A NEW APPROACH USING COMPUTER VISION FOR DRONE DETECTION

Pham Van Viet
Le Quy Don Technical University

ABSTRACT

Nowadays, one individual or organization can easily get a drone with an affordable budget. With the ability of carrying explosive materials, cameras and illegal things, drones can become security threats to military and civilian organizations. The detection of drones appearing in unauthorized areas becomes an urgent problem. This paper conducts empirical studies on training the deep convolutional neural network Faster R-CNN so that Faster R-CNN after training can most accurately detect drones in images. The obtained Faster R-CNN after training can then be used in drone detection, warning and defense systems for sensitive areas. Faster R-CNN is trained using a dataset of images with drone labeled bounding boxes and different training options. With proper training options determined through experiments, Faster R-CNN after training can detect drones with the average precision up to 0.774, which is 83% higher than Fast R-CNN with the average precision of 0.420 on the same dataset.

Keywords: *Machine learning; computer vision; convolutional neural network; faster R-CNN; drone detection.*

Received: 07/5/2020; Revised: 23/5/2020; Published: 19/8/2020

MỘT CÁCH TIẾP CẬN MỚI SỬ DỤNG THỊ GIÁC MÁY TÍNH CHO VIỆC PHÁT HIỆN MÁY BAY KHÔNG NGƯỜI LÁI

Phạm Văn Việt
Trường Đại học Kỹ thuật Lê Quý Đôn

TÓM TẮT

Ngày nay, một cá nhân hay tổ chức có thể dễ dàng có được một máy bay không người lái (drone) với mức ngân sách chấp nhận được. Với khả năng mang theo những vật liệu nổ, các camera và các vật phi pháp, các drone có thể trở thành các mối đe dọa về anh ninh đối với các tổ chức quân và dân sự. Phát hiện các drone xuất hiện trong các khu vực không được phép trở thành một bài toán cấp thiết. Bài báo này thực hiện các nghiên cứu thực nghiệm cho việc huấn luyện mạng nơ-ron tích chập nhiều tầng Faster R-CNN để Faster-CNN sau khi huấn luyện có thể phát hiện chính xác nhất các drone trong ảnh. Faster R-CNN sau khi huấn luyện có thể sử dụng trong các hệ thống phát hiện, cảnh báo và phòng thủ drone cho các khu vực nhạy cảm. Mạng Faster R-CNN được huấn luyện sử dụng tập dữ liệu ảnh với các hộp giới hạn gán nhãn drone và các lựa chọn huấn luyện khác nhau. Với các lựa chọn huấn luyện hợp lý được xác định thông qua các thực nghiệm, Faster R-CNN sau khi huấn luyện có thể phát hiện drone với độ chính xác trung bình lên tới 0,774, cao hơn 83% so với Fast R-CNN với độ chính xác trung bình là 0,420 trên cùng một tập dữ liệu.

Từ khóa: *Học máy; thị giác máy tính; mạng nơ-ron tích chập; Faster R-CNN; phát hiện máy bay không người lái.*

Ngày nhận bài: 07/5/2020; Ngày hoàn thiện: 23/5/2020; Ngày đăng: 19/8/2020

Email: v.v.pham2012@gmail.com
https://doi.org/10.34238/tnu-jst.3082

1. Introduction

Nowadays, one individual or organization can easily get a drone with an affordable budget. With the ability of carrying explosive materials, cameras and illegal things, drones can become security threats to military and civilian organizations. The detection of drones appearing in unauthorized areas becomes an urgent problem to alert, prevent and track the operation of these devices.

In order to detect drones, many different types of sensors such as RADAR, LIDAR, acoustic and RF (Radio Frequency) sensors can be used as reviewed in [1]. However, RADAR has limitations in detecting drones that fly at low velocities and are small. LIDAR has problems with large data output and cloud sensitivity. An acoustic sensor has problems with long operational range and noisy environment. An RF sensor cannot work when drones fly without ground control.

