NoonGil Lens+: Second Level Face Recognition from Detected Objects to Decrease Computation and Performance Trade-off

J Vianto[1], D B Setyohadi*[2], A S Prabuwono[3], M S Azmi[4], E Julianto[5]

[1,2,5]Departments of Informatics, Universitas Atma Jaya Yogyakarta, Yogyakarta

[3]Faculty of Computing & Information Technology in Rabigh King Abdulaziz University, P.O. Box 344, Rabigh 21911, Saudi Arabia

[4]Faculty of Information and Communications Technology, Universiti Teknikal Malaysia Melaka (UTeM), Malaysia

Email: djoko.budiyanto@uajy.ac.id[2]

Submitted: 19 Jan 2022, revised: 26 Feb 2022, accepted: 27 Feb 2022

Abstract. Artificial intelligence has developed in various fields. The development became more significant after Neural Networks (NN) began to gain popularity. Convolutional Neural Networks (CNNs) are good at solving problems such as classification and object detection. However, CNN's model tends to function to solve a specific problem. In the case of both object detection and face recognition, it is difficult to make a single model that works well. NoonGil Lens+ is expected to be an approach that can solve both problems at once. As well as being a solution, it is also hoped that this approach can reduce the trade-off of accuracy and execution speed. The approach we propose can be called Noongil Lens+, a system that connects YOLOv3 and FaceNet. It is inspired by a Korean series called 'STARTUP'. The author only develops the FaceNet model and the proposed system in this paper (NoonGil Lens+). Region Selection, a machine learning-based greedy approach was proposed to determine snapshots to fed into FaceNet for facial identity classification. FaceNet is trained on the CelebA dataset which has gone through the preprocessing process and is validated using the LFW dataset. NoonGil Lens+ was validated using 70 images of 7 celebrities, characters, and athletes. In general, the research was carried out successfully. NoonGil Lens+ using Region Selection has an accuracy of up to 75.2%. The Region Selection execution speed is also faster compared to Cascade Faces.

Keywords: face verification, object detection, deep learning, you look only once, FaceNet, convolutional neural networks

1. Introduction

Quoting pertuni.or.id, at least 3,750,000 Indonesians experience blindness or weak vision [1]. By using computer vision, we can help them to see the world. One of the most improving computer vision methods is Convolution Neural Networks (CNNs).

There is a fictional software called NoonGil, which is introduced in a Korean series 'START UP'. NoonGil can detect objects via smartphones camera then calls out to the user. It is made to help blind people. Basically, it inspired us to improve it by adding a facial recognition feature, which is not implemented in the series.

Making a single deep learning model detect objects and recognize faces is a huge challenge. The model must be deep enough, but on the other side, it can get into underfitting and overfitting problems. For instance, You Look Only Once (YOLO) has 26 layers consisting of 24 convolutional layers and 2 fully connected layers. YOLO runs at 55 FPS on Titan X GPU [2]. FaceNet has 17 layers that contain 11 *Inception layers*. Each Inception layer has 2 hidden layers, implicitly FaceNet has 28 layers [3]. On our FaceNet implementation (125x125), it can run 0.58 times faster than YOLO.

Commonly, this problem is solved by developing separate models like what we did. Our approach is trying to optimize computation and performance trade-off by using YOLOv3 'person' outputs as FaceNet input to recognize who is it, as simple as that. We use YOLOv3 implementation from Github on this paper without any change, the source will be cited in section III. We develop FaceNet based on its original paper [3] with a bit changes. Region Selection is introduced to connect YOLOv3 and FaceNet. Instead of using the face detection method, we use a directed algorithm to crop images from YOLOv3 output. Predicted face images are generated based on center positions and crop ratios from KMeans's centroids. Region Selection outperforms Faces Cascade (face detection) by accuracy and execution times.

The overview of this paper is as follows: Section II is talking about some papers that we've reviewed on related topics. We describe our method in section III, specifically about FaceNet in section III.A, and Region Selection in section III.B. We talk about how Noongil was evaluated and datasets that we used in section IV. We provide some quantitative results of our research in Section V.

