Performance Analysis of Deep Neural Networks for Object Classification with Edge TPU

Ahmad Ammar Asyraaf Jainuddin[1], Yew Cheong Hou[2], Mohd Zafri Baharuddin[1]*, Salman Yussof[2]
[1] College of Engineering, Universiti Tenaga Nasional, Kajang, Selangor, Malaysia
[2] Institute of Informatics and Computing in Energy, Universiti Tenaga Nasional, Kajang, Selangor, Malaysia
*Corresponding email: zafri@uniten.edu.my

***Abstract*—Deep learning becomes a more popular, widespread, and common tool in almost any task that requires information extraction from a large dataset. Hence, the data transmission speed between the data-gathering devices and processing units can be crucial in hardware selection depending on the machine learning application. Generally, the processing unit is usually centralized, and the data transferring time will increase when the data-gathering devices were installed further away from the processing unit. The work aims to provide the performance analysis on Google's new machine learning hardware called Edge TPU that was created specifically for edge devices. Furthermore, the work also reviewed the different types of deep neural network models as current benchmarks in deep learning were tested with different hardware used in edge applications. The review also discussed the comparison of the performance of the edge device using the deep neural networks in Tensorflow. From the results obtained, the performance of the edge device with the Edge TPU is faster than the device without it.**

Keywords - Edge TPU, Deep Learning Model, Tensorflow, Machine Learning.

I. INTRODUCTION

Artificial intelligence (AI) has become more popular than ever before as the progress further with numerous technological advancements that allows human to improve on the knowledge and analysis of subjects that difficult to be calculated or analyzed using the hand on engineering methods. The breakthrough of AI technology is contributed by the advanced processing power that has become much more powerful, cheaper, and smaller compared to the technology two decades ago. Nowadays, the human had arrived into a new generation where artificial intelligence is being integrated with tons of different fields out there [1]. AI technology today can also be contributed to the idea or concept of machine learning (ML). The development of AI technology has advanced and getting mature since the algorithms are contributed by worldwide researchers and also released to the public as open-source which able to utilize an application to ensure that AI technology becomes more widespread and easily applied in application [2].

Google has recently released new hardware that can compute certain AI at a very fast speed [3]. The hardware is called Edge TPU and can be connected to most devices that can run AI by just using a USB connection. These devices are not a replacement of any current technology but the hardware that can help speed up AI processing power significantly on edge devices. Currently, deep learning is considered one of the most popular ways of adopting AI into any kind of application, and its algorithm was available and ready to use in most popular programming libraries [4]. The widespread adoption of deep learning also produces many different kinds of deep neural network (DNN) architecture for different applications. Many deep learning architectures are widely used in object classification and detection which produce different inference time and accuracy [5, 6]. However, Google's Edge TPU is mostly untested with all of these deep neural network architecture, the benchmarking results are not fully available online for the more widespread adoption of the hardware into others' setup of AI application. The paper will provide information on Google's Edge TPU hardware performance when used with edge devices to help them with their decision on whether the hardware could prove useful for their application.

II. RELATED WORK

The AI technology itself has been evolving exponentially in the last few decades from a simple rule-based algorithm known as Expert System to the machine learning algorithm commonly seen today due to the advent of cheaper and more powerful computer technology [7]. A tensor processing unit or TPU is an AI accelerator application-specific integrated circuit (ASIC) developed by Google specifically for neural network machine learning in the Tensorflow ML library. The technology was announced to the public in 2016 after a year of usage inside their data centres [8]. In between the first announcement of the TPU until the announcement of the Edge TPU, the TPU has gone through three different generations with each iteration providing much more powerful and robust performance for its NN computation. In 2018, Google makes another announcement on the TPU by introducing the Edge TPU. The Edge TPU is Google's purpose-built ASIC chip designed to run ML models for edge computing, meaning that it is much smaller and consumes far less power compared to the TPUs hosted in Google datacentres [9]. The announcement came with a myriad of hardware offered by Google for ease of development with the Edge TPU. The Edge TPU will come in a single-board computer (SBC), a system on module (SoM), a USB accessory, a mini PCI-e card, and an M.2 card as seen in Figure 1.

