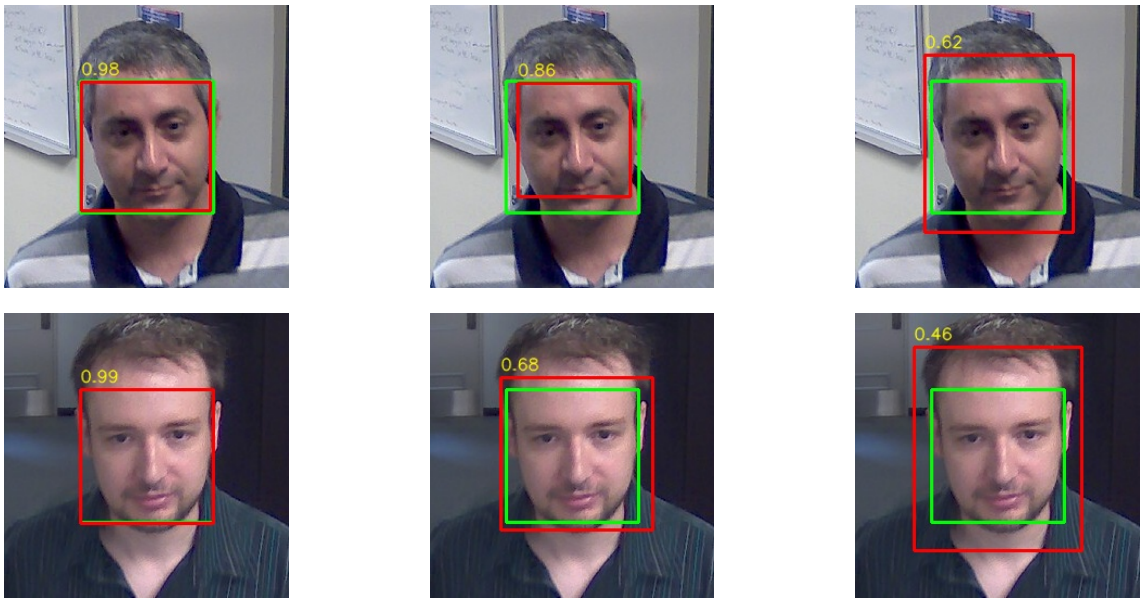


Fig. 9: We took one video sequence from each group in the ChokePoint dataset, and then we ran face prediction on them to produce predicted bounding boxes. The predicted bounding box with the best IoU for each method is drawn alongside its ground truth bounding box into its corresponding frames to produce the photos above. The green box represents the ground truth bounding box, the red box represents the predicted bounding box, and the yellow text is the IoU value.

method is still inferior compared to conventional detection methods, with a noticeable margin of up to 8%. Bounding boxes that tend to be larger produce more noise so that more information is irrelevant to the extracted feature. This causes the prediction to be more susceptible to error.

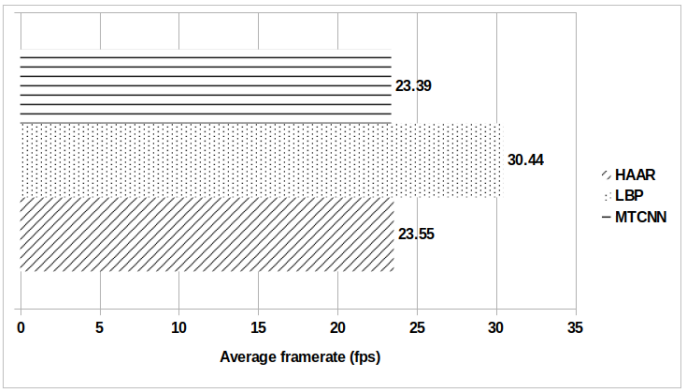


Fig. 10: Framerate achieved by the system with each face detection method. The greater the value the better. LBP is the only method that achieves 30 fps.

