ILKOM Jurnal Ilmiah Vol. 15, No. 2, August 2023, pp. 358-364
Accredited 2nd by RISTEKBRIN No. 200/M/KPT/2020; E-ISSN 2548-7779 | P-ISSN 2087-1716

Classifying BISINDO Alphabet using TensorFlow Object Detection API

Lilis Nur Hayati [a,c,1,*]; Anik Nur Handayani [a,2]; Wahyu Sakti Gunawan Irianto[a,3]; Rosa Andrie Asmara[b,4]; Dolly Indra [c,5]; Muhammad Fahmi [c,6]

[a] *Universitas Negeri Malang, Jl. Cakrawala No.5, Sumbersari, Kec. Lowokwaru, Kota Malang, 65145, Indonesia*
[b] *Politeknik Negeri Malang, Jl. Soekarno Hatta No.9, Kota Malang, 65141, Indonesia*
[c] *Universitas Muslim Indonesia, Jl. Urip Sumoharjo No.km.5, Panaikang, Kec. Panakkukang, Kota Makassar, 90231, Indonesia*
[1] *lilis.nurhayati.2205349@students.um.ac.id;* [2] *aniknur.ft@um.ac.id;* [3] *wahyu.sakti.ft@um.ac.id;* [4] *rosa_andrie@polinema.ac.id;*
[5] *dolly.indra@umi.ac.id;* [6] *mfahmi.anton@gmail.com.*
** Corresponding author*

Article history: Received May 05, 2023; Revised June 11, 2023; Accepted July 17, 2023; Available online August 16, 2023.

Abstract

Indonesian Sign Language (BISINDO) is one of the sign languages used in Indonesia. The process of classifying BISINDO can be done by utilizing advances in computer technology such as deep learning. The use of the BISINDO letter classification system with the application of the MobileNet V2 FPNLite 320 × 320 SSD model using the TensorFlow object detection API. The purpose of this study is to classify BISINDO letters A-Z and measure the accuracy, precision, recall, and cross-validation performance of the model. The dataset used was 4054 images with a size of 320 × 320 consisting of 26 letter classes, which were taken by researchers by applying several research scenarios and limitations. The steps carried out are: dividing the ratio of the simulation dataset 80:20, and applying cross-validation (k-fold = 5). In this study, a real time testing using 2 scenarios was conducted, namely testing with bright light conditions of 500 lux and dim light of 50 lux with an average processing time of 30 frames per second (fps). With a simulation data set ratio of 80:20, 5 iterations were performed, the first iteration yielded a precision result of 0.758 and a recall result of 0.790, and the second iteration yielded a precision result of 0.635 and a recall result of 0.77, then obtained an accuracy score of 0.712, the third iteration provides a recall score of 0.746, the fourth iteration obtains a precision score of 0.713 and a recall score of 0.751, the fifth iteration gives a precision score of 0.742 for a fit score case and the recall score is 0.773. So, the overall average precision score is 0.712 and the overall average recall score is 0.747, indicating that the model built performs very well.

Keywords: Artificial Intelligence; BISINDO; Computer Vision; Real-time; TensorFlow Object Detection API.

Introduction

Humans are living beings who need the help of others and interaction between the two humans is needed. As social beings, humans use the medium of interaction to communicate [1]. Communication is the process of exchanging information between sender and receiver carried out by two or more people so that the information in question can be understood. One of the communication media used by humans is sign language [2].

Sign language is a language that does not use sounds and the meaning of the information refers to certain hand gestures that have been made into mutual agreement in an effort to facilitate communication between deaf people. Although sign language can facilitate communication between deaf people, sign language itself is still difficult for the general public to understand [3].

Sign language is different in each country, because Indonesia itself has two types of sign language, namely *Bahasa Isyarat Indonesia* (BISINDO) and *Sistem Bahasa Isyarat Indonesia* (SIBI) [4]. SIBI is a sign language established by the Indonesian government since 1994 as the language of instruction in Special Education (SLB) [5].