Object detection has been studied for a long time. Its popularity has increased due to the development of CNNs. Object detection using CNNs shows good results. Regional Convolutional Neural Networks (R-CNNs) is an object detection algorithm that is reported to have good performance with a mean average precision (mAP) of 47.9% in the PASCAL VOC 2011 dataset [4]. Faster R-CNNs with the RPN method on the PASCAL VOC 2007 + 2012 dataset are reported to have an mAP of up to 73.2% [5]. Then, YOLOv3 with an input size of 320 is reported to have an mAP of 28.2% with a speed of 22ms [6].

The R-CNNs algorithm is divided into three modules. These modules are region proposals, feature extraction, and classification with a Support Vector Machine (SVM). Region proposals on R-CNNs use the selective search method. This module extracts 2000 region proposals, in the experiment, Ross Girschick et al. used "fast mode" selective search. R-CNNs extracted as many as 4096 dimensionless vector features in each proposal region using Caffe (a deep learning framework). The trained SVM is used to evaluate each feature that has been transformed using CNNs. The SVM evaluation results are in the form of a score for each class, which is then carried out with greedy non-maximum suppression [4].

Faster R-CNNs proposed Region Proposal Network (RPN) method. Faster R-CNNs is an algorithm that develops Fast R-CNNs in terms of speed while maintaining model performance. Compared to R-CNNs, Faster R-CNNs are only divided into two modules, namely fully convolutional network and Fast R-CNNs detector [5]. The RPN searches for square object proposals from an input image.

Joseph Redmon et al proposed the first version You Look Only Once (YOLO) algorithm in 2016. Experimental results show that his algorithm is faster than Faster Regional CNNs, without significantly reducing the model's accuracy (57.9% mAP on VOC 2012 dataset). YOLO utilizes image's features to predict bounding boxes and their classes within an input. This system split the input into grids. Each grid

predicts a number of bounding boxes and their confidence. The confidence value is the intersection over union (IOU) between the predicted box and any ground truth box. Each grid cell predicts class probability [2].

Within a few years, several variants appeared such as YOLO9000, YOLOv3 in 2017, and PP-YOLO in [7], [6], and [8]. In this study, the author uses YOLOv3 which is the last official version of the original authors (Joseph Redmon, et al). YOLOv3 follows the YOLO9000 bounding box prediction method by using dimension clusters as anchor boxes. The use of anchor boxes is reported to improve prediction performance [7]. The difference between anchor boxes and bounding boxes is the way they are encoded. YOLOv3 still uses K-Means Clustering to determine the initial bounding box size during training. YOLOv3 uses Darknet-53 for feature extraction [6].

Implementing facial recognition algorithms using CNNs poses a big challenge, namely a large dataset is required. Therefore, the algorithmic approach for facial recognition using conventional CNNs will not function well in systems where facial and identity databases are constantly being updated. This problem is addressed in the writings of LaraslavMelekhov et al. who trained CNNs to recognize general similarities in pairs of images from a dataset. It is called Siamese Network also known as the "learning from similarity" approach. The results of these studies prove that NNs can generalize the similarity to image pairs in the dataset [9][10]. Siamese Network was developed into Triplet Loss in [3] by Google's research team, this model accepts three inputs called anchors, negative, and positive. Anchor is the comparison, negative is the image that does not match (label) with the anchor, while positive is the image that matches the anchor. This model pushes the distance between the anchor and negative images further away while bringing the anchor and positive images closer.

2. Method

We propose the three stages the process, first data, CelebA dan Labeled Faces in Wild (LFW) is preprocessed. After that two-step approaches are performed, detection and followed prediction. Both of them use CNNs architectures. The two CNNs architectures are YOLOv3 and FaceNet. Both architectures are working together to build Noongil Lens+. We use YOLOv3 implementation from [11] without any developments or changes, so we won't discuss more about YOLOv3. Simply, NoonGil Lens+ can be illustrated as below:

YOLOv3 detects objects in an image, then every 'person' object is forwarded to FaceNet to predicts who it is, compare each FaceNet output with the embeddings database using euclidean distance. *J* is a label set of embeddings database.

