Person Detection in Point-of-Sale Application at Retail Food-Outlets

Jimmy Linggarjati
Department of Computer Engineering
Bina Nusantara University
Kebon Jeruk, Jakarta, Indonesia
jimmyl@binus.edu

Abstract

Person detection is an interesting machine learning application that can be implemented in embedded devices. One such need is to automate an alarm system to detect whether a person that works in a Point of Sale is indeed working at his/her working time. This presence monitoring system can be automated by using an esp-eye development board by Espressif. In this paper, the key important aspect of modelling the person detection behavior is discussed and implemented in ESP32 technologies. The rate of accuracy in detecting a person's presence is up to 65% based on a four (4) minute window-time, and therefore it is reliable to be used as a person's monitoring automated system, based on the current requirement from the food industry. The inference time is circa 0.7 MS, that is the time between each sample in a situation whether a person is in front of the camera or not.

Keywords
ESP32, ESP-EYE, person detection, inference and Convolutional Neural Network

1. Introduction
Recently, microcontroller has gained importance in the AI (Artificial Intelligence) world, by their role as an edge computing device. Edge computing devices are devices that will work directly on its collected data. And with the advances of microcontroller, in terms of clock speed and memory, the microcontrollers are now able to solve difficult problems such as image detection, by using artificial intelligence technique called neural network.

And at the same time the term IoT has flourished, and this has revolutionised the amount of data generated from the sensors to a staggering number. And this phenomenon causes a huge problem in a data centric concept, where the data needs to be transfer via internet, before they can be further processed to get the desired information. And therefore, an edge computing concept has emerged to decrease the data traffic on the internet as well as to increase the data privacy, by making the decision right inside the embedded devices.

For structured data, such as temperature or humidity values, there exists techniques that can help generate required information from the data that are generated from the sensors. But for unstructured data, such as image, the methodology shifts to what is called neural network technique, in which the input data is fed into layers of neurons that will be able to classify the image or identify the objects inside the image.

1.1 Objectives
In this paper, there is a requirement from the food industries to monitor their employee working time, by using a camera (Kim et al. 2018). Therefore, a web camera will be used to identify if a person is seen or not at his/her workplace. The main contribution of this paper is to show that a cheap microcontroller can be used to do this monitoring job, by utilizing the neural network technology. In particular, (Howard et al. 2017) have laid a foundation for a faster model called Mobile Nets, in which it uses depth wise separable convolutions to enable DNN on an embedded platform, such as ESP32 microcontroller.

The structure of this paper will be arranged as follow, section 2 will discuss literature review. Section 3 will discuss software design from the literature's review describe in section 2. While, in section 4 the implementation will be mentioned both in hardware and software parts. And finally, section 5 will show the results.

2. Literature Review

The overall proposed system is illustrated in a block diagram, as shown in Figure 1. The Espressif company has created a development kit called ESP-EYE. This product is placed in front of the working place of a person that need to be monitored.

If a person is not detected within a certain duration of time, the system will send an email to the supervisor, alerting his/her about this situation. And since, the ESP-EYE is already equipped with a Wi-Fi module, it becomes easy to connect to a Wi-Fi router and then send an email, which in this case, it uses a Gmail's mail-server.

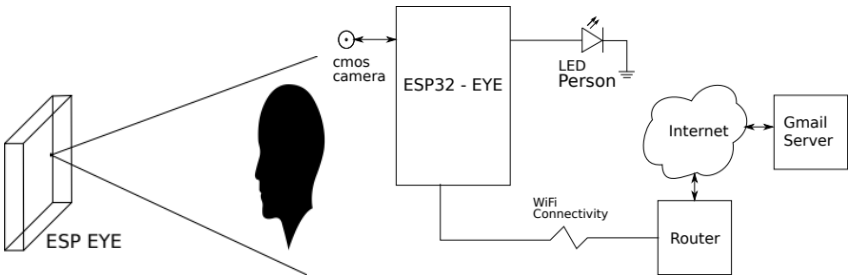

Figure 1. Proposed System using ESP-EYE

2.1 ESP-EYE

The ESP-EYE product, as shown in Figure 2, is based on the ESP32 chip, with an application dedicated to vision capability, such as detecting objects via a webcam.

Figure 2. The ESP-EYE

2.2 ESP-IDF

The ESP-IDF is the official compiler used for the ESP32 chip microcontroller that is manufactured by the Espressif company. And the ESP-EYE needs to be compiled with the ESP-IDF. There is no user IDE for this compiler. And therefore, the ESP-IDF can be used directly in command prompt, or it can be used alongside with the platform-io IDE (ESP-IDF Programming Guide).