However, since the establishment of SIBI, there are still many deaf people who feel limited communication and prefer BISINDO developed by deaf people through the Indonesian Deaf Welfare Movement (GERKATIN) [6]. Therefore, research on the introduction of BISINDO to the general public is needed to facilitate the exchange of information between deaf and non-deaf.

http://dx.doi.org/10.33096/ilkom.v15i2.1692.358-364

Research on sign language detection using BISINDO is still very limited and the methods used require classifiers and depend heavily on the accuracy of the characteristics used in the extraction of characteristics. Therefore, the authors in this study will develop it with the recognition of hand gestures using deep learning [7].

Deep learning is one of the fastest growing and adaptable technologies and is often used in many areas of speech recognition, image processing, graphics, medicine, computer vision, and more [8]. A common challenge lies in the need for big and complex data to solve problems effectively with deep learning. Deep learning has become a very important part of image processing [9] with many methods such as Fourier descriptor, [10], K-Means [11], chain code contour [12], Euclidean distance [13] and GLCM features [14] all used to detect objects. Object detection systems are designed to detect real-world objects using object models. Object detection involves several key elements such as datasets, algorithms, and class assignment techniques.

Some research using deep learning frameworks such as the TensorFlow object detection API is limited to specific object detection, including traffic light detection [15] and handheld weapon detection [16]. Likewise, in other studies that use the TensorFlow object detection API for vehicle detection [17], [18].

TensorFlow object detection API is an object detection system development platform developed by Google [19]. Some results from the study show that the results obtained are very good in classification accuracy but still few apply this TensorFlow Object Detection API to detect sign language. Therefore, in this study developed a classification of BISINDO letters using TensorFlow object detection API.

With this TensorFlow object detection API, images can be deployed at scale. This API provides an interface that makes it easy to develop and train object detection models through TensorFlow, a leading machine learning framework [20]. It also provides implementations of multiple trained object detection models such as Faster R-CNN, R-FCN, and SSDs, and makes it easier to develop more complex object detection models by leveraging the ability of TensorFlow to train models with big data [21]. Therefore, with this TensorFlow object detection API, it is very promising to use in this research.

Methods

A. Stages of Research

The study began by collecting images from 1560 BISINDO images collected from various backgrounds and classified into 26 folders of letters A-Z, each containing 60 different images. Then resize the image to 640×640 and preprocess it so that the images have the same aspect ratio by whitening the edges. The model used is the Mobile Net V2 FPN Lite 320×320 SSD Model pre-trained from TensorFlow object detection. Next, model evaluation is performed using confusion matrix and cross-validation to obtain accuracy, precision, and memory values. The stages of research are shown in **Figure 1**.

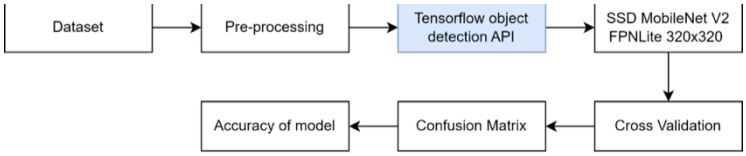

Figure1. Stages of Research

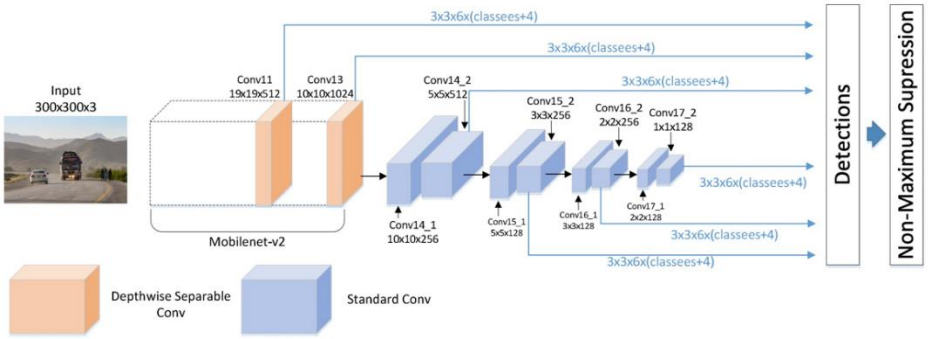

Figure 2. Network architecture of MobileNet-SSD [22]

