Journal of Robotics and Control (JRC)
Volume 2, Issue 6, November 2021
ISSN: 2715-5072 DOI: 10.18196/jrc.26123

Real-Time Human Detection Using Deep Learning on Embedded Platforms: A Review

Wahyu Rahmaniar[1]* and Ari Hernawan[2]
[1] Department of Electrical Engineering, National Central University, Zhongli, Taiwan
[2] Department of Computer Science and Information Engineering, National Central University, Zhongli, Taiwan
Email: wahyu.rahmaniar@gmail.com
*Corresponding author

***Abstract*— The detection of an object such as a human is very important for image understanding in the field of computer vision. Human detection in images can provide essential information for a wide variety of applications in intelligent systems. In this paper, human detection is carried out using deep learning that has developed rapidly and achieved extraordinary success in various object detection implementations. Recently, several embedded systems have emerged as powerful computing boards to provide high processing capabilities using the graphics processing unit (GPU). This paper aims to provide a comprehensive survey of the latest achievements in this field by using deep learning techniques in the embedded platforms. NVIDIA Jetson was chosen as a low power system designed to accelerate deep learning applications. This review highlights the performance of human detection models such as PedNet, multiped, SSD MobileNet V1, SSD MobileNet V2, and SSD inception V2 on edge computing. This survey provides an overview of these methods and compares their performance in accuracy and computation time for real-time applications. The experimental results show that the SSD MobileNet V2 model provides the highest accuracy with the fastest computation time compared to other models in our video datasets with several scenarios.**

Keywords*— *computer vision, convolutional neural network (CNN), deep learning, human detection, NVIDIA Jetson, object detection.

I. INTRODUCTION

In recent years there have been major developments in computer vision applications in object detection. This development is due to the increasing need for intelligence systems in robotics [1]-[2], surveillance [3], medical imaging [4], industry [5], and vehicle technology [6]. Human or pedestrian is one of the most essential objects to detect in the images for intelligent systems. Reliable human detection can be used for wide applications in video analysis such as people tracking [7], people counting [8], human-computer interaction [9], crowd detection [10], and human activity recognition [11]. However, human detection faces more challenges compared to traditional object detection because it has more complex environments, occluded objects, and variations in geometry and illumination.

Several techniques that have been presented for human detection. Those techniques are available from traditional object detection to the most refined implementation of pedestrian detection. Early methods in pedestrian detection are mainly focused on feature representation, such as SIFT [12], SURF [13]-[14], shape contexts [15], and the integral channel features (ICF) detector [16], which requires registration of the object to be searched for features in the images. These methods provide unlabeled data that the algorithm tries to understand by extracting its features and patterns so that they can only be used for certain, known objects. Following extensive research into human detection, researchers came up with machine learning techniques. These methods allow the algorithm to learn on labeled datasets and provide analytical results to evaluate their accuracy in the training data. In the machine learning approach, the problem is divided into two steps: object detection (features extraction) and object recognition (classification), as shown in Fig. 1. The features extraction such as histogram of gradient (HOG) [17]-[18], Haar-like features [19]-[20], and local binary pattern (LBP) [21] are often used and suitable for representing objects in the images. Then, these features are trained to classify the detected objects. The Adaboost cascade classifier [22] and support vector machine (SVM) [23] are two widely used classifiers due to their large generalizability and less classification complexity. However, in traditional machine learning techniques, most of the features implemented need to be identified by domain experts to reduce data complexity and make patterns more visible for learning algorithms to function. This limitation makes machine learning less reliable in detecting objects in real applications that have many unexpected conditions.

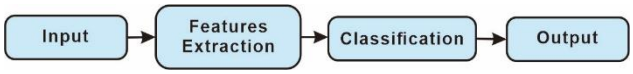


Fig. 1. Machine learning approach.