The following command is used by the ESP-IDF to compile and flash the ESP32 chip via usb port, inside the platform-io IDE.

idf.py flash -p /dev/ttyUSB0 monitor

2.3 TensorFlow

TensorFlow is a framework for developing neural network related applications, owned by Google corp. (David et al. 2021). As its named applied, tensor is the keyword in neural network that relates heavily to the fundamental of linear algebra.

From the example given in the tensorflow.org, a person detection is of interest to us, and it relates to a couple of interesting papers. Chowdhery et al. (2019) describes how the model of person detection is generated. And it comes from the previous work (Howard et al. 2017), which is called a mobile Nets model, especially design for mobile and embedded vision applications.

Research conducted in person detection has shown tremendous advances in YOLO application (Zhao and Wan 2019), but it requires big memory both in RAM and in flash memory. Therefore, it is not suitable for the lower end of embedded chips, such as ESP32. However, TensorFlow-lite has been created to accommodate the needs for running deep-learning networks inside lower end chips, such as ESP32.

Mustamo (2018) has shown a case study for object detection using TensorFlow. Alsing (2018) has created a mobile object detection based on TensorFlow and Transfer Learning. While Demosthenous and Vassiliades have proposed Continual Learning (CL) for on-device real-time training. Li (2020) introduced on-device machine learning (ODML) for machine learning beginner. While, Dokic et al. (2020) discussed inference time of tensorflow lite in microcontroller. Jacob et al. (2018) discussed quantization for integer-arithmetic only inference.

2.4 Person/Human Detection Literatures

(Rahmaniar, et. al. 2021) has made a review about human detection on embedded platform. It stated that traditional machine learning tecniques, such as HOG (Histogram of Gradient), Haar-like features and local binary pattern (LBP) are less reliable in detecting objects in a real world application.

Picon, et. al. (2021) used CNN in Raspberry Pi 3 to detect person with a confidence level of 80%. This confidence level is higher than what is obtained in this experiment (mentioned in section 5 – Results and Discussion). But the price for Raspberry Pi 3 is higher compared to ESP32 by quite a large margin.

Khalifa, et. al. (2019) also discussed about human detection using Raspberry Pi, and mentioned that CNN requires high computational resources to generate the Tensorflow-lite model for human classification. This statement is true, but Raspberry Pi can not also generate that Tensorflow model. Nguyen, et. al. (2016) gived a full review about feature extraction that targets to human detection system, eventhough its discussions are not specifically aimed to the embedded devices. Yu, et. al. (2019) and Ahmad, et. al. (2019) discussed human detection from a top view orientation. This is useful for counting people passing through a certain gate or location. Though the aim is the same, that is to detect human, the Tensorflow model is trained for different dataset, and in their case-study, they used transfer-learning to fine tune the Tensorflow models. Saqib, et. al. (2018) gave both theoritical and practical view on how to train the model using transfer-learning, to do specific human-head detection. This is useful if there are multiple persons in the image, and it can be used to also count the number of people in the scene.

3. Methods of Software Algorithms

The design of the software algorithm is based on the examples taken from TensorFlow.org. Figure 3 shows the flowchart diagram of the software algorithm for monitoring the presence of a person in the outlet.

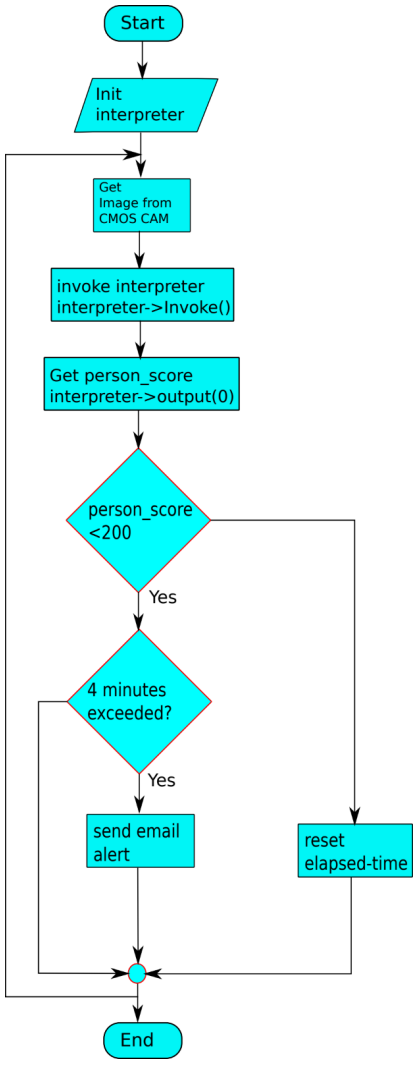

Figure 3. Flowchart Diagram of Person Monitoring System

Once the image is captured, the program invoke an interpreter function, which will return a person_score output, as an indicator if a person is detected or not (Figure 3). The number 200 represented the threshold for the decision of whether there is a person detected or not. If person_score is greater than 200 than a person is detected, and therefore an elapsed-time variable is reset, to recount from the beginning for the 4-minute window-time. If within the 4 minute window-time, there is no person detected, than an email is going to be sent with two figures (one of which there is no person, and another one which a person is last detected). The supervisor of the food outlet will therefore be informed about the situation in the food outlet, and furthermore the email data can be used further to produce meaningful information, such as the number of hours that the employee is working.