Figure 2 shows the MobileNet SSD network architecture, which uses a second-generation MobileNet network, called MobileNet-v2, as the backbone network model for the SSD detector [22]. The MobileNet-SSDv2 detector not

only retains the advantages of the fast processing of the original MobileNet-SSD, but also greatly improves detection accuracy. This advantage shows that MobileNet-SSDv2 is suitable for use in this research.

B. Model Planning

The planning model that is used in this study is shown in **Figure 3** which consists of stages in conducting the training and testing process.

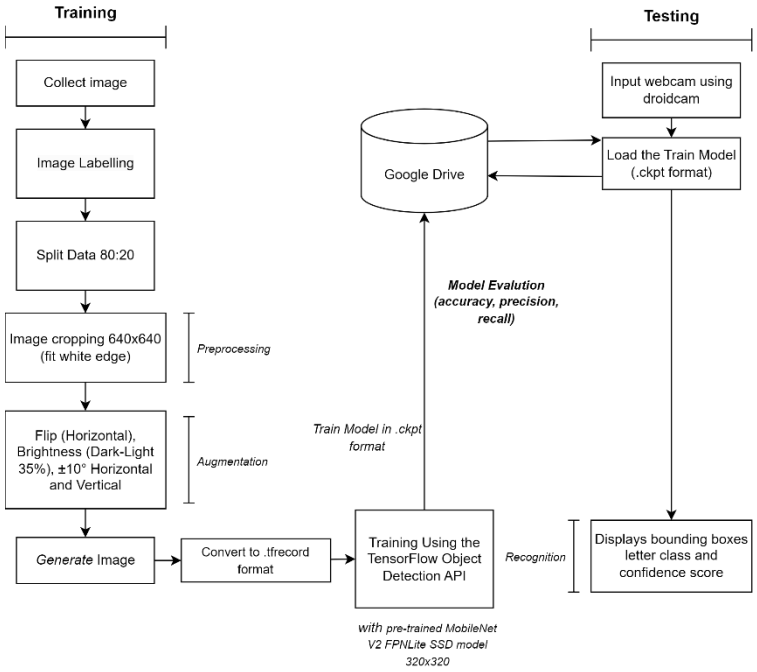

Figure 3. Model Planning

1) Collecting images

The first process is to collect image data based on 26 BISINDO letter classes. The total number of satellite imagery collected was 1560. The picture was taken using the iPhone camera and the actor himself is the author. The position for the shoot is 70 cm facing the front of the camera. The imagery varies from the background in the form of a black T-shirt and gray-and-white patterned walls, the perspective of the hands only and also from the head to the body and the dim and bright indoor lighting.

2) Image labelling

Next is the labelling or annotation stage. The collected images will be annotated using Roboflow. Roboflow has several useful features to facilitate the process of creating datasets such as preprocessing, augmentation and one of them is labeling image data so that it can be used at the training stage.

3) Splitting data

The next process is to split data on labeled images, split data with a ratio of 80:20, where 80% is training data and 20% is test data. All images, namely 1560, are divided equally into 60 images of each class A-Z with the aim of reducing the risk of overfitting.

4) Preprocessing

Further pre-processing of the image by resizing the image from 1920×1080 to 640×640 (fit white edge) to ensure that the image has the same size and reducing computing power and making the white edge is done to ensure the image has the same ratio 1:1, pre-processing is done with the Roboflow application. The reason the researchers chose 640×640 was to match the resolution of the webcam to be used during testing.

5) Augmentation

After the pre-processing process, it is performed image augmentation using Roboflow from image 1560 to 4054 by providing variations of data to be used by the model so as to reduce overfitting, increase the amount of data, reduce the potential need for new data, and of course expand the scope of data with many variations of data used. The augmentation used is flip, brightness from -35% to +35%, and also shear with ±10° Horizontal, ±10° Vertical.