In addition to using accuracy, system performance is also measured by the speed of the process. The system we built must be able to process data at real-time speed. The processing speed of a recognition system is limited to the recognition speed of the human eye [8]. The system we built is created to be able to interact with humans so it is required to operate at 30 fps. Fig. 10 shows that only the system with LBP detection method that can achieve 30 fps. Inference time and tracker re-initiation is the factor that gives the LBP detection method an advantage over the other two methods. The inference time that we measure is the time needed to detect faces on the frame.

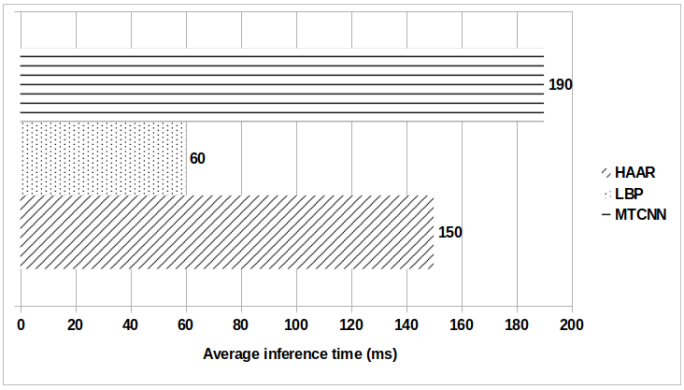


Fig. 11: The average inference time of each face detection method. The smaller the value the better. The best detection speed is achieved by LBP at 60 milliseconds.

Fig. 11 shows that LBP inference time is 2.5 times faster than HAAR and 3 times faster than MTCNN. This result is backed by the fact that LBP calculations are performed in integers while MTCNN and HAAR compute in float [1]. Re-initiation of the tracker is also less frequent on a system with the LBP detection method as described in Fig. 12.

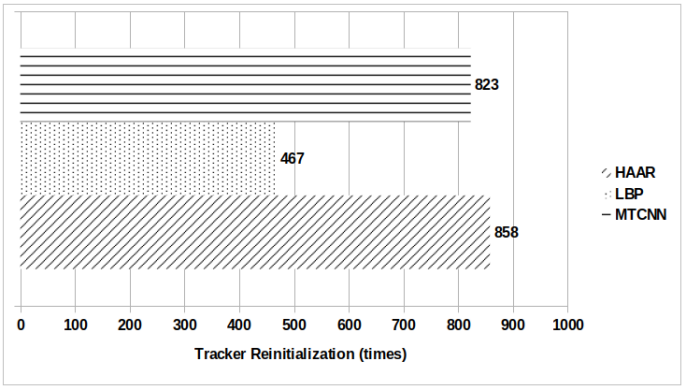


Fig. 12: The number of tracker re-initiations on the system with each face detection method. The smaller the value the better. LBP is the most efficient for tracking faces with the least amount of re-initiations.

When the tracker status is active, the tracker estimates the location of the tracked face on each frame instead of performing face detection on them. The face estimation process is much lighter in computation compared to the face detection process. The face estimation process returns a confidence value. When the confidence value drops below the threshold, the tracker re-initiates by re-tracking the face. Therefore, the more frequently the tracker initiates, the slower the processing time of the system, and vice versa.

The implementation of LBP face detection method makes the system less frequently to re-initiate the face tracker compared to the other methods. The behaviour occurs because the bounding box produced by LBP is sufficient, not too perfect like HAAR but also not too deviated like MTCNN, as shown in Fig. 9. In other words, there is sufficient variance in LBP bounding boxes. This helps the face tracking algorithm to estimate face better because it is more tolerant of translation and scale changes.