Figure 1. NoonGil Lens+. Embedding database is generated embeddings using FaceNet for each face that we want to recognize. Only 'person' objects are fed into FaceNet.

$$d(x,y)_j = \left((x_1-y_1)^2 + \ldots + (x_n-y_n)^2\right)^{1/2}; j=label(x),\ j \in J \quad (1)$$

S is distances set between E_d (Embeddings Database) items and E_z (predicted embeddings) of every 'person' object.

$$S = \{d(x,y)_j \mid x \in E_d, y \in E_z\} \quad (2)$$

$$P(t)=\{x \mid x \in S, x \leq t\} \quad (3)$$

After getting P at threshold t, we find the greatest number of labels as classification (z) and it is greater than N. If there are more than one label, we choose the least mean distance label.

2.1. FaceNet Model

FaceNet Validation rate (VAL) is reported depending on the size of the input. Input 96x96 px has good VAL compared to 224x244 px, which does not significantly decrease FaceNet's VAL. We choose input 125x125x1 px (grayscale) to reduce computation time, especially when training. We did not try another input size. We use 128-D embedding output size, which is reported to have better VAL than 64, 256, and 512.

Each convolution is applied batch norm, non-linear activation, and L2 regularization. Batch norm can decrease training times [12] and L2 regularization can increase model performance and decrease overfitting [13].

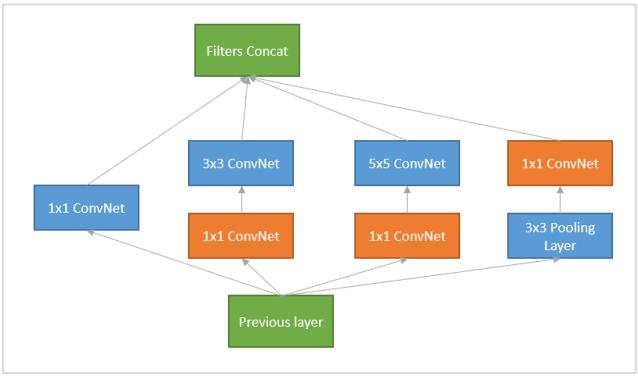


Figure 2. Inception model. using ConvNet as convolution layer.

FaceNet uses the Inception Model to increase the number of hidden layers without significantly increasing computation costs. The Inception model is fully inspired by [14], it is slightly different from FaceNet's original paper, which uses Inception-Resnet. We also use either max pooling or average pooling.

Table 1: FaceNet Adaptation. ', 2' is stride parameter. **p** on pooling Inception layers contains 1x1 reduction filter. FaceNet use L2 pooling, but we use average pooling.

type	**Outputsize**	**depth**	#1 x 1	#3 x 3**reduce**	#3 x 3	#5 x 5**reduce**	#5 x 5	**Poolproj(p)**
Input	125x125x1	0						
ConvNet (7x7, 2)	60x60x64	1						
Max pool	29x29x64	0						m 3 x 3, 2
Inception (2)	29x29x192	2		64	192			
Max pool	14x14x192	0						m 3 x 3, 2
Inception (3a)	14x14x256	2	64	96	128	16	32	m, 32p
Inception (3b)	14x14x320	2	64	96	128	32	64	avg, 64p
Inception (3c)	7x7x640	2	0	128	256,2	32	64,2	m 3 x 3, 2
Inception (4a)	7x7x640	2	256	96	192	32	64	avg, 128p
Inception (4b)	7x7x640	2	224	112	224	32	64	avg, 128p
Inception (4c)	7x7x640	2	192	128	256	32	64	avg, 128p
Inception (4d)	7x7x640	2	160	144	288	32	64	avg, 128p
Inception (4e)	3x3x1024	2	0	160	256,2	64	128,2	m,128p, 2
Inception (5a)	3x3x1024	2	384	192	384	48	128	avg, 128p
Inception (5b)	3x3x1024	2	384	192	384	48	128	m, 128p
Global avgpool	1x1x1024	0						
Fullyconn	1x1x128	1						
L2 norm	1x1x128	0						

2.2. Region Selection

Knowing that YOLOv3's bounding box still has unnecessary features for FaceNet, we propose Region Selection to minimize the problem. Region Selection approach used to find **(*K*×*G*+1)** number of images snippets, hopefully our snippets will contain face with minimum noise. We

crop images using top-center crop at P centers. Region Selection generates images in size $N \times (\frac{N}{s})$, where:

$$M = \{\min(r \times w, h) | r \in R\}, \boldsymbol{K} = |R|, \boldsymbol{G} = |P|, N \in M \quad (4)$$

R contains $\boldsymbol{K}$- ratios and P is center positions where images will be cropped, both sets are obtained using KMeans *Clustering* on VOC 2012 dataset. We cluster ratio between the face's width and its annotation width. In this paper, we use = 0.76, the value we got by mean scale comparison between first 1000 CelebA preprocessed faces width and height, let see all snippets generated by Region Selection are forwarded into FaceNet as input.