6) Generating dataset

The generation process is done by converting the records to a file in '.tfrecord' format in the Roboflow application. This is the file format used by TensorFlow for training and evaluation and stored in Google Drive.

7) Training using the TensorFlow object detection API

At this stage, the author will conduct a training and evaluation process using the TensorFlow object detection API framework with the latest version v2.11.0 and Pretrained TensorFlow 2 ZOO model, namely Mobile Net V2 FPN Lite 320 × 320 SSD in Google COLAB shown in **Table 1**.

Table 1. Pretrained TensorFlow 2 model ZOO

Model	Speed (ms)	COCO mAP
CenterNet HourGlass104 512x512	70	41.9
EfficientDet D1 640x640	54	38.4
EfficientDet D2 768x768	67	41.8
EfficientDet D3 896x896	95	45.4
SSD MobileNet V1 FPN 640x640	48	29.1
SSD MobileNet V2 FPNLite 320x320	**22**	**22.2**
SSD ResNet101 V1 FPN 640x640 (RetinaNet101)	57	35.6
SSD ResNet152 V1 FPN 640x640 (RetinaNet152)	80	35.4
Faster R-CNN ResNet50 V1 640x640	53	29.3
Faster R-CNN ResNet50 V1 800x1333	65	31.6
Faster R-CNN ResNet101 V1 640x640	55	31.8
Faster R-CNN ResNet101 V1 800x1333	77	36.6
Faster R-CNN ResNet152 V1 800x1333	101	37.4
Faster R-CNN Inception ResNet V2 640x640	206	37.7

Based on **Table 1**, the analysis results show that MobileNet V2 FPNLite 320 × 320 SSD as an object detection model is feasible for real-time use because it has a superior speed with a time of 22 ms and a mAP value of 22.2, which shows superior performance on object detection tasks. In addition, other trained models have resolutions greater than 320 × 320 which can potentially affect the length of training time. The file used in the training process is an export from Roboflow, i.e. ". tfrecord". Next, the configuration is carried out on the "pipeline. config" that specifies the number of batch sizes and also the optimal training steps for the speed of the training process and superior model performance. After the training process, the model is obtained in .ckpt format and saved to Google Drive for real-time testing using OpenCV on Jupyter Notebook. An evaluation process is then run on the model and the output is displayed in the form of precision and recall values.

8) Model Evaluation

Confusion matrix is a method to measure object detection performance such as accuracy, precision, recall [8]. Accuracy is obtained based on the ratio of correct predictions to overall data. Precision is obtained based on the ratio of correct positive predictions compared to overall positive prediction results. Remember it is based on a positive true prediction ratio compared to overall correct data. In the confusion matrix, there are four terms or four possible decisions that may occur in detecting objects, such as True Positive (TP), False Positive (FP), False Negative (FN) and True Negative (TN). The following calculation of accuracy, precision and recall uses the confusion matrix equation shown in **Figure 4** with the values of accuracy, precision and recall shown in Equations 1, 2 and 3.

		Actual Class	
		Positive (**P**)	Negative (**N**)
Predicted Class	Positive (**P**)	True Positive (**TP**)	False Positive (**FP**)
	Negative (**N**)	False Negative (**FN**)	True Negative (**TN**)

Figure 4. Confusion Matrix

$$Accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN} + \mathrm{TN}} \quad (1)$$

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \quad (2)$$

$$Recall = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (3)$$

9) Webcam input

Webcam input is provided by connecting a 12-megapixel iPhone camera to a computer using Droidcam, which is used for model testing purposes.

10) Load the model

After the webcam input stage, the author took the ".ckpt" model stored in Google Drive, then used it for real-time testing using OpenCV and Jupyter Notebook. The computers used in these tests have the specifications shown in **Table 2**.