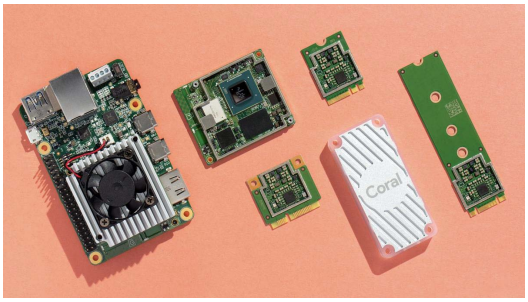


Figure 1 Coral Hardware with Edge TPU built-in [9].

978-1-7281-7310-8/20/$31.00 ©2020 IEEE

The advent of edge computing is caused by the necessity to increase the processing speed of devices running real-time applications on-site as the bandwidth of the internet is taken over by the advent of the internet of things (IoT). Simply means there are more devices connected to the internet today causing congestion in certain areas, thus decreasing the speed of data transfer for everyone as the network needs to cope with millions of data passing through it. Edge computing simply provides information processing close to the edge (where things and people produce and/or consume that information) making faster computation speed for real-time application and data transfer to the people who needs it [10]. Figure 2 shows how data gathered through IoT devices are passed through the edge computing devices to act like a gateway where data are interpreted quickly before being pass through the internet to be sent to a server or cloud storage for further processing if necessary or just for data-keeping. It allows for much better communication between the IoT devices and the cloud server as the bandwidth of the server are not crowded by all these devices connecting to it all at once, rather the edge devices act as the transporter of data from the IoT devices to the cloud storage or server [11].

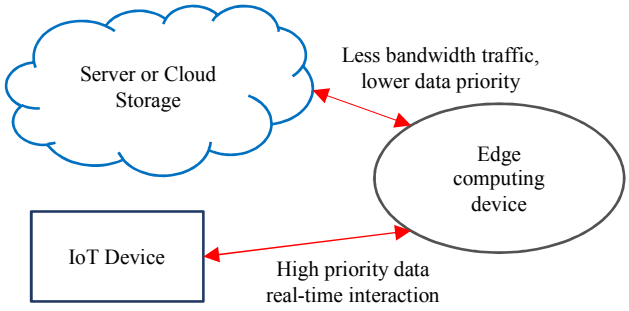


Figure 2 Coral Hardware with Edge TPU built-in.

Ultimately, cloud computing provides a decentralized computing infrastructure for many in the industry thus reducing cost for centralized cloud computing. While the technology of edge computing is getting better as time passes, its computational power still cannot be compared to actual processing capabilities of servers hosted somewhere in the cloud. Cloud computing provides the capability to pre-process data as fast as possible before being sent to the cloud for further computation. The technology, however, provides the perfect environment for AI application that needs to run real-time on any location. A study by M. Alnemari et al. shows how they can create a method to efficiently deploy DNN for edge computing in their research. Their studies show how they manage to compress a few existing architectures of DNN up to 98.71% to make the DNN model able to run on the edge devices while still maintaining high accuracy. Their research includes DNNs such as VGG-16 on CIFAR-10, CIFAR-100, and ImageNet among others [12]. As stated before, one of the challenges faced in deploying IoT or DNN on edge devices is meeting their latency requirements. As some of the application using these technology needs very little to zero latency. A study by J. Chen et al provides a method of fast inference through a couple of DL with edge computing combinations. These DNN being deployed of course still needs to be prune for use with edge devices, however, depending on the application, these architecture laid out by these researches could potentially help save costs and provide a much more robust application of DNN [13].