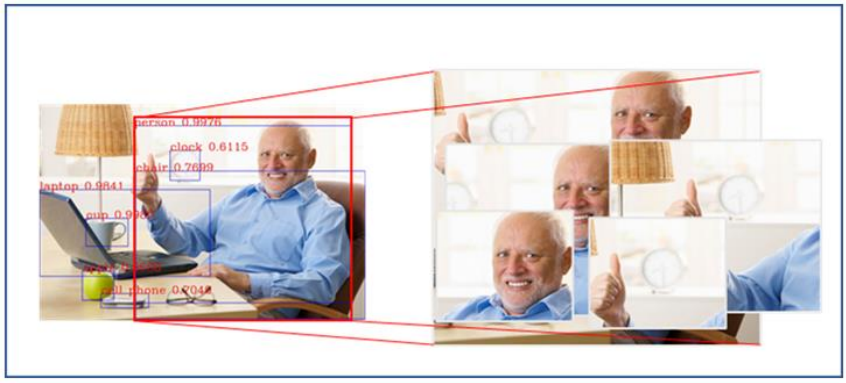


Figure 3. Region Selection with K = 2, G = 2. Snippets are generated by assuming the face might be at P positions and cropped with various ratios.

Background Complement. Region Selection imperfectly crops face from an image, resulting in only background, half face and background, and not fully cropped face snippets. This problem may increase bias of our predictions. We do Background Complement in order to decrease the bias. This approach is trying to filters Region Selection's snippets by comparing how similar the snippet is to the backgrounds database. If a snippet has a lesser distance to the backgrounds database, it would be filtered out.

3. Dataset and Evaluation

We use CelebA dan Labeled Faces in Wild (LFW). CelebA contains 202,599 faces images of 10,177 identities. It has annotations in the form of 40 binary attributes and 5 facial landmark locations. This dataset is used as a training dataset and validation with a 4:1 split size identity [15]. We validate the FaceNet model's accuracy using 500 pairs of positive images and 500 pairs of negative images [16].

The average number of images per identity on CelebA is around 20 images. It contains uncentered faces, but the face is already upright. Each image has a 178x218 pixel size. Every image came with noise or unnecessary features like hair and background.

Figure 4. CelebA images.

LFW images are 250x250 pixels in size, the face is centrally positioned. But it still contains noises and unnecessary features.

Figure 5. LFW images.

We validate FaceNet on the LFW dataset using precision and recall metrics with positive images pairs as relevant items. For each LFW image we do a center crop with 0.5 ratios.

$$P_{tp}(t) = \{(i, j) \in P_{same}, d(x_i, x_j) \leq t\} \quad (5)$$

$$P_{fp}(t) = \{(i, j) \in P_{diff}, d(x_i, x_j) \leq t\} \quad (6)$$

$P_{tp}(t)$ are images pairs that are predicted with distances less than equal to *t,* where pairs are P_{same} elements which is positive images set. P_{fp}(t)are pairs that predicted as same instead of different P_{diff} is different pairs set.
Precision and recall are defined as:

$$Precision(t) = \frac{|P_{tp}(t)|}{|P_{tp}(t)|+|P_{fp}(t)|} \quad (7)$$

$$Recall(t) = \frac{|P_{tp}(t)|}{|P_{same}|} \quad (8)$$

3.1. Preprocessing and Data Augmentation
Noise problem on CelebA dataset can easily eliminated using face detection Multi-task CNN (MTCNN). It is proved can predict facial landmark and bounding box of face up to 94% accuracy [17]. We use the python version of MTCNN on the Tensorflow framework which can be found here [18]. We executed MTCNN pre-processing for 2 days to all CelebA images. The process time quite long because of unoptimized algorithms, like using for loop and not using all GPU/CPU capability.

