MJCIS Vol.15 No.1 Jun 2019

Mansoura Journal of Computers and Information Sciences

Multiple Pedestrian Detection Depending on Faster Region-based Convolutional Neural Network (RCNN)

Ghalia Shariha
Information Technology Dept., Faculty of Computers and Information, Mansoura University, Egypt
ghalia2050@yahoo.com

Mohammed Elmogy
Information Technology Dept., Faculty of Computers and Information, Mansoura University, Egypt
melmogy@man.edu.eg

Eman El-Daydamony
Information Technology Dept., Faculty of Computers and Information, Mansoura University, Egypt
eman.8.2000@gmail.com

Ahmed Atwan
Information Technology Dept., Faculty of Computers and Information, Mansoura University, Egypt
atwan_2@yahoo.com

ABSTRACT

Pedestrian detect plays a crucial role in security, intelligent surveillance, vehicles, and robotics. Occlusion handling is a challenging worry in tracking multiple people. The tracking is based on the highest accuracy object detectors. In the current paper, we proposed a framework that detects multiple pedestrians in the image, which depends on Faster Region-based Convolutional Neural Network (R-CNN). We applied the transfer learning concept by using the VGG19 & VGG16 deep networks, which are trained before on Image-Net to extract the feature map. Relying on trained weights, to reduce the time of training, we used the transfer learning concept. The framework was tested on Penn-Fudan pedestrian database. The pedestrian detection accuracy was measured by using the area under the curve (AUC) of the receiver operating characteristic (ROC) that e is achieved 95.6%. In addition, the proposed system achieved Miss Rate (MR) equals 1.98, accuracy (ACC) equals 97.31%, and F1-score equals 93.17%. The achieved results show the promise of our proposed technique to detect multiple pedestrians in a single scene.

Keywords: Pedestrian Detection; Multiple Pedestrians; Deep Learning; Faster Region-based Convolutional Neural Network (RCNN).

1. INTRODUCTION

Pedestrian detection is considered a branch under target detection, and the tag it detects is a pedestrian. The target detection is finding the site of an object in a given picture and marking the category of the object. Moreover, it is the first phase for several applications, such as driving assistance systems, intelligent digital content management, and smart video surveillance. The problem of target detection is to detect the position of the target with knowledge of its class. However, this problem is not very easy to be addressed. The object can also be in multiple categories, such as color, illumination, scale, and pose variance.

Consequently, this may be done by looking for still images and indexing them because they contain the object in different position, size, and background. The detection target has two directions: traditional and deep learning algorithms. Typical representatives of traditional algorithms are discussed in many studies, which include an HoG (Histogram of Oriented Gradients) that was formulated by Dalal et al. [1]. HoG feature is fast and fitting for many real problems. The authors run the sliding window on the pyramid on every window. Then, they calculated the features of HoG that are fed to the support vector machines (SVM) classifier. They could run this within the real-time on videos for the system of pedestrian detection, face detection, and other object detection cases. Viola et al. [2] developed a person detector that could move efficiently by use of the AdaBoost algorithm [3]. A chain of complex and progressive location rejection regulations was trained on the foundation of Haar-like aspects [4], space-time variations, and wavelets. Papageorgiou et al. [5] proposed a system that could detect pedestrians based on SVM with polynomial kernel and Haar wavele. The descriptors of input images are indicated as variant-based sub-windows [6].

Pedestrian testing based on deep learning typically is represented on the Region-based Convolutional Neural Network (R-CNN) series of algorithms. RCNN is the most widely used deep learning-based pedestrian detection algorithm. According to Girshick et al. [7], R-CNN consists of two stages. The first step, the Selective Search (SS), helps in identifying a manageable bounding-box object for region candidates [8]. Then, it reveals the features of a CNN for classification. R-CNN was slow because the CNN is running on approximately 2000 regions produced by SS.