III. METHODOLOGY

A. *Installation of Edge TPU runtime*

In this work, the runtime package is required to communicate with the Edge TPU. However, the required package only compatible with the Debian-based operating system and it limits the operating system use in the hardware. Hence, the hardware should available to the runtime package to test the Edge TPU device when connecting the edge devices through a USB connection. Since the package is only compatible through a Debian-based operating system, the installation steps are simple with few lines as listed in the documentation.

B. *Deep learning model*

Different types of deep learning models had been chosen as a benchmark with the selected edge devices based on the state-of-art in artificial intelligence. The only selected deep learning models in this testing are MobileNet V1, MobileNet V2, EfficientNet-EdgeTPU (S), Inception V1, Inception V4. All models listed above contain an image with an input size of 224×224 except the Inception V4 which only contains the input size of 299×299. These models are selected because they are available online for the public and also had been trained with the dataset of ILSVRC2012 to recognize 1,000 types of objects [14].

C. *Hardware*

For the testing, four edge devices are selected because they are popular with the public for its compactness, availability, and ease of use, especially for prototyping projects. As shown in Figure 3, the selected edge devices are Raspberry Pi 3 B+, Raspberry Pi 4 B with 1GB ram, Raspberry Pi 4 B with 4 GB ram, and Intel NUC with the I7 processor. All listed hardware contains the USB 3.0 port connection except Raspberry Pi 3 B+, meaning that the data transfer from the hardware to the Edge TPU using the port will faster. Other accessories also included camera, USB Accelerator (has built-in Edge TPU), monitor, variable light, and Coral Development Board. The USB Accelerator is a product of Google released under the Coral name as hardware where the Edge TPU can be used freely. As shown in Figure 4, other accessories also include a camera, USB accelerator, and Coral development board. For the entirety

Pi 4 B (4GB)	Pi 4 B (1GB)	Pi 3 B+	Intel NUC

Figure 3 Edge devices tested in this work.

a) Camera b) USB Accelerator

c) Coral Development Board

Figure 4 Other accessories used in this testing

of the work, the Edge TPU will be connected to the listed edge devices through the USB accelerator in the benchmarking application. The camera is only used during the real-time benchmarking for each deep learning model. The camera will be pointed towards an object for the deep learning models to detect accurately. A variable light source is also used to ensure that the deep learning models able to work successfully under certain light conditions. Another hardware used in this work is the Coral Development Board, a single board computer that contains an Edge TPU coprocessor. The work will include the data of the benchmarking that has been done to the hardware.

D. Benchmarking Application

This section will discuss the benchmarking application program used in this work. This part is separated into two parts of benchmarking applications which are benchmarking application program with test images and camera respectively. The two programs are designed to automate the benchmarking process to reduce human error and laboring time. Only the selected dataset and learning model are decided by the user as input to the program. Figure 5 shows the program flowchart for the benchmarking application with test images (Test 1) or frame from the camera (Test 2). The program is started by initializing the prerequisite libraries, followed by an inquiry to load the deep learning model and testing dataset. A loop will then run starting from loading the test image, starting the timer followed by running the inference and recording the inference time after the end of inferencing. Then, the inference result will then be a check against the test image category and record the result in terms of inference time and accuracy. The result of different deep learning models is then recorded and analyzed. Different from the image testing, the frame of the connected camera is replaced as a test image in the deep learning model. A total of 100 iterations are calculated for testing the image frame from the camera. Figure 6 shows the output example of this benchmarking application.