Figure 6. MTCNN preprocessing result sample.

To improve generalization, we do data augmentation when training [19]. Each CelebA's image produces 4 new images. The manipulations performed include random vertical/horizontal flip, random image rotation, and random brightness. Manipulations are randomly chosen, hopefully we get various number times of manipulations and differently function used to manipulate. We ran our unoptimized data augmentation algorithm for 13 hours.

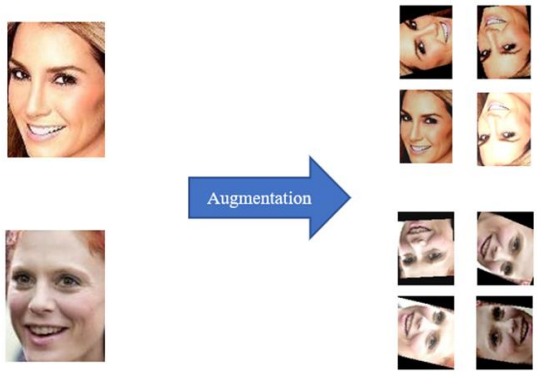
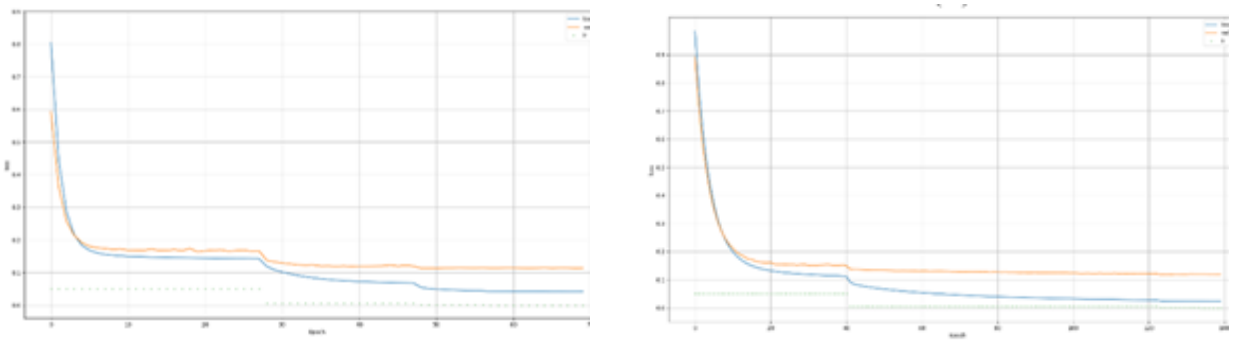


Figure 71. Images Augmentation.

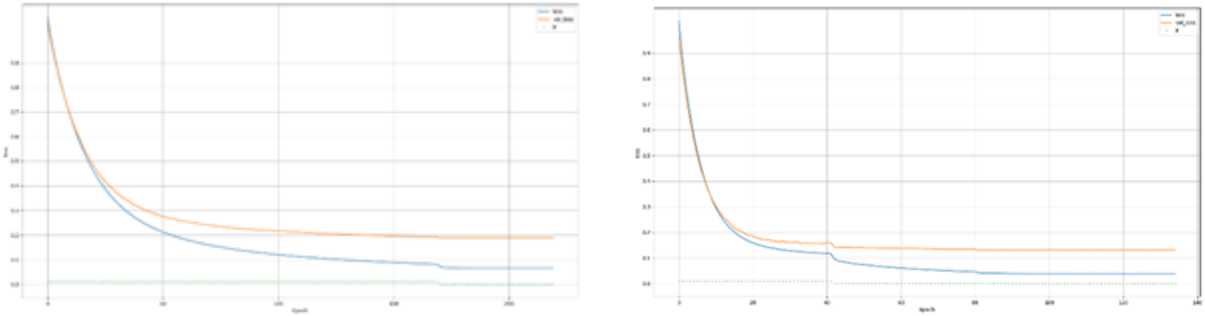
4. Experiment

FaceNet training is very dependent on batch size parameters and images on each batch. We use a batch size of 50 images. At the first, we group images by identity then arrange them into 5 images per batch. The batches are scrambled then we arrange 10 batches to become the final batch that will be used to train our model. NoonGil Lens^{+} is trained using Tesla P4 for about 1460 seconds per epoch. We train the model until its validation loss is convergence. Our result showed that Adagrad converged faster than SGD.