With Spatial Pyramid Pooling (SPP)-net, Zhang et al. [9] calculated the representation of CNN for the entire image only one time. They used it for calculating the CNN representation for every patch produced by SS. This may be done via applying a pooling type of operation on this section of the maps of last convolutional (Conv) layer that is identical to the region. The rectangular section of Conv layer is identical to a region that may be calculated via projecting the region on Conv layer and focusing on the downsampling occurrence in the intermediate layers referring to dividing the coordinates via sixteen in case of VGG. Despite being a significant drawback with SPP-net, it is not easy to perform back-propagation by spatial pooling layer. Girshick et al. [10] used the ideas from SPP-net and using R-CNN to improve the training process by integrating three separate models into a single training framework and increments results of shared computation that called Fast R-CNN. Rather than eliciting the vectors features of CNN separately for every area proposal, this algorithm collected those into single CNN precede passing the whole picture, and the region proposals share this matrix of a feature. Then, the same features are divided for learning the classifier of an object and the represented bounding-box.

Lin et al. [16] presented a simple framework to build feature pyramids inside CNN that is called feature pyramid network (FPN). The algorithm revealed betterment importantly in many robust baselines. Therefore, it presents the feature pyramids application and a practical solution for research regardless of the requirements of calculated pictures pyramids. Finally, the algorithm proposed that there is a robust representational force of deep CNN and their robustness in the sizes difference, but it is crucial to solving the multiscale problems by using pyramid representations. He et al. [15] proposed a mask R-CNN that expanded faster R-CNN to segment images on the pixel level. The major point was decoupling the categorization and mask at the pixel level. Depending on the structure of faster-RCNN algorithm, they added a third section to predict a target mask combined with the sections for localization and classification. The mask section was small and used a fully connected network that is utilized in every region of interest (RoI). They developed the RoI pooling layer that may be best mapped to the region's target on the original picture.

When conducting an appraisal for the work illustrated above in the object detection location, there are two main challenges, which should be taken into consideration while developing a new pedestrian tracking system. First, high accuracy is needed for pedestrian detection especially in cluttered scenes and illumination variations. Second, the fast detection rate is required to make use of real-time applications.

To overcome the above limitations, our proposed system used faster R-CNN algorism for detection pedestrian with a small dataset. It reduces the train time when taking weight learning from VGG19 architecture of CNN. In other words, we applied the transfer learning technology to reduce the processing time.

The present paper is structured as follows. Section two discusses some basic concepts. Section three presents the proposed framework for constructing the suggested system. Section four reveals the experiment and results. Finally, Section five lays out the conclusion and directions for our future work.

2. Basic Concepts

The pedestrian detection system includes many methods, even though these methods have a pipeline through which they can be compared. The pedestrian detection system is separated into three key stages, which are feature extraction, region proposal, and classification of the region.

2.1 Convolutional Neural Network (CNN)

CNN is a popular algorithm for image classification and typically comprises of convolution layers, activation function layers, pooling (max pooling or average pooling) layers to reduce dimensionality without losing many features. There is a feature map generated by the last layer of a convolutional layer. Many pre-trained models are developed to directly use them without going through the pain of training models due to computational limitation. Many models got popular as well like Le-Net [23], Alex-Net [19], ZF-Net [24], VGG-Net [12], Google-Net/Inception [25], and Res-Net [18], by Image Net [17] that has more than a million images.

The selected models for this paper were VGG16 and VGG19. The VGG19 network has forty-seven layers, nineteen layers with weights are learnable sixteen layers of convolutional, and followed with three fully connected (FC) layers. Moreover, the structure of VGG16 network contains forty-one layers followed with sixteen layers with learnable weights, thirteen convolutional layers, and 3 FC layers. Both of them may classify images into one thousand object categories like a mouse, keyboard, pencil, human, and several other animals. Consequently, the network presented rich representations of a feature for many images.

2.2 Faster (R-CNN)

The Faster R-CNN algorithm [11] is classified into two phases. The first phase produces a variety of candidate locations of the target combined with small convolutional network Proposal Network (RPN) [13]. The second phase divides every site candidate as a background or foreground categories using a CNN.

The main idea in the faster algorithm is generating candidate regions by using RPN, and it is quicker than the SS and edge box (EB) [20]. Through alternating training, the RPN with Faster R-CNN networks shares parameters. The Fast R-CNN algorithms use a full image as input and a group of region suggestion. The network processes all pictures with many Convo layers. Moreover, max-pooling layers to generate a Convolution feature map.