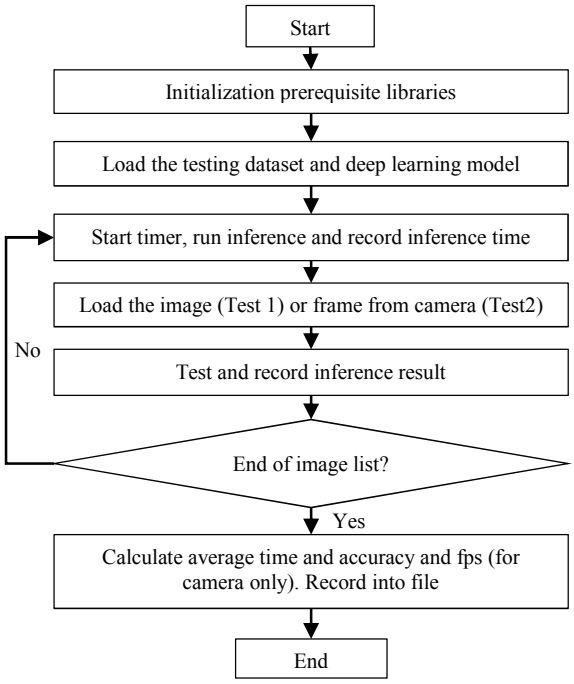


Figure 5 Program flowchart of benchmarking application with test images or camera frame.

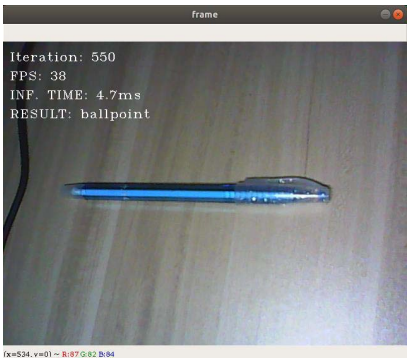


Figure 6 GUI of the benchmarking application with camera.

IV. RESULT AND DISCUSSION

This section details the results of the benchmarking process done in this work. It will also detail some observation and analysis of results based on the findings. A mathematical formula is also used and shown in this section to provide more explanation for the findings. It is important to take note that the benchmarking results with test images are done by passing through 99 different images with 10 different categories to the deep learning model for the benchmarking test with the camera, only one object is tested for classification by the deep learning model. The benchmarking results of the Coral Development Board are only for benchmarking with test images using Edge TPU API. The results of Test 1 and Test 2 will be shown in Figure 7 and Figure 8 respectively. Glancing through the results of the benchmark, the average inference time in the deep learning models are the fastest with the Intel NUC followed by Raspberry Pi 4 B with 4GB ram and Raspberry Pi 4 B with 1 GB ram (depending on the library used for the benchmarking) and lastly is the Raspberry Pi 3 B+. This result is expected as the Intel NUC host the most powerful

and fastest CPU out of all of them followed by the Raspberry Pi 4 B and Raspberry Pi 3 B+. Next, analyzing the results based on the DNN models, it seems that MobileNet V1 and V2 have the fastest average inference time followed by Inception V1, EfficientNet S, and lastly Inception V4. The accuracy of all models is the same across all platforms as they are tested against the same test images. The accuracy results, however, show that the Inception V1 and V4 as the most accurate DNN models when tested against our test images followed by MobileNet V2, MobileNet V1, and EfficientNet S which are quite closed to each other. Furthermore, this work managed to benchmark the DNN models using Edge TPU API provided by the Coral documentation.

From the results, it seems that the Edge TPU API provides better performance when using the CPU entirely as the processing power of the DNN models. This is quite an interesting discovery as the Edge TPU API is purposely built for the Edge TPU hardware for ease of use. For the benchmarking with the camera, it is seen that the accuracy for each model on different hardware is different when compared to the benchmarking results with test images. The results show that the Raspberry Pi 3 B+ produces the most accurate performance in real-time followed by Raspberry Pi 4 B and Intel NUC. Overall, the difference in performance