(A) (B)

Figure 8. Adagrad with 0.05 initial learning ratio (A) and SGD (B).



(A) (B)

Figure9. Adagrad 0.01(A) and SGD 0.01(B). Each training spikily shown decreasing loss when the learning rate is dropped.

4.1. Noongil Lens^{+} Validation

Noongil has accuracy up to 75.2% with 85.5% accuracy in distinguishing 'people' in the database or not and 67.9% accurate in predicting people in the database using the region selection **K** = 5 and **G** = 3.

OpenCV Faces Cascade. Is a machine learning-based pattern recognition. The original text for the Cascade Classifier can be found in [20]. OpenCV's built-in Faces Cascade implementation makes it difficult to detect faces with certain angles, thus reducing Noongil's accuracy. If no face is detected, the whole "person" image will be forwarded to FaceNet.

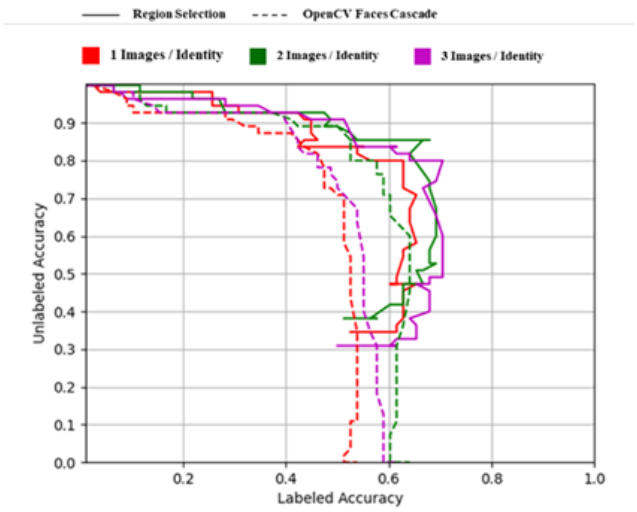


Figure 10. Comparison of model accuracy based on **Region Selection** and **Cascade OpenCV Faces** in a database with 1-3 images per identity. **Region Selection** using 2 background complement images.

Images on the database determine the accuracy. The greater the number of images per identity, the better the accuracy is expected, but it will make the execution times longer without a significant increase in accuracy, for example in a database with 2 images per identity and 3 images per identity, there is no significant difference inaccuracy. Noongil has not been able to distinguish images with significantly different expressions and too many different facial ages. In the OpenCV Faces Cascade 3 images method, the accuracy decreased. Region Selection has better accuracy than Cascade Faces.

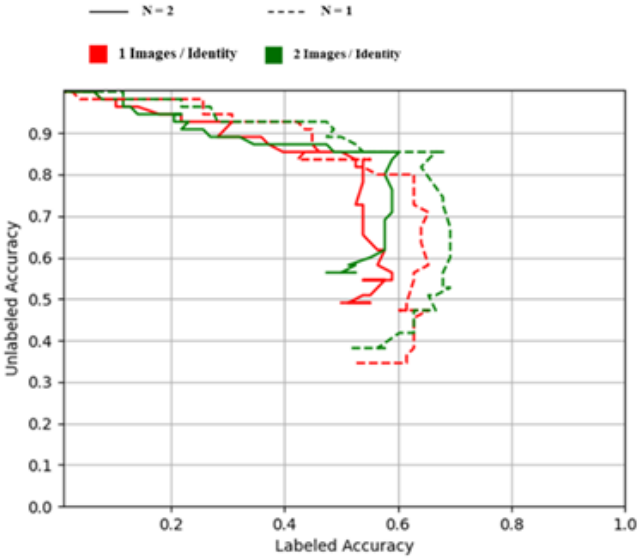


Figure 11. Comparison of accuracy for different N.