We proposed distinctive sizes regions after RPN. These regions refer to several sizes of CNN feature maps. It is difficult to produce an influence structure to perform the task on the features of several sizes. The interest Pooling region may solve the problem by decreasing the feature maps into equal size, unlike Max-Pooling that has a fixed size. RoI pooling splits the input feature map to a fixed group of equal regions and applies Max-Pooling on each area. Thus, the production of RoI Pooling is fixed without consideration of the size of the input.

Then, every vector features are entered into a series of FC layers, which are classified into two output layers. One generates estimates of softmax probability on pedestrian class and background class. The second is four real value for each pedestrian detected. Every four values encode refined bounding-box location for pedestrian detected.

Algorithm 1: The following pseudo-code summarizes the Faster R-CNN method.

```
1. Inputs: Images 224*224*3
2. feature_maps = process(image)
3. ROIs = region_proposal(feature_maps)
4. for ROI in ROIs
5.     patch = roi_pooling(feature_maps,ROI)
6.     class_scores, box = detector(patch)
7.     class_probabilities = softmax(class_scores)
8. end
9. Outputs: Classifications and bounding box
   coordinates of objects in the images.
```

3. The Proposed Framework

This work aims to detect multiple pedestrians from sigle scene. The basic idea of the proposed system depends on trained Faster R-CNN algorithm [11] by modifying the VGG19-network to be used for detection. The transfer learning is used to reduce learning time be training two layers just with changed hyper- parameterizations of learning, as listed in Table 1.

Table 1: The parameter values of the trained VGG-Net.

Feature	Value
Size of input	$224 \times 224 \times 3$
Number of epochs	$10/20/40$ Epochs and *three* sub Epoch
Layer numbers	$47\ in\ vgg19\ \&\ 41\ in\ vgg16$
Learning rate	$le-4$

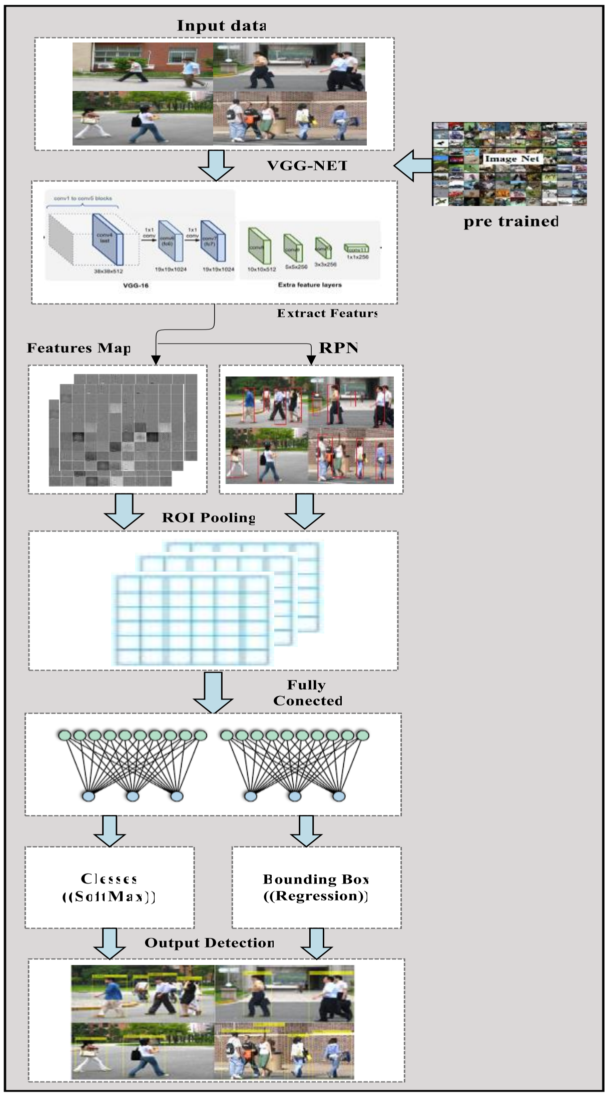

Fig. 1: The proposed framework for multiple pedestrians detection based on Faster R-CNN.

The algorithm with the labeled dataset for pedestrian used VGG19 pre-trained deep architecture network, in which we retrain a pre-trained network to classify Penn-Fudan pedestrian dataset [14] images and change the final layers.

The proposed method can be used for detecting both small-scale pedestrians and high-resolution person regions. The detection process can be adjusted by giving information regarding the possible size of pedestrians in the video frames in order to improve the detection rate.