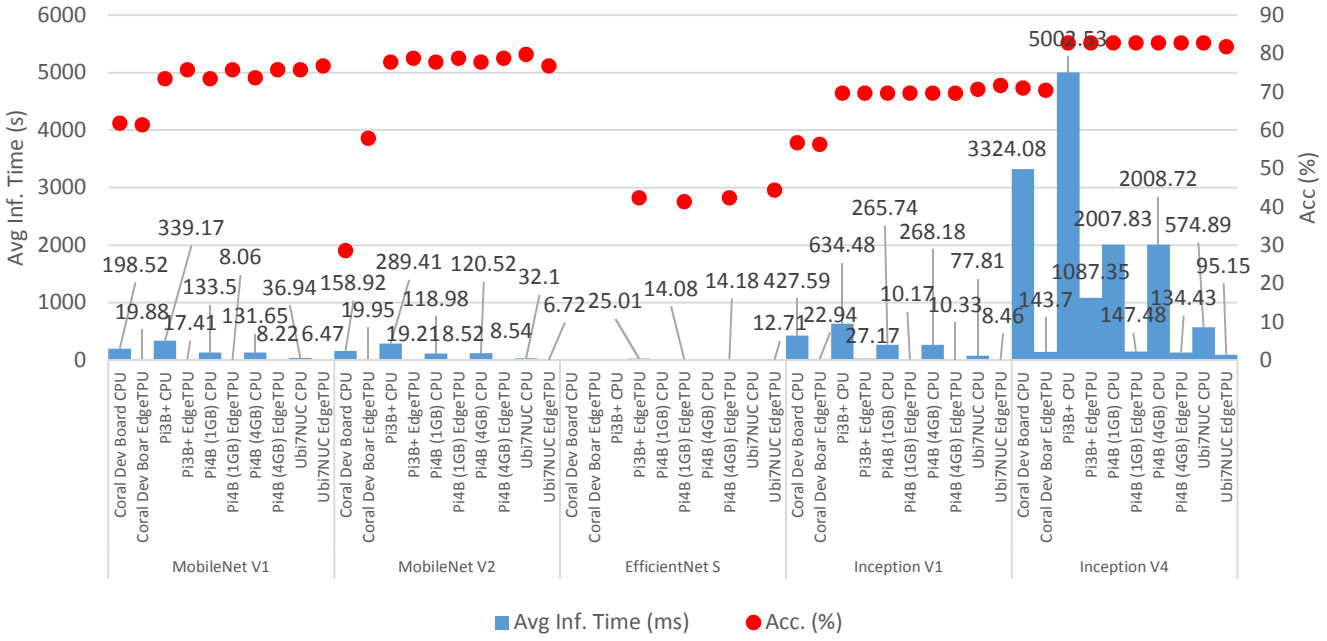


Figure 7 Benchmarking results with test images using Edge TPU

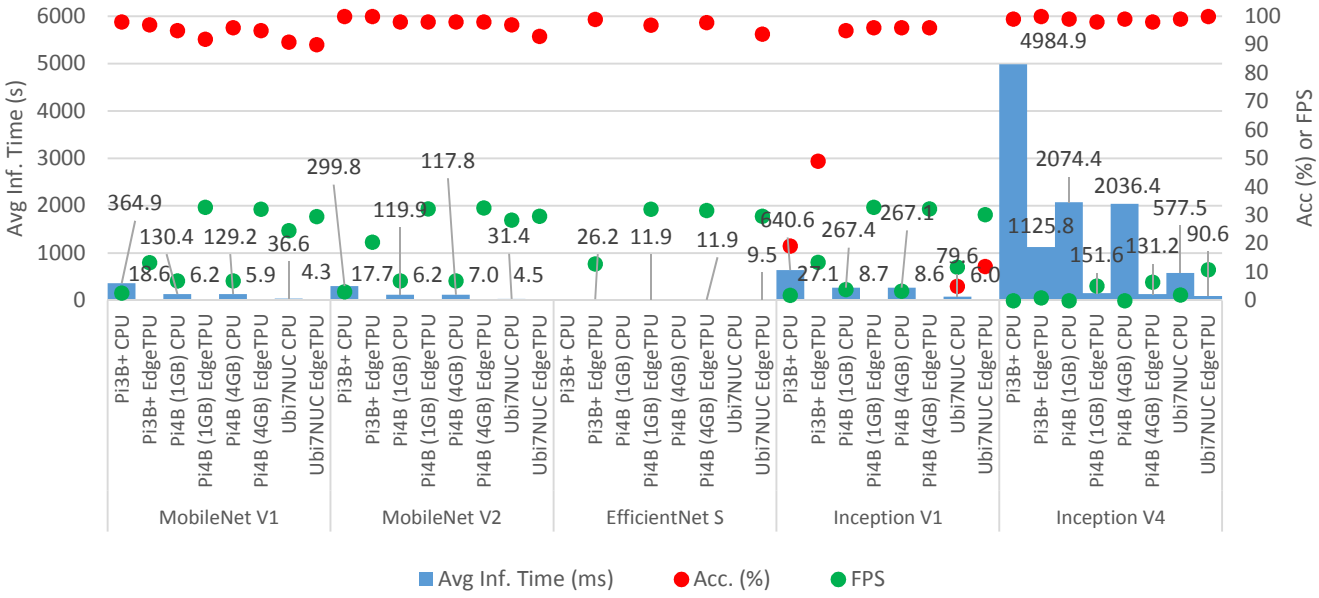


Figure 8 Benchmarking results with camera using Edge TPU

between Raspberry Pi 4 B with 1 GB ram and Raspberry Pi 4 B with 4 GB of ram is quite minuscule. It means that the overall performance of DNN models is not quite dependent on the RAM of the hardware. Also, take note that the average inference time of Intel NUC and Raspberry Pi 4 B are close to each other, the average inference time of the Raspberry Pi 4 B+ technically almost twice for Raspberry Pi 4 B. It is because the Edge TPU connection between this two hardware is different. Raspberry Pi 4 B contains a USB 3.0 module that can be used to connect to the Edge TPU while the Raspberry Pi 3 B+ does not have a USB 3.0 connection.

As the amount of data is huge, this work will only focus on the Intel NUC results for reference. This is because the Intel NUC produces the best results out of all hardware that had been tested. It can be confirmed by applying a simple mathematical equation for calculating the best results. The equation used is as follow:

$$Score_{img} = \frac{Accuracy}{Average\ Inference\ Time} \quad (1)$$

The equation is generated by dividing accuracy with average inference time, it is because the score will be higher if the average inference time is smaller or when the accuracy is higher. For the benchmarking with the camera, the following formula can also be used:

$$Score_{cam} = \frac{\mathrm{FPS} \times Accuracy}{Average\ Inference\ Time} \quad (2)$$

Applying both equations to the data gathered in the results will produce the following Table 1.

TABLE 1 Score for benchmarking with a test image and camera using the Edge TPU API.

Hardware	I7NUC	Pi 4B (4GB)	Pi 4B (1GB)	Pi 3B+
EdgeTPU Image	11.87	9.23	9.40	4.35
EdgeTPU Camera	625.56	515.85	518.05	115.91

To get a better view of the results for the Intel NUC, a scatter plot is used to properly visualize all of the data points on the table. The scatter plot for test images and camera frames can be seen in Figure 9 and Figure 10 respectively. Figure 9 shows a scatter plot for the benchmarking results with test images using Edge TPU API. This library as observes before produced better results in CPU performance. This scatters plot shows a divide between the average inference times when using the Edge TPU hardware and when using the CPU. One data missing from this scatter plot is the performance of EfficientNet S using the CPU, the result of running this DNN model on the CPU causes the program unable to compute due to runtime error. The reason for this error to occur could possibly be caused by the Edge TPU API which is created unable to handle floating-point operation. This is because the Edge TPU hardware is created solely to compute the quantization problem rather than the typical floating-point seen in many DNN models.