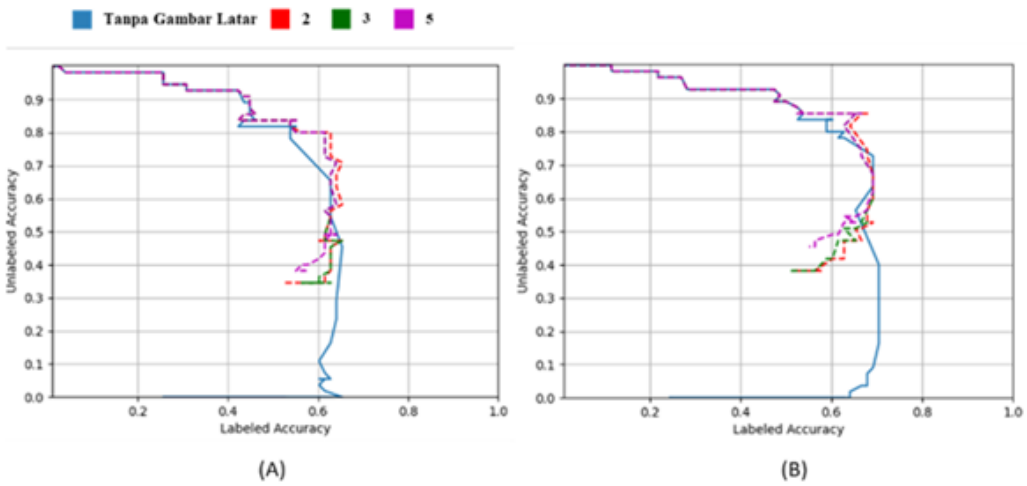


Figure 12. How background affect the model accuracy, (A) 1 image/id database & (B) 2 images/id database.

We need to choose the proper background images, in order to achieve maximum accuracy using a minimum number of backgrounds and to speed up the model execution times.

4.2. Runtime Efficiency

Table 2. Comparison of Noongil's execution speed on an image with one object of the person on various images per identity databases. **YOLOv3** execution times are **107ms**.

Images/Identity	Background Complement			Faces Cascade
	0	2	5	
1	**162ms**	**162ms**	163ms	168ms
2	**164ms**	**164ms**	166ms	169ms
3	**165ms**	167ms	170ms	169ms

NoongilLens^{+} using Region Selection relatively faster than Faces Cascade. The weakness of Region Selection is that the execution slows down as the number of background complement and per-identity images increases. Implementation of Euclidean distance and final classification is done using python without optimization, so it takes more CPU computation time.

4.3. FaceNet Validation

FaceNet validation is performed on the epoch when the initial convergent validation loss occurs to avoid an overfitting model. Only even iterations are selected because each model is stored every even iteration using the checkpoint method. Adagrad 0.05 shows the best accuracy results up to precision (0.90) and recall (0.91), while Adagrad 0.01 has the worst accuracy. The level of accuracy is directly proportional to the validation loss during training. Adagrad 0.01 anomaly occurred during training, this could be caused by the random state of the dataset. The author did not retrain due to limited times problems, and this can be confirmed on the SGD graph, which has almost the same performance with a different initial learning rate, which means that the initial learning rate does not affect model performance but has an impact on the length of the training process to achieve convergence.

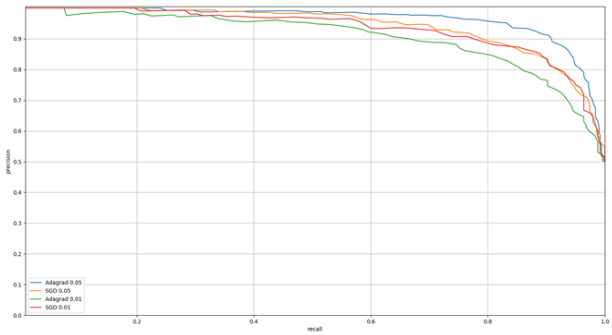


Figure 13. Comparison of the level of accuracy on the optimization method and the learning rate [epoch]. **Adagrad 0.05 [50]; SGD 0.05 [128]; Adagrad 0.01 [168]; SGD 0.01 [122]**.

4.4. Region Selection Ratios & Center Positions

The number of ratios and positions determines the accuracy and speed of execution. In a certain number, the ratio and position reach optimal accuracy because there is no significant difference between centroids.

Table 3. Centroid on K-items ratios and G-items center positions. Noongil is using K = 3 and G = 5, to make sure no overlapping centroids and as different as possible.