In recent years, deep learning has become widely known for its breakthrough in computer vision. Deep learning can solve all tasks in one algorithm, as shown in Fig. 2, whereas machine learning needs to divide the algorithm into several parts for each task to be combined at the final stage. Deep learning can learn high-level features from the data incrementally, which can eliminate the need for domain expertise and core feature extraction. This is expected to overcoming the problem of detecting the object in the video, such as the various size of objects, changes in illumination, and real-time computation. The deep neural network (DNN) gain a significant breakthrough by introducing regions with convolutional neural network (CNN) features [24], applying

high-capacity CNN to the bottom-up region, and supervised pre-training for an additional task. CNN has the characteristic to understand rich and complex features without the need to design features manually [25]. Although CNN outperforms commonly known feature descriptors, *i.e*., SIFT, SURF, HOG, LBP, and Haar-like, it took major tradeoffs during its training time. Thus, the improved model of CNN has been suggested, including Faster R-CNN and YOLO [26]–[28]. Faster R-CNN is achieved by optimizing the classification along with the bounding box, adding additional subnetworks to generate regions, and fixed grid regression. This model eliminates selective search algorithms that allow the network to learn the proposed regions. Unlike region-based algorithms, YOLO detects objects by predicting their location and class probability using a single convolutional network. However, YOLO has limitations in detecting small objects in the image due to the spatial constraints of the algorithm. Besides, Faster R-CNN and YOLO provide high-accuracy object detection with real-time performance on a PC, whereas embedded platforms have limited capabilities to run these models due to the large memory resource requirements.

Fig. 2. Deep learning approach.

A study in [9] mentioned that the ideal object detection algorithm is one that can meet high accuracy and efficiency. Although the problem of object detection seems straightforward, the aspects of accuracy and computation time need to be considered, which makes it a challenge for applications in real environments. First, the variability of objects belonging to the same class is one of the biggest difficulties, such as changes in perspective, partial occlusion, unexpected noise, and changes in illumination. These factors and several other things that may occur in the field can cause the algorithm to lose information. Second, the issues of time efficiency, memory management, and storage are required to train this detector. Authors in [29] discuss the performance of deep learning techniques for object detection on embedded platforms. To achieve high detection accuracy, an object detection algorithm must be able to cope with intra-class variations, environmental conditions, and image disturbances. Furthermore, to achieve high efficiency on an embedded platform, an algorithm must be able to be used on low-end mobile devices that have limited memory, limited speed, and low computing capabilities.

In this paper, we evaluate and compare human detection models on cost-effective embedded platforms such as the NVIDIA Jetson TX2 and Nano. The deep learning models use general-purpose datasets, *i.e*., PASCAL VOC, COCO, and ILSVRC, for training and evaluation. The research investigates the ability of the NVIDIA Jetson to run models such as PedNet, multiped, SSD MobileNet V1, SSD MobileNet V2, and SSD inception V2. As processing speed is a key factor in embedded systems, this study also conducted comprehensive comparisons among those deep learning techniques to find the most efficient model. The main contributions of this study can be summarized as follows: integration and optimization of people detection algorithms in real applications into embedded platforms, end-to-end comparisons between existing people detection models in terms of accuracy and performance, and datasets that can be used to detect and track people in the building.

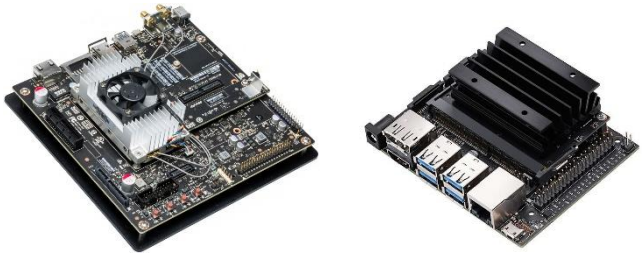
The remainder of the paper is organized as follows. Section II explains the embedded system platform and human detection models. Section III describes and discusses the experimental results. Then, Section IV draws the conclusions.

II. Human Detection Models

A. Embedded Platform Benchmark

To realize object detection using artificial intelligence (AI) in real applications on the embedded platform, several conditions are needed, *e.g*., high accuracy, fast computation time, small model size, and efficient energy consumption. Furthermore, the development of a computer vision algorithm is not only based on the techniques, but also on advanced parallel computing architectures that enable algorithms to run efficiently [30]. As a result, the hardware industry has begun to focus on embedded platforms which can provide high-precision performance at low latency.

In this paper, we compare several models for human detetcion using two integrated hardware accelerators emerging from NVIDIA: Jetson TX2 (Fig. 3(a)) and Nano (Fig. 3(b)). Jetson TX2 is an embedded platform that can run computer vision applications with fast and small power consumption (7.5W - 15W). Thus, it provides a solution for implementing software for object detection using AI in real-time. Jetson Nano has a smaller size with slower performance but uses less power (5W – 10W) and lower cost than TX2. These small and powerful devices make it possible to execute computer vision algorithms efficiently in parallel.