Figure 10 shows the scatter plot of benchmarking results with the camera, Inception V1 lags far behind the rest in terms of CPU performance while the rest of the DNN models shows about the same FPS but different inference time. EfficientNet S with Edge TPU lags behind the rest while MobileNet V2 with CPU has the fastest inference time compared to the rest. The light intensity shone on the object affects the camera performance and reduces the FPS of the benchmarking thus reducing the performance of the results. It is because darker lighting causes the camera to take more time to load one frame. These results can be seen in Figure 11. Based on these findings, it can be said that camera quality plays a huge part in making a better and faster performance application running machine learning models.

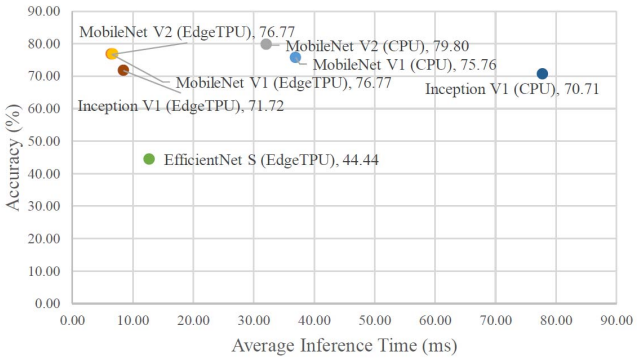


Figure 9 Scatter plot of benchmarking results with test images using Edge TPU API.

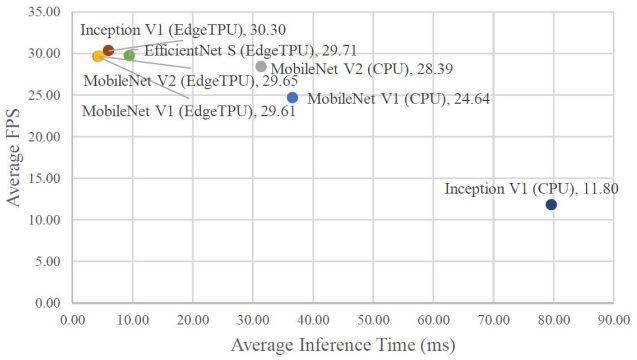
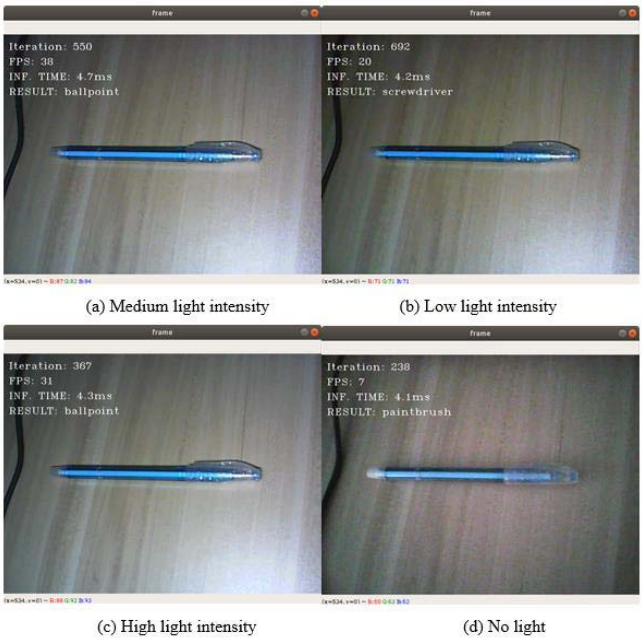


Figure 10 Scatter plot of benchmarking results with camera using Edge TPU API.



(a) Medium light intensity (b) Low light intensity

(c) High light intensity (d) No light

Figure 11 Benchmarking results on different light intensity.