Ratio	Position
{0.22, 0.41}	{0.35, 0.61}
{0.19, 0.32, 0.53}	{0.28, 0.49, 0.69}
{0.17, 0.27, 0.38, 0.6}	{0.24, 0.41, 0.55, 0.73}
{0.16, 0.24, 0.32, 0.43, 0.63}	{0.23, 0.38, 0.5, 0.62, 0.77}

5. Conclusion

In this paper, we propose an approach to solve object detection and face recognition problems in a system to reduce time and accuracy trade-offs. The system is named as Noongil Lens^{+}. Noongil using Region Selection has better performance than using Faces Cascade in terms of accuracy and speed. Noongil accuracy can be improved by increasing K or G, but it can decrease system speed. The execution speed will also decrease if there are more identities in the database. To solve this problem, I will later try to develop Noongil using RPN and Anchor Boxes to improve model performance and speed.

References

[1] Pertuni, "Siaran Pers: Peran Strategis Pertuni Dalam Memberdayakan Tunanetra Di Indonesia.," 2017. [Daring]. Tersedia pada: https://pertuni.or.id/siaran-pers-peran-strategis-pertuni-dalam-memberdayakan-tunanetra-di-indonesia/. [Diakses: 28-Apr-2021].

[2] J. Redmon, S. Divvala, R. Girshick, dan A. Farhadi, "You only look once: Unified, real-time object detection," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, hal. 779–788, 2016.

[3] F. Schroff, D. Kalenichenko, dan J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 07-12-June, hal. 815–823, 2015.

[4] R. Girshick, J. Donahue, T. Darrell, dan J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 1, hal. 580–587, 2014.

[5] E. Hanna dan M. Cardillo, "Faster R-CNN2015," *Biol. Conserv.*, vol. 158, hal. 196–204, 2013.

[6] J. Redmon, "[Yolov3]YOLOv3: An Incremental Improvement," 2017.
[7] J. Redmon dan A. Farhadi, "Yolo V2.0," *Cvpr2017*, no. April, hal. 187–213, 2017.
[8] X. Long *et al.*, "PP-YOLO: An effective and efficient implementation of object detector," *arXiv*, 2020.
[9] S. Chopra, R. Hadsell, dan Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," *Proc. - 2005 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognition, CVPR 2005*, vol. I, no. May 2014, hal. 539–546, 2005.
[10] I. Melekhov, J. Kannala, dan E. Rahtu, "Siamese network features for image matching," *Proc. - Int. Conf. Pattern Recognit.*, vol. 0, hal. 378–383, 2016.
[11] "YOLOv3 Tensorflow 2.0 Implementation." [Daring]. Tersedia pada: https://github.com/zzh8829/yolov3-tf2.
[12] S. Ioffe dan C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *32nd Int. Conf. Mach. Learn. ICML 2015*, vol. 1, hal. 448–456, 2015.
[13] C. Cortes, G. Research, dan N. York, "L 2 Regularization for Learning Kernels," *Proc. Twenty-Fifth Conf. Uncertain. Artif. Intell.*, 2004.
[14] C. Szegedy *et al.*, "Going Deeper with Convolutions," 2015.
[15] Z. Liu, P. Luo, X. Wang, dan X. Tang, "Deep learning face attributes in the wild," *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2015 Inter, hal. 3730–3738, 2015.
[16] G. B. Huang, M. Mattar, T. Berg, E. L. Labeled, R. Images, dan E. Learned-miller, "Labeled Faces in the Wild : A Database for Studying Face Recognition in Unconstrained Environments," *Work. faces in'Real-Life'Images Detect. alignment, Recognit.*, 2008.
[17] K. Zhang, Z. Zhang, Z. Li, dan Y. Qiao, "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks," *IEEE Signal Process. Lett.*, vol. 23, no. 10, hal. 1499–1503, 2016.
[18] I. Centeno, "MTCNN." [Daring]. Tersedia pada: https://pypi.org/project/mtcnn/.
[19] J. Wang dan L. Perez, "The effectiveness of data augmentation in image classification using deep learning," *arXiv*, 2017.
[20] M. J. Viola, Paul; Jones, "Robust Real-Time Face Detection," *Int. J. Comput. Vis.*, hal. 137–154, 2004.