(a) (b)

Fig. 3. NVIDIA Jetson board. (a) TX2. (b) Nano.

The graphics processing unit (GPU) allows embedded hardware to optimally execute specialized tasks in AI. GPU is an accelerator with a focus on graphics processing which began to grow as the entertainment industry advances, including the audio and video processing and gaming

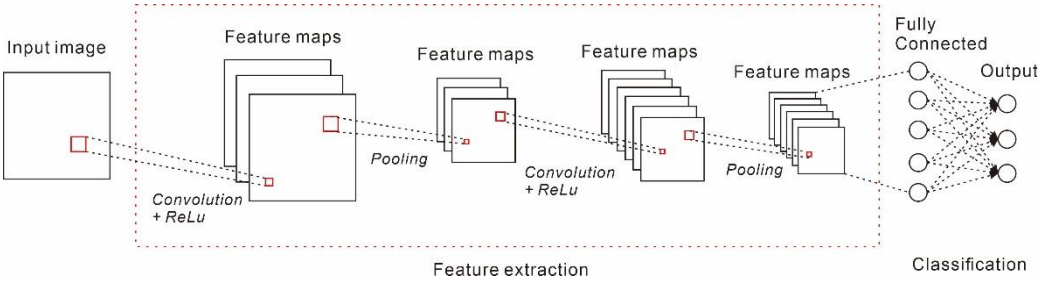


Fig. 4. Architecture of CNN.

industries. GPUs excel at matrix and vector operations used on neural networks. The GPU in NVIDIA Jetson consists of multiple processing cores, *e.g.*, 128-core Maxwell 921MHz on Nano and 25-core Pascal 1.3GHz on TX2. Besides, modern GPUs rely not only on powerful graphics engines but also on parallel processors with high memory bandwidth for computationally demanding algorithms [31]. The GPU uses Compute Unified Device Architecture (CUDA) [32] for its parallel computing platform. NVIDIA developed the CUDA Deep Neural Network Library (cuDNN) [33] as a GPU-accelerated library that can be used to improve system performance, particularly for deep learning implementations such as forward and backward convolution, pooling, normalization, and activation layers.

B. Convolutional Neural Network

CNN was introduced as a self-organizing neural network model in [34] and developed as gradient-based learning in [35]. The learning model in CNN can be used to train and test computer vision tasks. The CNN neural network consists of the convolutional layers, the non-linearity layers, and the pooling layers. A convolution layer with filters scans the image and creates a feature map predicting which class each feature belongs to. Rectified Linear Unit (ReLU) is used in a non-linearity layer that creates a robust neuron response to data corruption. This will not result in a large negative value in the feature map output despite a lot of corruption in the image. The pooling layer uses a max-pooling to reduce the resolution of the feature map and preserve the most important information. Then, the fully connected layer and softmax functions are applied to classify objects with probabilistic values between 0 and 1. Fig. 4 shows a typical CNN architecture.

In the fully connected layer, the output matrix in the previous layer is flattened into a vector for the input at the next stage. Inputs on feature analysis are combined to get the weights to predict the correct label and create a model. Furthermore, an activation function such as softmax is used to classify the output as a person.

Convolutional operations can find the correct direction for space reduction, whereas the pooling and non-linearity operations can deduce space in that direction. CNN is particularly suited for accurate modeling of objects because images consist of small details or features. Besides, CNN could create a mechanism to analyze each feature separately that informs conclusions about an image.

CNN has been the main architecture for deep learning platform since the popularization of deep convolutional neural networks on ImageNet [36]. To achieve higher detection accuracy, a deeper and more complex network is required. However, modeling like this requires more memory capacity and computation time in real-world applications, and it is necessary to detect and recognize objects promptly on time on limited computing platforms.

On the Jetson board, TensorRT can be used for high-performance inference on NVIDIA GPUs. The TensorRT is specially designed to quickly and efficiently run a trained CNN network of GPUs. TensorRT optimizes the CNN network by combining layers and optimizing the selected kernel to increase latency, throughput, power efficiency, and memory consumption [37]. In deep learning applications, TensorRT will optimize the network to run with lower precision, which will further improve performance and reduce memory requirements.