V. CONCLUSION

More powerful edge devices are needed to enhance the computation on-site rather than wait for data or result of DNN transferring from a centralized server hosting. By utilizing the Edge TPU hardware developed by Google, a relatively low-cost machine learning computational chips compared to the big hardware like GPU and cloud TPU that can run a quantize DNN at a very fast speed depending on the hardware and DNN models. Additionally, edge devices use in this work show outstanding performance especially the newly released Raspberry Pi 4 B which can be bought in cheap and the Intel NUC I7 which is very compact size and offers the performance as similar as a personal computer. Both of these devices are suitable options for edge computing with DNN depending on the application needed. It is also important to note that the quality of equipment used alongside any DNN application will affect the performance of the DNN application. One example seen in this work is the camera had been used for benchmarking producing different results for different light intensity shone on the object with medium-light intensity becoming the perfect lighting condition for the benchmarking application.

ACKNOWLEDGMENT

This work was funded by Yayasan Canselor Uniten (201901001YCU/17). The authors would like to acknowledge our research collaborators, Itxotic Sdn. Bhd., for their contribution, expertise and equipment.

REFERENCES

[1] P. M. Oliveira, P. Novais, and L. P. Reis, "Progress in Artificial Intelligence," in 19th EPIA Conference on Artificial Intelligence, Vila Real, Portugal, 2019.

[2] A. Fogg, "A History of Machine Learning and Deep Learning," import.io, 30 May 2018. [Online]. Available: https://www.import.io/post/history-of-deep-learning/. [Accessed 20 December 2019].

[3] E. Zeman, "Google Debuts Coral to Speed Up Local AI Development," ProgrammableWeb, 8 March 2019. [Online]. Available: https://www.programmableweb.com/news/google-debuts-coral-to-speed-local-ai-development/2019/03/08. [Accessed 1 January 2020].

[4] F. Q. Lauzon, "An Introduction to Deep Learning," in 11th International Conference on Information Science, Signal Processing and their Applications, Montreal, QC, Canada, 2012.

[5] pkulzc, "models/detection_model_zoo.md at master · tensorflow/models · GitHub," [Online]. Available: https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md. [Accessed 2019 June 25].

[6] L. Zheng, "Benchmark," 25 October 2018. [Online]. Available: https://github.com/apache/incubator-tvm/wiki/Benchmark. [Accessed 25 September 2019].

[7] D. W. Patterson, Introduction to Artificial Intelligence and Expert System, New Delhi: Prentice Hall of India Private Limited, 1990.

[8] G. M. Ung, "Google's Tensor Processing Unit could advance Moore's Law 7 years into the future," PCWorld, 18 May 2016. [Online]. Available: https://www.pcworld.com/article/3072256/googles-tensor-processing-unit-said-to-advance-moores-law-seven-years-into-the-future.html. [Accessed 2 January 2020].

[9] M. Lynley, "Google is making a fast specialized TPU chip for edge devices and a suite of services to support it," Techcrunch, 26 July 2018. [Online]. Available: https://techcrunch.com/2018/07/25/google-is-making-a-fast-specialized-tpu-chip-for-edge-devices-and-a-suite-of-services-to-support-it/. [Accessed 2 January 2020].

[10] K. Shaw, "What is edge computing and why it matters," Networkworld, 13 November 2019. [Online]. Available: https://www.networkworld.com/article/3224893/what-is-edge-computing-and-how-it-s-changing-the-network.html. [Accessed 2 January 2019].

[11] C. M. Fernandex, M. D. Rodriguez, and B. R. Munoz, "An Edge Computing Architecture in the Internet of Things," in 2018 IEEE 21st International Symposium on Real-Time Distributed Computing (ISORC), Singapore, 2018.

[12] M. Alnemari, and N. Bagherzadeh, "Efficient Deep Neural Networks for Edge Computing," in 2019 IEEE International Conference on Edge Computing (EDGE), Milan, 2019.

[13] J. Chen, and X. Ran, "Deep Learning With Edge Computing: A Review," Proceedings of the IEEE, vol. 107, no. 8, pp. 1655-1674, 2019.

[14] Google LLC., "Models," Google LLC., [Online]. Available: https://coral.ai/models/. [Accessed 20 August 2019.