Detecting drones using computer vision is a good option with many advantages. The computer based system with modern cameras can detect small drones from distance. The system also can detect drones flying at low speeds and does not depend on whether drones are with or without ground control. Other advantages of the system include the abilities of visualization and interpretation. Therefore, cameras are now widely used to detect drones. Cameras become a part of modern drone detecting systems such as ND-BU001 [2] and DroneSentry [3].

Drone detection using computer vision is to determine if a drone is in an input image and where the drone is in the image. The location of a drone is represented by the smallest rectangle surrounding the drone. A research trend is using feature descriptors such as SIFT (Scale Invariant Feature Transform), SURF (Speeded-Up Robust Features), HOG (Histogram of Oriented Gradients) for drone representation. These descriptors extract feature vectors from a set of training images with labels. A classifier such as SVM (Support Vector Machine) is trained on the extracted vectors. The classifier is then used to detect drones on sliding windows in an input image. This method has two disadvantages. The first one is that features have to be extracted skillfully to capture important information. The second one is that the sliding window technique causes computationally costly exhaustive search. In [4], Haar feature (feature achieved by Harr-like transformation), HOG feature and LBP (Local Binary Pattern) feature are used with CBC (Cascades of Boosted Classifiers) for drone detection. CBC has successive classifiers in order of their complexities. In order to reduce training time, a successive classifier is trained only on samples passing its previous classifiers.

The study in [5] uses a preprocessing approach with morphological operations on gray image to highlight potential drones and a temporal filtering approach to detect drones appearing in a long enough duration. Morphological operations are dilation and erosion. Dilation adds pixels into the boundaries of drones, while erosion removes pixels from the boundaries. After these operations, the temporal filtering approach using hidden Markov models is used to detect and track drones.

The method in [6] uses sliding window technique to divide the video into slices. Each slice has N frames. These slices overlap each other. The larger overlapping duration, the higher the accuracy. Then this method creates spatio-temporal cubes (st-cubes) with different scales. Each st-cube is represented by the parameters of width, height, and time duration. Motion compensation algorithm is used for frames in a st-cube to create a st-cube with drones at the center of the frames. Each st-cube is then classified as containing of a drone or not by

using boosted trees or convolutional neural network. If there are multiple detected drones at a position, the detected drone with the highest score is retained.

Another research trend is using deep neural networks. Studies in [7], [8], [1] propose to use beginning-to-end drone detection models based on convolutional neural networks YOLOv2 [9] and YOLOv3 [10]. The lower layers of YOLO are trained to extract high-level features. Then the features from the layers at the two highest levels are combined to get the final feature map of an input image. The feature map is divided by a grid. The first task associated with a grid cell is to predict bounding boxes and confidences that these boxes contain a drone. The second task of a grid cell is calculating conditional probability an object belonging to a class when the probability a bounding box containing an object is known.

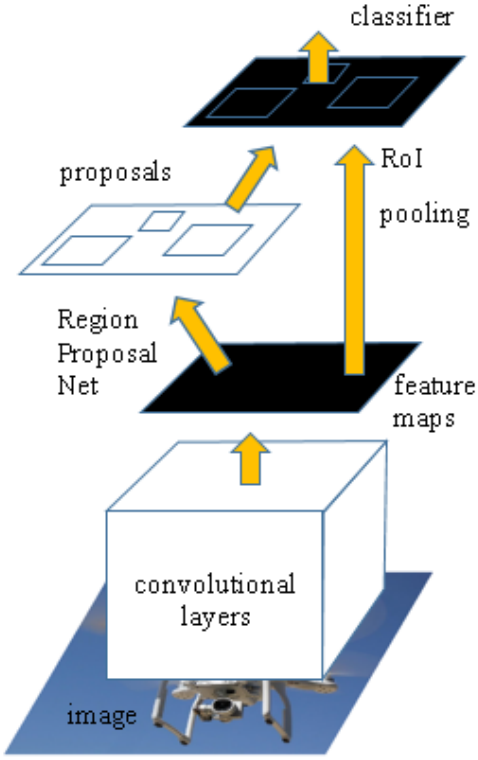

Figure 1. *Faster R-CNN* [11]

In this paper, we propose to use the deep convolutional neural network Faster R-CNN to detect drones (flycams in particular). In [11] and [12], Faster R-CNN and Fast R-CNN are applied to detect aeroplanes, but not to detect drones that are different from aeroplanes in sizes and shapes. In [13], Fast R-CNN is applied to detect drones, but the method's average precision is low (0.42). The drone detection in this paper is stated as a machine learning problem as follows. The problem is given with the input of a set of images that may contain drones or may not, where drones in an image are localized by bounding rectangles. The task of the problem is constructing a machine learning model to determine if drones exist in an image and where drones are.