In this paper, CNN-based models such as PedNet, multiped, SSD MobileNet V1, SSD MobileNet V2, and SSD inception V2 are used for real-time human detection on embedded platforms.

C. PedNet and Multiped

The PedNet model is specifically designed for pedestrian detection and the multiped model is designed for pedestrian and luggage detection [38]. PedNet using the convolution filter of size 3×3 and a max-pooling window size of 2×2 throughout the network. ReLU is used as an activation function with batch normalization after every convolution layer. The backbone of the PedNet consists of an encoder–decoder network for down sampling and up sampling the feature maps, respectively. The input to the network is a set of three frames and the output is a binary mask of the segmented regions in the middle frame. Irrespective of classical deep models where the convolution layers are followed by a fully connected layer for classification, PedNet is a Fully Convolutional Network (FCN) as shown in Fig. 5.

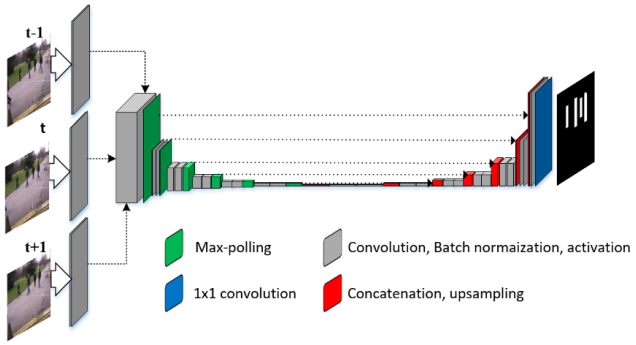


Fig. 5. PedNet architecture [37].

D. SSD MobileNet

The MobileNet model replaces the standard convolution with depthwise separable convolution, which reduces the complexity and model size [39]. Fig. 6 shows the MobileNet V1 convolutional blocks. The depthwise separable convolution divides the kernel into two: input filtering using a depthwise convolution layer and input combining using a 1×1 convolution or pointwise layer. The 3×3 convolution is followed by a batch norm and ReLU6 non-linearity that use low-precision computing. Then, the pointwise layer is followed by a batch norm and ReLU6 after each convolutional layer. ReLU6 is more robust than regular ReLU and prevents the activations from getting too big.

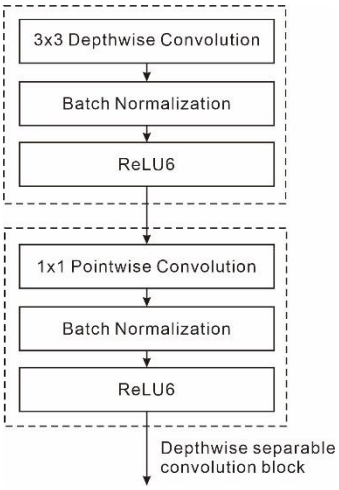


Fig. 6. MobileNet V1 convolutional blocks.

The MobileNet V1 architecture is consists of a regular 3×3 convolution at the first layer without pooling layer between depthwise separable blocks. The first layer serves to expand the number of channels in the data before entering the depthwise convolution. Therefore, this layer and the corresponding pointwise layer increase the number of output channels. The strides of 2 are used on the depthwise layer to reduce the spatial dimensions of the data. In the end, there is a global average pooling layer and a fully-connected classification layer followed by a softmax.

Fig. 7 shows the MobileNet V2 convolutional blocks with the residual connection that make it different from MobileNet V1. In the first layer, the 1×1 convolution aims to expand the number of channels in the data. The expansion layer aims to make the output have more channels than the input. Then, the lightweight depthwise convolutions are used to filter features as sources of ReLU6 non-linearity. In the third layer, the 1×1 convolution layer has an opposite function of MobileNet V1. This layer projects data with a high number of dimensions into a tensor with a lower number of dimensions also called the bottleneck layer. As an improvisation of MobileNet V1, the MobileNet V2 architecture consists of a regular 1×1 convolution, a global average pooling layer, and a classification layer.