Table 2. Computer Specification

Hardware	Specification
CPU	AMD Ryzen 5 2600 3.4 GHz 6-Core Processor
RAM	16 GB (2 x 8 GB) DDR4-3200 MHz
GPU	GTX 980 4 GB Vram
Storage	1 TB NVME

11) Recognition

After the train model is loaded, the system will display a bounding box where there is a BISINDO letter class along with a confidence score on the OpenCV interface.

Results and Discussion

A. Model evaluation

Model evaluation is carried out using cross-validation k = 5 fold, where the dataset used is 80:20 split data between training data and testing data. The evaluation process includes 5 training iterations that will provide evaluation results in the form of Average Precision (AP) and Average Recall (AR). The methods used to evaluate the model are AP @[IoU=0.50:0.95 | area= all | maxDets=100] and AR @[IoU=0.50:0.95 | area= all | maxDets= 1]. AP measures the average precision of all bounding boxes generated by the model, while AR measures how well the model finds objects in the image. Here are the evaluation results for each iteration shown in **Table 3**.

Table 3. Cross-validation K=5

Cross-validation	Precision	Recall
Iteration 1	0.758	0.79
Iteration 2	0.635	0.677
Iteration 3	0.712	0.746
Iteration 4	0.713	0.751
Iteration 5	0.742	0.773
Average	0.712	0.7474

By doing cross-validation, the average value is obtained, namely, AP with a value of 0.712 or 71% and AR with a value of 0.7474 or 75%.

B. Model testing

The system was tested using an iPhone camera with the OpenCV library in 10 real-time tests through self-tests, other tests, tests with different backgrounds, and tests with accessories. All but the tests were performed at a distance of 70 cm from the camera. Scenarios 8 and 10 at a distance of 140 cm. The minimum score threshold on the test is 0.3

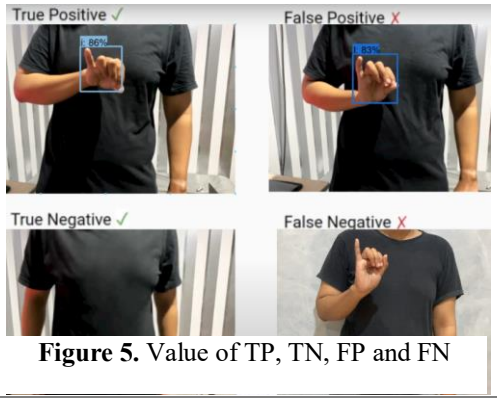

Figure 5. Value of TP, TN, FP and FN

or 30%. Each test was conducted under two conditions, namely in a bright place with 500 lux and dim with 50 lux. The results of model evaluation testing using the confusion matrix will obtain the accuracy, precision and recall values shown in **Figure 5**.

As for the illustration in **Figure 5**, a TP is determined if the character predicted by the model matches the character it should have. TN is determined if the predicted letter does not appear and as it should. While FP is determined if as shown above the predicted letter is the letter L but not as it should be, the letter I. And FN is determined if the letter is not displayed or not detected. Here are the test results with calculation of TP, TN, FP, and FN values can be seen in **Table 4**.

Table 4. Results of research scenario evaluation

No.	Scenario	Accuracy	Precision	Recall
1.	Self-Testing (bright)	0.963	0.963	0.963
2.	Self-Testing (dimmed)	0.907	0.893	0.926
3.	Testing others (bright)	0.942	0.926	0.962
4.	Testing others (dimmed)	0.769	0.850	0.654
5.	Testing with different backgrounds (bright)	0.845	0.813	0.897
6.	Testing with different backgrounds (dimmed)	0.800	0.758	0.862
7.	Testing using accessories (bright)	0.857	0.862	0.862
8.	Testing using accessories (dimmed)	0.833	0.852	0.821
9.	Testing with a distance of 140 cm (bright)	0.804	0.714	0.833
10.	Testing with a distance of 140 cm (dim)	0.791	0.682	0.882

Based on testing with 5 scenarios, namely self-testing scenarios, testing others, testing with different backgrounds, testing using accessories and testing with a distance of 140 cm with bright and dim lighting shown in **Table 4**. For scenario 1, it produces good values on accuracy, precision and recall for self-testing scenarios in bright conditions. For scenario 2, it produces good values on accuracy, precision and recall for other testing scenarios in bright conditions. For scenario 3, it produces good values on accuracy, precision and recall for testing scenarios with different backgrounds in bright conditions.