The following sections include: Section 2 gives a summary of Faster R-CNN, section 3 presents experiments to determine options for Faster R-CNN to most accurately detect drones in images, and the last section is about conclusion and future work.

2. The convolutional neural network Faster R-CNN

This study uses the convolutional neural network Faster R-CNN to detect drones in images. This section presents the summary of Faster R-CNN for drone detection, (for more detailed see [11]). Faster R-CNN is the union of the region proposal network RPN and the object detection network Fast R-CNN [12]. The two networks share convolutional layers as shown in Figure 1. Fast R-CNN uses regions proposed by RPN network to detect objects. Section 2.1 introduces the design and properties of RPN network. Section 2.2 presents the algorithm for training the two networks with shared features.

2.1. Region Proposal Network (RPN)

The region proposal network RPN has the input of an image in any size and has the output of rectangular shape object proposals. Each proposal has a score that measures the membership belonging to a class (drone class

or background class). RPN network shares a set of convolutional layers with Fast R-CNN network. The output of the shared convolutional layers is a feature map as shown in Figure 2.

In order to generate region proposals, a small network with fully connected convolutional layers slides over the feature map. The small network is presented as a point in Figure 2. The small network has the input of a spatial window on the feature map. Each sliding window is mapped to a lower-dimensional feature (256-d as shown in Figure 2). This feature is then taken as the input for the two sibling fully connected layers for regression and classification respectively.

At each sliding window position, multiple region proposals are predicted, where the maximum number of proposals for each position is denoted as *k*. The regression layer (reg layer) outputs the *4k* encoded coordinates of *k* bounding boxes. The classification layer (cls layer) outputs *2k* scores estimating probability that a proposal contains an object or not. The *k* proposals are represented as *k* boxes which are called anchors.

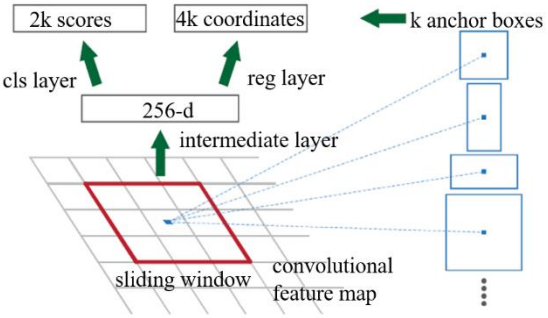

Figure 2. *Region Proposal Network (RPN)* [11]

In order to train RPN network, a binary class label (of being an object or not) is assigned to each anchor. A positive label is assigned to an anchor if the anchor has the highest IoU (Intersection-over-Union) with a ground-truth box or the IoU with a ground-truth box is in a specified range. If there are multiple anchors or no anchors satisfying the second condition, the first condition is applied. Negative labels are assigned to anchors that are not assigned positive labels if their IoU with all the ground-truth boxes are in a specified range. Anchors that are not assigned positive or negative labels have no meaning to the training objective.

An objective function of losses from classification and regression is minimized. RPN can be trained through back propagation and SGD (Stochastic Gradient Descent). SGD searches for the minimal point of the loss function through a number of epochs. At each epoch, multiple iterations are performed over the entire training set. At each iteration, the gradient descent algorithm takes a step proportional to the negative of the gradient (or approximate gradient) of the loss function at the current point using a mini-batch. A mini-batch can be obtained from a number of images containing positive and negative anchors. Positive and negative anchors are sampled with the rate up to 1:1. The gradient of the loss function is estimated using a mini-batch, instead of using a large set of anchors from all the images of the training set. This estimation speeds up the search for minimum loss.