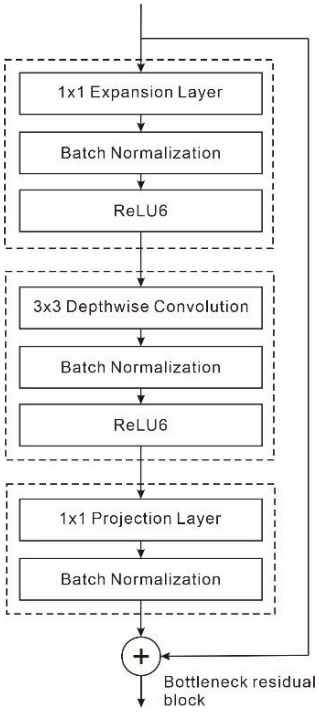


Fig. 7. MobileNet V2 convolutional blocks.

MobileNet is an architecture for classification or feature extractor purposes, and a single shot multi-box detector (SSD) is an architecture that produces the bounding box localizations for detection purposes. The SSD [40] approach is based on a feed-forward convolutional network that uses only one shot to produces a fixed-size bounding box and detect a class of objects in the image. Convolutional feature layers are used to detected multiple scales of various sizes. In addition, multiple feature maps are used to improve the accuracy of each object class prediction.

E. SSD Inception V2

SSD inception [41]-[42] were proposed to improve SSD classification accuracy and reduce computational complexity without affecting detection speed. The number of the input channel is limited by adding a 1×1 convolution layer in the inception module to reduce the computation cost of deep neural networks.

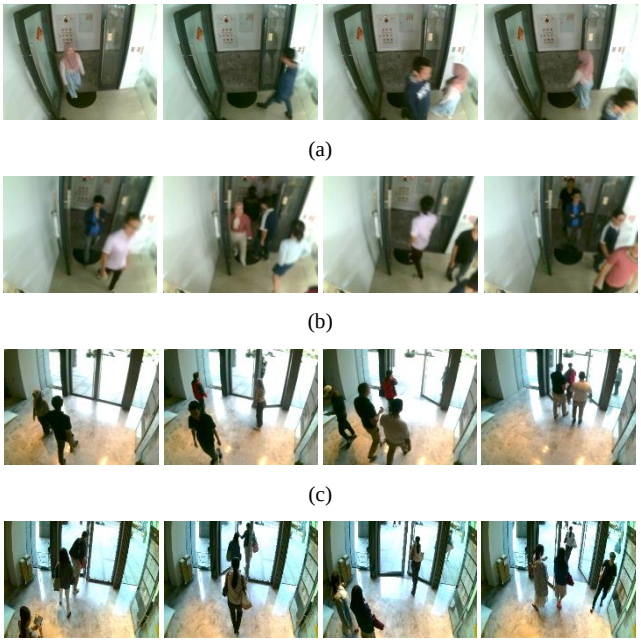
SSD inception V2 [43] reduces representational bottleneck and uses the smart factorization method. The 5×5 convolution is factorized into two 3×3 convolution layers to make the computational times faster. Then, the $n \times n$ filter size convolution is factorized into $1 \times n$ and $n \times 1$ convolution combination to reduce computational cost. The filter bank in the module is widened to avoid reductions in excessive dimensions that lead to bottlenecks and loss of information.

III. Experimental Results and Discussion

In this study, video shooting was carried out in a building using an RGB camera with a resolution of 1280×720. For real-time purposes, image resolution is resized into a resolution of 320×240. There are four types of datasets used with each scenario, as shown in Fig. 8. Each video can be described as follows:

- Video-1: This dataset consists of people entering and leaving the door simultaneously in an irregular position and at a close distance to each other. In one frame there are about 1-2 people.
- Video-2: It has the same scenario as video-1, but in one frame there are 1-4 people with different movement speeds, some people walk slowly, and others walk faster.
- Video-3: This dataset has darker lighting in a larger room than video-1 and video-2.
- Video-4: It has the same scenario as video-3, but has more people in one frame with different movement speeds, and some people are covered up.

Video-1 and video-2 were taken about 2m from the camera, and some objects only show the upper half of the body. Video-3 and video-4 were taken with a distance of more than 2m from the camera, and the object often shows the full human body.



(a) (b) (c) (d)

Fig. 8. Datasets. (a) Video-1. (b) Video-2. (c) Video-3. (d) Video-4.