For scenario 4, it produces good values on accuracy, precision and recall for testing scenarios using accessories in bright conditions. For scenario 5 it produces good values on accuracy, precision and recall for the testing scenario with a distance of 140 cm in bright conditions.

Conclusions

BISINDO character classification performed using the TensorFlow Object Detection API on a MobileNet V2 FPNLite 320 × 320 SSD model at a time rate of 33 ms using real-time testing with model evaluation using a 5-fold cross-validation method. The result showed that the average precision score is 0.712 and the average recall is 0.7474, indicating that the model built performs reasonably well. Based on the testing result, all 5 scenarios which are self-testing, testing others, testing in different backgrounds, testing with accessories and at a distance of 140 cm obtain superior values in bright light conditions (500 lux). In dark conditions, the result is still not promising. For our future work, it will be tried to maximize the models for dim conditions or lack of lighting.

References

[1] V. Kissya, "Penggunaan Bahasa Isyarat Dalam Komunikasi Antara Penyandang Tuna Rungu, Guru, Serta Keluarga Di (Sekolah Luar Biasa Pelita Kasih) Rumah Tiga Ambon," vol. 16, no. 1, 2022.

[2] A. Dicky Saputra and I. B. Artambo Pangaribuan, "Klasifikasi Alfabet Bahasa Isyarat Indonesia (BISINDO) dengan Metode Template Matching dan K-Nearest Neighbors (KNN)," 2020.

[3] A. Wadhawan and P. Kumar, "Sign Language Recognition Systems: A Decade Systematic Literature Review," *Archives of Computational Methods in Engineering*, vol. 28, no. 3, pp. 785–813, May 2021, doi: 10.1007/s11831-019-09384-2.

[4] A. Sri Nugraheni, A. Pratiwi Husain, and H. Unayah, "Optimalisasi Penggunaan Bahasa Isyarat dengan SIBI Dan BISINDO pada Mahasiswa Difabel Tunarungu di Prodi PGMI UIN Sunan Kalijaga," 2021.

[5] T. Handhika, R. I. M. Zen, Murni, D. P. Lestari, and I. Sari, "Gesture recognition for Indonesian Sign Language (BISINDO)," in *Journal of Physics: Conference Series*, Institute of Physics Publishing, Jun. 2018. doi: 10.1088/1742-6596/1028/1/012173.

[6] R. Z. Fadillah, A. Irawan, M. Susanty, and I. Artikel, "Data Augmentasi Untuk Mengatasi Keterbatasan Data Pada Model Penerjemah Bahasa Isyarat Indonesia (BISINDO)," *Jurnal Informatika*, vol. 8, no. 2, 2021, [Online]. Available: http://ejournal.bsi.ac.id/ejurnal/index.php/ji

[7] N. O' Mahony *et al.*, "Deep Learning vs. Traditional Computer Vision," 2020.

[8] S. Dong, P. Wang, and K. Abbas, "A survey on deep learning and its applications," *Computer Science Review*, vol. 40. Elsevier Ireland Ltd, May 01, 2021. doi: 10.1016/j.cosrev.2021.100379.

[9] D. Indra, H. M. Fadlillah, Kasman, L. B. Ilmawan, and H. Lahuddin, "Classification of good and damaged rice using convolutional neural network," *Bulletin of Electrical Engineering and Informatics*, vol. 11, no. 2, pp. 785–792, Apr. 2022, doi: 10.11591/eei.v11i2.3385.