It is worth to restress that in this paper, the CNN model from the MobileNets Tensorflow-lite, is not modified or altered in any way. And therefore, this paper only uses the model, and to show that it can be used in this application for monitoring a person in his/her work station.

4. Hardware-Software Implementation and Data Collection

The prototyping of the hardware does not require any changes from the ESP-EYE development product kit, except that a person LED is added to indicate visually if a person is detected in the vicinity of the cmos-camera. While in the software parts, the original code from the tensorflow.org is altered to suit the needs for monitoring the person in the outlet, as shown in Figure 5.

4.1 Hardware

Figure 4 shows the internal of esp-eye inside a box, along with additional LEDs to visually see if the device indeed detects or otherwise it does not detect a person.

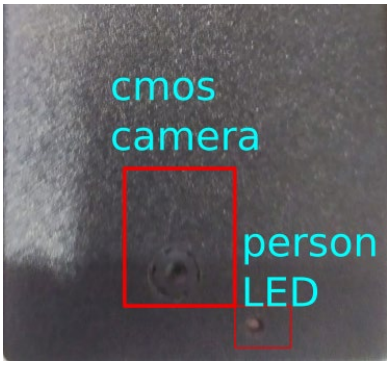

Figure 4. ESP-EYE in a Box with LED indicator

4.2 Software

The following snapshot code shows the modification that was made to suit the flowchart diagram designed in Figure 5. The elapsed-time is a variable that holds time between each image's loop detection. And it shows that when a person is detected, approximately 0.7 seconds are needed for inferencing.

```
// send email if no-person is detected for 4 minutes window-time
if (person_score < 200) {
  if (((esp_timer_get_time() - elapsed_time)/1000) >= HOLD_TIME) {

    camera_fb = (camera_fb_t*)image_provider_get_camera_fb();
    TF_LITE_REPORT_ERROR(error_reporter, "person not detected");
    free(jpeg_image);
    bool ret = frame2jpg(camera_fb, 80,  &jpeg_image, &jpeg_img_size);
    if (ret != true) {
      TF_LITE_REPORT_ERROR(error_reporter,"jpeg compression failed");
    }
    esp_err_t esp_ret = smtp_client_send_email(jpeg_image, jpeg_img_size);
    if (esp_ret != ESP_OK) {
      ESP_LOGE(TAG, "Failed to send the email, returned %02X", esp_ret);
    }
    elapsed_time = 0;
  }
} else
  if (person_score > 200) elapsed_time = 0; /* reset possible elapsed_time */
```

Figure 5. Coding to send email alarm after 4 minutes of no-person status

5. Results and Discussion

The result for detecting the person in the outlet can be shown from Table 1 and Table 2. It depicted that the person detection model from tensorflow is suitable for this application of person's monitoring system. Table 1 shows a condition in which there is no person in the vicinity of the web-camera. And the results show that within a 4-minute window-time, the software took 303 images. And all the images are inferenced by the person's detection model, as having no person inside the image.

While in Table 2, it shows a condition in which a person is in the vicinity of the web-camera. But, out of 303 images taken by the web-camera, not all of them were inferenced as having a person in the image. In fact, almost 35% were inferenced as having no person in the image. Fortunately, when the person is detected in an image, then the 4-minute window-time is reset, and therefore no email is sent as an alarm by this person's monitoring system.

Table 1. Inference made by the person's detection model in a 4-minute window-time, when there is no person.

Num. Of Get Image	Person is not on the scene	
	Threshold: <240 No Person	Threshold: >= 240 Person
1	1	0
2	1	0
3	1	0
.	.	.
.	.	.
303	1	0
Total:	303	0

Table 2. Inference made by the person's detection model in a 4-minute window-time, when there is a person.

Num. Of Get Image	Person is on the scene	
	Threshold: <240 No Person	Threshold: >= 240 Person
1	1	0
2	1	0
3	0	1
.	.	.
.	.	.
303	0	1
Total:	107	196

Figure 6 shows some snapshots from the POS (Point of Sale) in which the ESP-EYE is installed. It is interesting to see that a person can still be detected eventhough they were using mask and cap.