(a)

(b) (c) (d) (e)

Fig. 9. Human detection results on 4 datasets (Upper: Video-1 and video-2. Lower: Video-3 and video-4). (a) Pednet. (b) Multiped. (c) SSD MobileNet V1. (d) SSD MobileNet V2. (e) SSD Inception V2.

TABLE I. PERFORMANCE METRICS COMPARISON

Model	Accuracy Results		
	PR	Recall	F1-measure
PedNet	0.55	1	0.71
Multiped	0.62	1	0.76
SSD MobileNet V1	0.91	0.99	0.94
SSD MobileNet V2	0.94	0.99	0.98
SSD Inception V2	0.89	0.98	0.93

The performance metrics comparison of the five models is summarized in Table I. Precision rate (*PR*), recall, and F1-measure can be computed as follows

$$PR = \frac{TP}{TP + FP} \quad (1)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (2)$$

$$\text{F1-measure} = 2 \times \frac{PR \times \text{Recall}}{PR + \text{Recall}} \quad (3)$$

where *TP* is true positive, *FP* is false positive, and *FN* is false negative. *TP* is the bounding box on the image that detects people correctly. *FP* is the location of the bounding box that detects a background or other object as a person. *FN* is a person in the image but is not detected. In Table I, True Negative (*TN*) is not used as a performance metric, because *TN* describes an empty box as a non-object. In this situation, there may be many empty boxes which will be detected as *TN* which are not necessary to determine the accuracy of the models, where the detected background as a person is categorized as *FP*.

Table I shows that both pednet and multiped have more *FP* than the other models, but have almost no *FN*. Thus, the recall value for both pednet and multiped is 1. Our results show that both pednet and multiped are not good for human detection in our scenario, where pednet has the lowest F1-measure of 0.71. For other models such as SSD MobileNet V1, SSD MobileNet V2, and SSD inception V2, the results have almost the same accuracy. The SSD MobileNet V2 achieves the highest F1-measure of 0.98, where SSD MobileNet V1 and SSD inception V2 are not good enough to detect when people are moving rather fast. In video-3 and video-4 the lighting is darker than video-1 and video-2, the detection result using SSD inception V2 is not better than SSD MobileNet V1 and SSD MobileNet V2. This situation causes SSD inception V2 to has lower *PR* and recall compared to SSD MobileNet V1 and V2.

In [44], pednet and multiped have the best accuracy for their datasets, such as pedestrian with the small object sizes in the images with a resolution of 1280×720. In this case, pednet and multiped models and the training data are very suitable. In our results, pednet has better accuracy for video-3 and video-4 that have more people from afar than video-1 and video-2, so the human body shape can be seen more clearly. In our case, the objects in the images were taken with a camera angle of about 30 to 60 degrees that sometimes causes the shape of the human is not very clear. In our datasets, because of the camera angle, the distance between the object and the camera, and the movement of the object, sometimes objects only appear upper half of the human body that is covered up due to crowd.

In Fig. 9, the bounding box in red shows the results of human detection on 4 datasets. Pednet and multiped cannot detect humans on video-1 and video-2 where the human body shape was unclear and slightly blur, as shown in Fig. 9(a) and (b). SSD MobileNet V1 cannot detect humans in the image that are slightly blurred and far from the camera, as shown in Fig. 9(c). SSD MobileNet V2 can detect almost all humans in the image with several conditions and positions that cannot be detected with other models, as shown in Fig. 9(d). SSD inception V2 cannot detect humans in darker illuminated images, as shown in Fig. 9(e).

TABLE II. COMPUTATION TIME COMPARISON IN FPS

Model	NVIDIA Jetson Board	
	Nano	TX2
PedNet	10.21	15.46
Multiped	11.05	16.83
SSD MobileNet V1	16.35	24.79
SSD MobileNet V2	17.32	26.04
SSD Inception V2	13.37	20.43

Table II summarizes the comparison of the average computation time in each model. In [44], pednet has the fastest computation time compared to other models. This indicates that pednet is more suitable for small objects in the image, such as pedestrians on the road. In our results, SSD MobileNet V2 has the fastest computation time on NVIDIA Jetson Nano and TX2. A comparison between the two boards shows that the Nano has about 0.65 times the computation time performance of the TX2. The performance of the SSD MobileNet V2 on the Nano is fast enough to be used in the real-time application at 17.32 FPS.