All new layers are initialized by weights taken from Gauss distribution with mean of 0 and standard deviation of 0.01. The shared convolutional layers are initialized through a pre-trained model/network for ImageNet classification [14].

2.2. Sharing Features for RPN and Fast R-CNN

Both individually trained RPN and Fast R-CNN will correct their convolutional layers in different ways. This requires a technique that allows sharing convolutional layers between the two networks, rather than learning these networks individually. One of the techniques is the four-step alternating training algorithm. As a first step, RPN network is trained as described in section 2.1. This network is initialized through a pre-trained model for ImageNet classification and refined from

beginning to end for region proposal. In step two, the detection network Fast R-CNN is independently trained using the proposals generated by RPN from step one. This detection network is also initiated by the pre-trained model on ImageNet. At this moment, these two networks do not share convolutional layers. In step three, the detection network is used to initiate RPN training, but the shared convolutional layers are fixed and only the RPN's own layers are refined. Now, these two networks share convolutional layers. Finally, fixing the convolutional layers, Fast R-CNN's own layers are refined. Both networks share the same convolutional layers and make up a unified network.

3. Experiments and results

In this section, dataset for training and testing Faster R-CNN network for drone detection is first described. Fixed parameters for experimenting with Faster R-CNN training are then presented. Experiments to identify options for Faster R-CNN training to most accurately detect drones are lastly presented.

3.1. Training and testing dataset

The dataset for training and testing Faster R-CNN network for drone detection consists of a total of 498 images of the quadcopter DJI Phantom 3 from Google image search tool, and screenshots from videos from YouTube [13]. Of these 350 images are used for training and 148 images are used for testing.

In addition, data augmentation is used to improve the accuracy of the network through random modification of an original image during training. Data augmentation makes the training data more diverse without having to increase the number of labeled training samples. The modification is done by randomly flipping an image and the bounding boxes horizontally at each iteration of a training epoch. The testing data is not augmented. Testing is only done with original data so that evaluation is not biased. Figure 3 illustrates image creation by horizontal flip. The left image is an original image, the right image is the image created by flipping the original one.

Figure 3. *Data augmentation*

3.2. Fixed parameters

In experiments to determine options for Faster R-CNN training to detect drones accurately, we fix parameters presented in Table 1. The learning rate and the momentum coefficient are set to 0.001 and 0.09. The two coefficients affect the speed and accuracy of SGD (Stochastic Gradient Descent) method. The learning rate determines the length of each jump in finding the minimal point by SGD method. The smaller the learning rate, the more accurate the search. The momentum coefficient relates the determination of a current jump to previous jumps. This coefficient is chosen from 0 to 1. The larger this coefficient, the more the effect of previous jumps. This coefficient of zero means that a current jump has nothing to do with previous jumps. If this coefficient receives a value other than zero, the search is performed faster. The maximum number of training epochs is set to 30. The IoU range to determine an anchor box negative is [0 0.3] and the range to determine an anchor box positive is [0.6 1]. These ranges are commonly used ones [11, 15].

Table 1. *Fixed parameters*

Parameter	Value
Learning rate	0.001
Momentum co-efficient	0.09
Maximum number of epochs	30
IoU range for negative anchors	[0 0.3]
IoU range for positive anchors	[0.6 1]

3.3. Experiments to identify options for Faster R-CNN training to most accurately detect drones

In this experimental section, we look for options for Faster R-CNN training to most accurately detect drones. Options include the number of images taken from the training set to determine a mini-batch at an iteration of a training epoch to estimate the gradient of the loss function, the number of anchor boxes at each sliding window position, and the pre-trained model/network for initializing RPN and Fast R-CNN networks. We also compare the training using augmented data and not using. Finally we compare the achieved accuracy of Faster R-CNN to that of Fast R-CNN.