[10] S. E. Basri, D. Indra, H. Darwis, A. W. Mufila, L. B. Ilmawan, and B. Purwanto, "Recognition of Indonesian Sign Language Alphabets Using Fourier Descriptor Method," in *3rd 2021 East Indonesia Conference on Computer and Information Technology, EIConCIT 2021*, Institute of Electrical and Electronics Engineers Inc., Apr. 2021, pp. 405–409. doi: 10.1109/EIConCIT50028.2021.9431883.

[11] D. Indra, R. Satra, H. Azis, A. R. Manga, and H. L, "Detection System of Strawberry Ripeness Using K-Means," *ILKOM Jurnal Ilmiah*, vol. 14, no. 1, pp. 25–31, Apr. 2022, doi: 10.33096/ilkom.v14i1.1054.25-31.

[12] D. Indra and S. Madenda, "Feature Extraction of Bisindo Alphabets Using Chain Code Contour," *Int J Eng Technol*, Aug. 2017, doi: 10.21817/ijet/2017/v9i4/170904142.

[13] D. Indra, S. Madenda, and E. P. Wibowo, "Recognition of Bisindo alphabets based on chain code contour and similarity of Euclidean distance," *Int J Adv Sci Eng Inf Technol*, vol. 7, no. 5, pp. 1644–1652, 2017, doi: 10.18517/ijaseit.7.5.2746.

[14] D. Indra, H. M. Fadlillah, Kasman, and L. B. Ilmawan, "Rice Texture Analysis Using GLCM Features," in *International Conference on Electrical, Computer, and Energy Technologies, ICECET 2021*, Institute of Electrical and Electronics Engineers Inc., 2021. doi: 10.1109/ICECET52533.2021.9698594.

[15] T. V. Janahiraman and M. S. M. Subuhan, "Traffic light detection using TensorFlow object detection framework," in *2019 IEEE 9th International Conference on System Engineering and Technology, ICSET 2019 - Proceeding*, Institute of Electrical and Electronics Engineers Inc., Oct. 2019, pp. 108–113. doi: 10.1109/ICSEngT.2019.8906486.

[16] Christiani I, "Deteksi Senjata Genggam Menggunakan Faster R CNN Inception V2," 2022.

[17] Iskandar D, "Implementasi Framework Mask R-CNN Object Detection API Dalam Mengklasifikasi Jenis Kendaraan Bermotor," 2022.

[18] D. Indra and F. Shantya Budi, "Implementasi Sistem Penghitung Kendaraan Otomatis Berbasis Computer Vision Implementation Of Automatic Vehicles Counting System Based On Computer Vision," *Jurnal Sistem Komputer*, vol. 12, no. 1, p. 2020, 2023, doi: 10.34010/komputika.v12i1.9082.

[19] Ratna P, Sumin A, and Wirawan S, "Pembuatan Aplikasi Deteksi Objek Menggunakan TensorFlow Object Detection API dengan Memanfaatkan SSD MobileNet V2 Sebagai Model Pra - Terlatih," *Jurnal Ilmiah Komputasi*, vol. 19, no. 3, Mar. 2020, doi: 10.32409/jikstik.19.3.68.

[20] S. A. Sanchez, H. J. Romero, and A. D. Morales, "A review: Comparison of performance metrics of pretrained models for object detection using the TensorFlow framework," in *IOP Conference Series: Materials Science and Engineering*, Institute of Physics Publishing, Jun. 2020. doi: 10.1088/1757-899X/844/1/012024.

[21] J. Koci, A. O. Topal, and M. Ali, "Threat Object Detection in X-ray Images Using SSD, R-FCN and Faster R-CNN," in *Proceedings - 2020 International Conference on Computing, Networking, Telecommunications and Engineering Sciences Applications, CoNTESA 2020*, Institute of Electrical and Electronics Engineers Inc., Dec. 2020, pp. 10–15. doi: 10.1109/CoNTESA50436.2020.9302863.

[22] Y.-C. Chiu, C.-Y. Tsai, D. Ruan, G.-Y. Shen, and T.-T. Lee, "MobileNet-SSDv2: An Improved Object Detection Model for Embedded Systems," 2020.