In SSD MobileNet V2, pointwise convolutions that make the number of channels smaller and residual bottleneck block reduce the amount of data on the network, making detection times faster than other models. The SSD MobileNet V2 is suitable to be used as a human detection model on our dataset, which has a good result with blur object due to motion, darker lighting, and half covered objects. High accuracy and fast computation time in human detection using SSD MobileNet V2 are very suitable for applications on embedded platforms.

IV. CONCLUSIONS

This paper has presented human detection testing in the building using NVIDIA Jetson TX2 and Nano. The experimental results show that the Jetson boards have a good performance for implementing computer vision using deep learning. In addition, the Jetson boards also provide an acceleration library that can be used to improve the processing performance of object modeling in deep neural networks. Thus, its development allows the implementation of complex models to create a variety of computer vision applications. SSD MobileNet V2 is a model that can detect humans on our datasets more accurately and faster than other models in the jetson inference, such as PedNet, multiped, SSD MobileNet V1, and SSD inception V2. Fast computation time on implementation using SSD MobileNet V2 makes it possible to design online applications with real-time performance on the embedded systems. The results of this analysis are expected to be used as a reference in selecting a model for human detection applications.

REFERENCES

[1] W. Rahmaniar, W. Wang, and H. Chen, "Real-time detection and recognition of multiple moving objects for aerial surveillance," Electronics, vol. 8, no. 12, pp. 1373–1390, 2019.

[2] W. Rahmaniar and A. E. Rakhmania, "Online digital image stabilization for an unmanned aerial vehicle (UAV)," Journal of