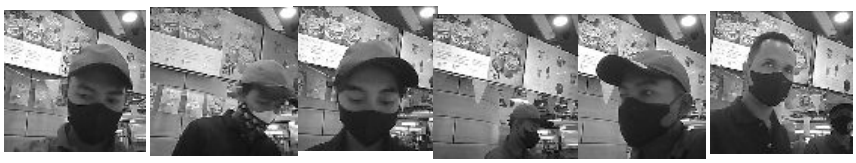

Figure 6. Snapshots Taken from ESP-EYE

6. Conclusion

The proof of working monitoring system, using a cheap esp-eye development kit, is very encouraging for developers in the embedded world, to further explore the essence of deep machine learning algorithm. In this work, the ESP-EYE has been tested in the retail-food industry, in one of our supermall buildings in Jakarta. And the result has been satisfactory for monitoring a person at his/her work's station.

Future work will focus on developing our own model, using google cloud computing resources, to improve the probability of detecting a person, by adding more training pictures, that are more suited to the environment, in which this application is used for.

References

Ahmad, M., Ahmed, I., & Adnan, A. Overhead view person detection using YOLO. In *2019 IEEE 10th annual ubiquitous computing, electronics & mobile communication conference (UEMCON)* (pp. 0627-0633). IEEE. October 2019.

Alsing, O. Mobile object detection using tensorflow lite and transfer learning, 2018.

Chowdhery, A., Warden, P., Shlens, J., Howard, A., & Rhodes, R. Visual wake words dataset. *arXiv preprint arXiv:1906.05721,* 2019.

David, R., Duke, J., Jain, A., Janapa Reddi, V., Jeffries, N., Li, J., ... & Rhodes, R. TensorFlow Lite Micro: Embedded Machine Learning for TinyML Systems. *Proceedings of Machine Learning and Systems*, *3,* 2021.

Demosthenous, G. and Vassiliades, V. Continual Learning on the Edge with TensorFlow Lite. *arXiv preprint arXiv:2105.01946,* 2021.

Dokic, K., Martinovic, M. and Mandusic, D. Inference speed and quantisation of neural networks with TensorFlow Lite for Microcontrollers framework. In *5th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM)* (pp. 1-6). IEEE, 2020.

ESP_IDF Programming Guide, Espressif Systems (Shanghai) Co., Ltd., https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/, 7 September 2022

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861,* 2017.

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., ... & Kalenichenko, D. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2704-2713), 2018.

Khalifa, A. F., Badr, E., & Elmahdy, H. N. A survey on human detection surveillance systems for raspberry pi. *Image and Vision Computing*, *85*, 1-13. 2019.

Kim, C. E., Oghaz, M. M. D., Fajtl, J., Argyriou, V., & Remagnino, P. A comparison of embedded deep learning methods for person detection. *arXiv preprint arXiv:1812.03451,* 2018.

Li, S. Tensorflow lite: On-device machine learning framework. *Journal of Computer Research and Development*, *57*(9), 1839, 2020.

Mustamo, Pirkko. "Object detection in sports: TensorFlow Object Detection API case study." *University of Oulu* 2018.

Nguyen, D. T., Li, W., & Ogunbona, P. O. Human detection from images and videos: A survey. *Pattern Recognition*, *51*, 148-175. 2016.

Picon, G., Benítez, D. S., Pérez, N., Riofrío, D., & Moyano, R. F. Towards a low-cost embedded vision-based occupancy recognition system for energy management applications. In *2021 IEEE CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)* (pp. 1-6). IEEE. December 2021.

Rahmaniar, W., & Hernawan, A. Real-time human detection using deep learning on embedded platforms: A review. *Journal of Robotics and Control (JRC)*, *2*(6), 462-468, 2021.

Saqib, M., Khan, S. D., Sharma, N., & Blumenstein, M. Person head detection in multiple scales using deep convolutional neural networks. In *2018 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-7). IEEE. July 2018.

Yu, J., Seidel, R., & Hirtz, G. Omnipd: One-step person detection in top-view omnidirectional indoor scenes. *Current Directions in Biomedical Engineering*, *5*(1), 239-244. 2019.

Zhao, L., & Wan, Y. A New Deep Learning Architecture for Person Detection. In *IEEE 5th International Conference on Computer and Communications (ICCC)* (pp. 2118-2122). IEEE, 2019.

Biography

Jimmy Linggarjati S.Kom., MSc. graduated from Bina Nusantara University in 1999 majoring in Computer Engineering and received a master's degree from TU-Delft University in 2002 majoring in Electrical Engineering. He is currently a lecturer at Bina Nusantara University. His research interests include on-device machine learning and control systems.