V. Conclusion

The NOVA robot, which was developed by Bandung Techno Park (BTP), is a robot built for various purposes, such as a receptionist robot. The robot is developed with human interactivity in mind, and

one of the main goals is that NOVA can actively greet and interact with humans. Previously, NOVA's facial recognition and tracking features, which we call NOVA vision, were implemented using a neural network face detection method. However, the implementation cannot perform at 30 fps, which is a necessary requirement for NOVA vision. Therefore, we investigate conventional face detection methods for the system, namely HAAR cascade and LBP cascade. Based on the test results, it can be concluded that the LBP cascade method produces the best real-time performance. The LBP cascade method in our implementation achieves fastest framerate at 30.44 fps, which outperforms MTCNN, with better precision (95%), recall (83%), accuracy (79%), and F1-score (88%). The framerate achieved by the LBP cascade method meets the 30 fps real-time processing standard, and more efficient to compute compared to MTCNN. The test results also show that MTCNN has the lowest average evaluation value of the three face detection methods tested and also recorded the slowest framerate. These results show that the method developed with neural networks does not always produce better performance than conventional methods. The face recognition system that we built will be applied to NOVA robot that prioritize processing speed for interactivity. And from the test results, we conclude that the LBP detection method is more suitable than the MTCNN method to be applied to the system.

ACKNOWLEDGMENT

Thank you to Allah SWT, Bandung Techno Park, and family who always support along the way.

REFERENCES

[1] Timo Ahonen, Abdenour Hadid, and Matti Pietikäinen. Face Recognition with Local Binary Patterns. In Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Dough Tygar, Moshe Y. Vardi, Gerhard Weikum, Tomás Pajdla, and Jiří Matas, editors, *Computer Vision - ECCV 2004*, volume 3021, pages 469–481. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[2] Safa Alver and Ugur Halici. Attentive Deep Regression Networks for Real-Time Visual Face Tracking in Video Surveillance. *arXiv:1908.03812 [cs]*, August 2019.

[3] Dav Bolme, J. Ross Beveridge, Bruce A. Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2544–2550, San Francisco, CA, USA, June 2010. IEEE.

[4] Qiong Cao, Li Shen, Weidi Xie, Omkar M. Parkhi, and Andrew Zisserman. VGGFace2: A dataset for recognising faces across pose and age. *arXiv:1710.08092 [cs]*, May 2018.

[5] Martin Danelljan, Gustav Häger, Fahad Shahbaz Khan, and Michael Felsberg. Accurate Scale Estimation for Robust Visual Tracking. In *Proceedings of the British Machine Vision Conference 2014*, pages 65.1–65.11, Nottingham, 2014. British Machine Vision Association.

[6] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

[7] Theodoros Evgeniou and Massimiliano Pontil. Support Vector Machines: Theory and Applications. volume 2049, pages 249–257, January 2001.

[8] Qing-Yi Gu and Idaku Ishii. Review of some advances and applications in real-time high-speed vision: Our views and experiences. *International Journal of Automation and Computing*, 13(4):305–318, August 2016.

[9] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, July 2002.

[10] G. R. Panjaitan, K. M. Lhaksmana, E. Prakasa, and L. Musa. Detecting the position of the sealring and the edge on the sensor chip. *Journal of Physics: Conference Series*, 1192:012063, March 2019.

[11] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A Unified Embedding for Face Recognition and Clustering. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, June 2015.

[12] Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4):427–437, July 2009.

[13] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *arXiv:1602.07261 [cs]*, February 2016.

[14] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–511–I–518, Kauai, HI, USA, 2001. IEEE Comput. Soc.

[15] Adhi Dharma Wibawa and Atyanta Nika Rumaksari. Soccer Players Detection Using GDLS Optimization and Spatial Bitwise Operation Filter. *Journal of Data Science and Its Applications*, 2(1):1–10, April 2019.

[16] Yongkang Wong, Shaokang Chen, Sandra Mau, Conrad Sanderson, and Brian C. Lovell. Patch-based probabilistic image quality assessment for face selection and improved video-based face recognition. In *CVPR 2011 WORKSHOPS*, pages 74–81, Colorado Springs, CO, USA, June 2011. IEEE.

[17] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, October 2016.