The evaluation of Faster R-CNN's training options is based on the average precision (AP) of the predictions on the set of all the test images. AP is a commonly used measurement for evaluating convolutional neural networks [9], [11]. To calculate AP, the set of all the predictions on the test images are arranged in descending order of the predictions' confidences. Suppose the set of all the predictions has N predictions. N sub-sets of predictions are extracted from the set of all the predictions. The k^{th} sub-set consists of predictions from 1 to k. Precisions and recalls are calculated on N subset of predictions. The average precision is approximately equal to the area under the polyline formed by points ($Recall_k$ $Precision_k$), where k is from 0 to N. In the formulas (1), $Precision_k$ and $Recall_k$ are the precision and recall of the k^{th} sub-set and AP is the average precision, where k is from 1 to N. TP_k, FP_k and FN_k are the numbers of true positives, false positives, and false negatives of the k^{th} sub-set of predictions respectively. $Precision_0$ and $Recall_0$ are set to 1 and 0, which are the precision and recall for the sub-set with no predictions.

$$Precision_k = \frac{TP_k}{TP_k + FP_k} \quad (1)$$

$$Recall_k = \frac{TP_k}{TP_k + FN_k}$$

$$AP = \sum_{k=1}^{N} Precision_k (Recall_k - Recall_{k-1})$$

We first experiment with the number of images to sample boxes containing drones or not for mini-batches. In this experiment, we fix the number of anchor boxes at each sliding window position being 2 and the pre-trained network being resnet50 [16]. The results of the experiment are presented in Table 2. The results show that the number of images of 1 is the best, where the Faster R-CNN's average precision is 0.741. This means that sampling drone boxes on a few of ground-truth boxes containing drones (only ground-truth boxes on one image) gives more accurate detectors. This can be explained by the fact that a ground-truth box is used to sample multiple drone boxes at multiple different views, so the network after training can detect drones at various views (the network is highly robust to testing data). The number of images for sampling mini-batches of 1 is chosen for further experiments.

Table 2. *Average precisions by different numbers of images for sampling mini-batches*

Number of images to sample mini-batches	Average precision
1	0.741
2	0.700
3	0.672
4	0.695
5	0.692

We then experiment with different numbers of anchor boxes at each sliding window position. In this experiment, the number of images for sampling mini-batches is chosen to be 1 and the pre-trained network is resnet50. The results of the experiment are in Table 3. We can see that the number of anchor boxes does not affect much the average precision of Faster R-CNN after training. To carry out the next experiments, we chose the number of anchor boxes to be 10, corresponding to the highest average precision of 0.744.

Table 3. *Average precisions by different numbers of anchors*

Number of	Average precision
2	0.741
4	0.707
6	0.713
8	0.723
10	0.744

We also compare the uses of different pre-trained networks including resnet50 [16], alexnet [17], googlenet [18], mobilenetv2 [19], vgg19 [20]. In this experiment, the number of images for sampling min-batches is 1 and the number of different anchor boxes at each sliding window position is 10. The experimental results in Table 4 show that vgg19 network achieves Faster R-CNN with the highest average precision of 0.774, followed by resnet50, mobilenetv2. The pre-trained networks giving Faster R-CNNs with much lower average precisions are googlenet and alexnet.

Table 4. *Average precisions by different pre-trained networks*

Pre-trained network	Average precision
resnet50	0.744
alexnet	0.502
googlenet	0.643
mobilenetv2	0.727
vgg19	0.774

In addition, we compare the use of the original data to the use of augmented data for training on the same best pre-trained network vgg16 and experimental parameters selected above. The results of this experiment show that using augmented data for training can increase the average precision by 5%. The average precision when using data argumentation is 0.774, while that when not using is 0.735.

In comparison with Fast R-CNN, the average precision of the detection method using Faster R-CNN network in this study is also significantly higher than the detection method using Fast R-CNN network performed by Reiser [13]. In Reiser's experiments on the same drone dataset, the average precision is 0.420. Thus, the average precision of Faster R-CNN in this study is 83% higher than that of Fast R-CNN (0.774 compared to 0.420).