Robotics and Control, vol. 2, no. 4, pp. 234–239, 2021.
[3] E. P. Ijjina, D. Chand, S. Gupta, and G. K, "Computer vision-based accident detection in traffic surveillance," in Proc. of Int. Conf. Comput. Commun. Netw. Technol., 2019, arXiv:1911.10037.
[4] W. Rahmaniar and W. Wang, "Real-time automated segmentation and classification of calcaneal fractures in CT images," Appl. Sci., vol. 9, no. 15, pp. 3011–3028, 2019.
[5] R. Shanmugamani, M. Sadique, and B. Ramamoorthy, "Detection and classification of surface defects of gun barrels using computer vision and machine learning," Meas. J. Int. Meas. Confed., vol. 60, pp. 222–230, Jan. 2015.
[6] M. V. Rajasekhar and A. K. Jaswal, "Autonomous vehicles: The future of automobiles," in Proc. of IEEE International Transportation Electrification Conf., 2016, pp. 1-6.
[7] J. García, A. Gardel, I. Bravo, J. L. Lázaro, and M. Martínez, "Tracking people motion based on extended condensation algorithm," IEEE Trans. Syst. Man, Cybern. Part ASystems Humans, vol. 43, no. 3, pp. 606–618, 2013.
[8] A. B. Chan and N. Vasconcelos, "Counting people with low-level features and bayesian regression," IEEE Trans. Image Process., vol. 21, no. 4, pp. 2160–2177, Apr. 2012.
[9] E. Zhang, B. Xue, F. Cao, J. Duan, G. Lin, and Y. Lei, "Fusion of 2D CNN and 3D DenseNet for dynamic gesture recognition," Electronics, vol. 8, no. 12, p. 1511, Dec. 2019.
[10] M. Ling and X. Geng, "Indoor crowd counting by mixture of gaussians label distribution learning," IEEE Trans. Image Process., vol. 28, no. 11, pp. 5691–5701, 2019.
[11] H. H. Ali, H. M. Moftah, and A. A. A. Youssif, "Depth-based human activity recognition: A comparative perspective study on feature extraction," Futur. Comput. Informatics J., vol. 3, no. 1, pp. 51–67, 2018.
[12] T. Nguyen, E. A. Park, J. Han, D. C. Park, and S. Y. Min, "Object detection using scale invariant feature transform," Advances in Intelligent Systems and Computing, vol. 238, pp. 65–72, 2014.
[13] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)," Comput. Vis. Image Underst., vol. 110, no. 3, pp. 346–359, Jun. 2008.
[14] W. Rahmaniar and W.-J. Wang, "A novel object detection method based on Fuzzy sets theory and SURF," in Proc. of International Conference on System Science and Engineering, 2015, pp. 570–584.
[15] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 4, pp. 509-522, 2002.
[16] B. E. Demiröz, A. A. Salah, Y. Bastanlar, and L. Akarun, "Affordable person detection in omnidirectional cameras using radial integral channel features," Mach. Vis. Appl., vol. 30, no. 4, pp. 645–655, Mar. 2019.
[17] N. Dalal and W. Triggs, "Histograms of oriented gradients for human detection," in Proc. of IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR), 2005, pp. 886–893.
[18] Y. Pang, Y. Yuan, X. Li, and J. Pan, "Efficient HOG human detection," Signal Processing, vol. 91, no. 4, pp. 773–781, Apr. 2011.
[19] P. Viola and M. J. Jones, "Robust real-time face detection," Int. J. Comput. Vis., vol. 57, no. 2, pp. 137–154, May 2004.
[20] P. Viola, M. J. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance," in Proc. of IEEE International Conference on Computer Vision, 2003, pp. 734–741.
[21] L. Nanni, A. Lumini, and S. Brahnam, "Survey on LBP based texture descriptors for image classification," Expert Syst. Appl., vol. 39, no. 3, pp. 3634–3641, 2012.
[22] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in Proc. of IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognition (CVPR), 2001, pp. I-511-I–518.
[23] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," Data Min. Knowl. Discov., vol. 2, no. 2, pp. 121–167, 1998.
[24] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in Proc. of IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR), 2014, pp. 580–587.
[25] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.
[26] R. Girshick, "Fast R-CNN," in Proc. of IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1440–1448.
[27] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proc. of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779–788.
[28] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, 2017.
[29] C. B. Murthy, M. F. Hashmi, N. D. Bokde, and Z. W. Geem, "Investigations of object detection in images/videos using various deep learning techniques and embedded platforms - a comprehensive review," Appl. Sci., vol. 10, no. , pp. 3280-3326, 2020.
[30] X. Feng, Y. Jiang, X. Yang, M. Du, and X. Li, "Computer vision algorithms and hardware implementations: A survey," Integr. VLSI J., vol. 69, no. June, pp. 309–320, 2019.
[31] C. Zhang, P. Patras, and H. Haddadi, "Deep learning in mobile and wireless networking: a survey," IEEE Commun. Surv. Tutorials, vol. 21, no. 3, pp. 2224–2287, 2019.
[32] J. Nickolls, I. Buck, M. Garland, and K. Skadron, "Scalable parallel programming with CUDA," Queue, vol. 6, no. 2, pp. 40–53, Mar. 2008.
[33] S. Chetlur et al., "cuDNN: efficient primitives for deep learning," Oct. 2014, arXiv:1410.0759.
[34] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," Biol. Cybern., vol. 36, no. 4, pp. 193–202, Apr. 1980.
[35] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in Proc. of IEEE, vol. 86, no. 11, 1998, pp. 2278–2323.
[36] A. Krizhevsky, I. Sutskever, and H. Geoffrey E., "ImageNet classification with deep convolutional neural networks," Adv. Neural Inf. Process. Syst. 25, pp. 1–9, 2012.
[37] "NVIDIA Deep Learning TensorRT Documentation." https://docs.nvidia.com/
[38] M. Ullah and A. Mohammed, "PedNet : A spatio-temporal deep convolutional neural network for pedestrian segmentation," J. Imaging, vol. 4, no. 9, pp. 1–18, 2018.
[39] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv:1704.04861, 2017.
[40] W. Liu et al., "SSD: Single Shot MultiBox Detector," arXiv:1512.02325v5, 2016.
[41] C. Szegedy et al., "Going deeper with convolutions," in Proc of IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR), 2015, pp. 1–9.
[42] C. Ning, H. Zhou, Y. Song, and J. Tang., "Inception single shot multibox detector for object detection, " in Proc. of IEEE International Conference on Multimedia & Expo Workshops (ICMEW), 2017, pp. 549-554.
[43] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in Proc. of IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 2818–2826.
[44] L. Barba-guaman and A. Ortiz, "Deep learning framework for vehicle and pedestrian detection in rural roads on an embedded GPU," Electronics, vol. 9, no, 4, pp. 1–17, 2020.