4. Conclusion

In this paper, we conducts empirical studies on training the deep convolutional neural network Faster R-CNN to most accurately detect drones (flycams in particular). Through experiments, we found that the number of images to sample a mini-batch for each training iteration being 1 is the best for training Faster R-CNN. This means that a ground-truth box containing a drone sampled multiple times from different views will make the detector obtained after training more adaptable. The number of anchor boxes at each sliding window position does not affect much the Faster R-CNN's average precision for drone detection. The best pre-trained network for training Faster R-CNN to accurately detect drones is vgg19, followed by resnet50, and mobilenetv2. The pre-trained networks giving Faster R-CNNs with much lower average precisions are googlenet and alexnet. Training data augmentation increases the average precision of Faster R-CNN by about 5%. With the best training options determined through experiments, Faster R-CNN can detect drones with the average precision up to 0.774, which is 83% higher than Fast R-CNN with the average precision of 0.420.

Our next research direction is to research and improve the time of training Faster R-CNN and detecting drones. We also plan to study and develop datasets to make drone detection more accurate. The obtained drone detector will be then integrated into drone detection, warning and defense systems in sensitive areas.

REFERENCES

[1]. E. Unlu, E. Zenou, N. Riviere, and P.-E. Dupouy, "Deep learning-based strategies for the detection and tracking of drones using several cameras," *IPSJ Transactions on*

Computer Vision and Applications, vol. 11, no. 7, pp. 1-13, 2019.

[2]. NovoQuad, "ND-BU001 Standard Anti-Drone System," 2020. [Online]. Available: https://www.nqdefense.com/products/anti-drone-system/nd-bu001-standard-anti-drone-system/. [Accessed Mar. 15, 2020].

[3]. DRONESHIELD, "DroneSentry: Autonomous Drone Detection & Countermeasure," 2020. [Online]. Available: https://www.droneshield.com/sentry. [Accessed Mar. 15, 2020].

[4]. G. Fatih, Ü. Göktürk, S. Erol, and K. Sinan, "Vision-Based Detection and Distance Estimation of Micro Unmanned Aerial Vehicles," *Sensors*, vol. 15, no. 9, pp. 23805-23846, 2015.

[5]. L. Mejias, S. McNamara, J. Lai, and J. Ford, "Vision-based detection and tracking of aerial targets for UAV collision avoidance," IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, 2010.

[6]. A. Rozantsev, V. Lepetit, and P. Fua, "Detecting Flying Objects Using a Single Moving Camera," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 5, pp. 879-892, 2016.

[7]. C. Aker, and S. Kalkan, "Using Deep Networks for Drone Detection," IEEE International Conference on Advanced Video and Signal Based Surveillance, Lecce, Italy, 2017.

[8]. M. Wu, W. Xie, X. Shi, P. Shao, and Z. Shi, "Real-Time Drone Detection Using Deep Learning Approach," International Conference on Machine Learning and Intelligent Communications, Hangzhou, China, 2018.

[9]. J. Redmon, and A. Farhadi, "YOLO9000: better, faster, stronger," IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 2017.

[10]. J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," 2018. [Online]. Available: arXiv:1804.02767. [Accessed Mar. 15, 2020].

[11]. S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," Conference on Neural Information Processing Systems, Montréal Canada, 2015.

[12]. R. Girshick, "Fast R-CNN," IEEE International Conference on Computer Vision, Santiago, Chile, 2015.

[13]. C. Reiser, "Bounding box detection of drones (small scale quadcopters) with CNTK Fast R-CNN," 2017. [Online]. Available:https://github.com/creiser/drone-detection. [Accessed Mar. 15, 2020].

[14]. O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211-252, 2015.

[15]. D. Zhou, F. J., X. Song, C. Guan, J. Yin, Y. Dai, and R. Yang, "IoU Loss for 2D/3D Object Detection," International Conference on 3D Vision, Québec, Canada, 2019.

[16]. K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 2016.

[17]. A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," Conference on Neural Information Processing Systems, Navada, USA, 2012.

[18]. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going Deeper With Convolutions," IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 2015.

[19]. A. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," 2017. [Online]. Available: arXiv:1704.04861. [Accessed Mar. 15, 2020].

[20]. K. Simonyan, and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," International Conference on Learning Representations, San Diego, CA, USA, 2015.