In the first step, the role of the RPN network is inputting an image and outputting a batch of rectangular candidate regions. The input image is represented as $heigh \times width \times Depth\ that$ goes (of any size) through a VGG16 and VGG-19 CNN pre-trained for the task of classification (Image-Net), which will output a set of convolutional feature maps from the last convolutional layer.

The anchor is the core of the RPN network. A sliding window is turned on featured maps. The window sliding size is $n \times n$, for every window create a group of nine anchors in each anchor that have a midpoint(x_a, y_a) but with 3 diverse aspect ratios (AR) and 3 diverse scales, as shown in Fig. 2. Note that all these coordinates are computed with respect to the original image. A value p^* is computed for each anchor, which is specific for these anchors interfere with the ground-truth bounding boxes.

$$p^* = \begin{cases} 1\ if\ IoU > 0.7 \\ -1\ if\ IoU < 0.3 \\ 0\quad otherwise \end{cases} \tag{1}$$

where (IoU) is intersection over union is defined below:

$$IoU = \frac{Anchor \cap GTBox}{Anchor \cup GTBox} \tag{2}$$

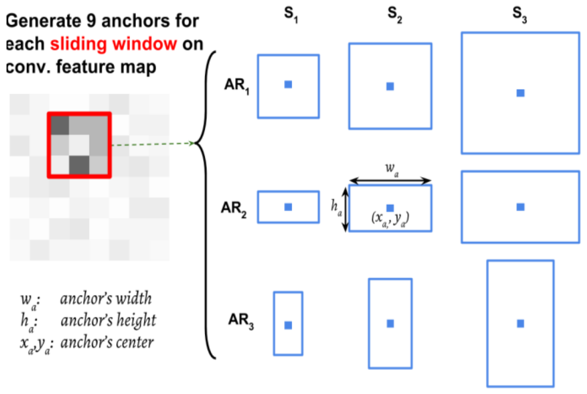

Fig. 2: The generation of the anchors.

The RoI pooling stage usages the max pooling to change the features in any valid zone of interest to a few feature map with a constant spatial extent of $H \times W$ where H and W are layer hyper-parameters that are separate of any specific RoI. every RoI is described by a four-tuple (r, c, h, w) that correspond its top-left corner (r,c), and its height and width (h, w).
The 3×3 locative features are extracted from convolutional features maps. They are entered into a smaller network to a regression layer (reg). The output from this layer locating a bounding-box (x, y, w, h), and classification layer (cls), the output of this sub-network layer is a probability p specify whether the expected bounding-box include a pedestrian is (1) or (0) for the background.

A faster R-CNN uses multi-task cost function combines the losses of classification and bounding box regression:

$$\boldsymbol{L = L_{cls} + L_{reg}} \tag{3}$$

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \boldsymbol{\lambda} \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*) \tag{4}$$

Here, i refer to the list of an anchor in a small-batch and p_i mention to the expected prospect of anchor i being a pedestrian. The ground-truth label p_i^* is 1 when the anchor is positive and 0 when it is negative anchor. t_i is a vector that represents coordinates with 4 parameters for the predicted bounding box. t_i^* is the ground-truth box associated for a positive anchor. The 2 expressions are set by N_{cls} and N_{reg} and weighted by a balancing parameter ($\boldsymbol{\lambda}$).

The cost function is divided into two parts, corresponding to the two branches of RPN, namely the classification error of the target or not and the regression error of bbox, where $L_{reg}(t_i,t_i^*) = R(t_i - t_i^*)$ is used in Fast-RCNN. Note that L_{reg} is multiplied by p_i in the regression error, so the bbox regression only calculates the error for the anchor containing the target. In other words, if the anchor does not contain the target, the box output location does not matter. So, for bbox's ground truth, only consider the anchor that is determined to have a target and use the coordinates of the label as ground truth. In addition, when calculating the bbox error, it is not the coordinates of the four corners, but $t_y, t_x, t_h,$ and t_w . The specific calculation is as follows:

$$t_{y=(y-y_a)/h_a}, \quad t_y^* {=(y^*-y_a)/h_a}$$

$$t_{x=(x-x_a)/w_a}, \quad t_{x=(x^*-x_a)/w_a}^*$$

$$t_{h=\log(h/h_a)}, \quad t_{h=\log(h^*/h_a)}^* \qquad (5)$$

$$t_{w=\log(w/w_a)}, \quad t_{w=\log(w^*/w_a)}^*$$

4. The Experimental and Results

We tested the proposed framework on Penn-Fudan data-set. It contains images of pedestrians. There are 170 images with 345 labeled pedestrians. The heights of labeled pedestrians in this database fell into 180 x 390 pixels and separated the dataset into 60 percent for training, 30 percent for testing, and 10 percent for validation. We implemented our system on a Windows 10 plateform with Intel(R) Core (TM) i7-7700K at 4.20 GHz, 16 GByte RAM, and also a single 11GB memory GPU NVIDIA GeForce GTX1080Ti. All algorithms are implemented by using Matlab2018a.

We assessed the implementation of the proposed framework by using the area under the ROC curve (AUC) [26], satisfactory, specificity, sensitivity, accuracy (ACC), and F1-score. We are plotting the curve of performance based on true positive rate (TPR) and false positive rate (FPR). These metrics were calcuated by the True Positive (TP), False Positive (FP), True Negatives (TN) and False Negatives (FN) as in Eqs. 6-11. Fig. 3 shows some examples of the detection of pedestrian by using our proposed system.

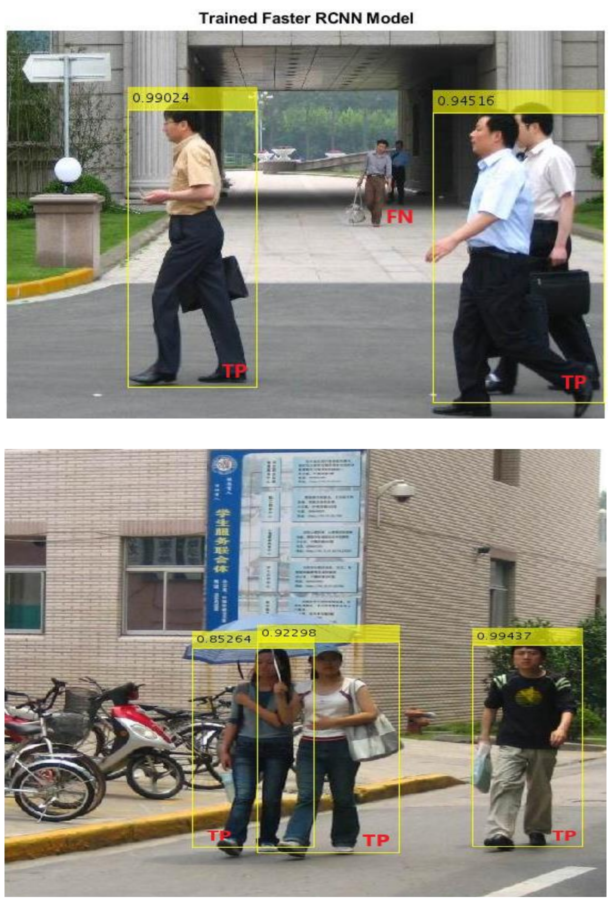

Fig. 3: Some output examples from our proposed system.

TP is the whole number of actual pedestrians detected. TN is a whole number of actual backgrounds detected. FP is a whole number of a background region detected incorrectly. FN is a whole number of a real pedestrian's area is missed.

TPR is called recall or sensitivity that is calculated by Eq. 6:

$$\boldsymbol{TPR} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}} \qquad (6)$$

Similarly, TNR is called also selectivity or specificity, which is obtained Eq. 7.

$$\boldsymbol{TNR} = \frac{\boldsymbol{TN}}{\boldsymbol{FP+TN}} \qquad (7)$$

$$AUC = 0.5\ (TPR + TNR) \qquad (8)$$

$$ACC = \frac{TP+TN}{TP+TN+FP+FN} \qquad (9)$$

$$F1 - score = \frac{2TP}{2TP+FP+FN} \qquad (10)$$

$$MR = FN/\mathrm{FN+TP} \qquad (11)$$

To draw the ROC curve , the TPR is presented versus the FPR, which is calculated from Eq. 12.

$$FPR = \frac{TP}{TP+FP} \qquad (12)$$

The x-axis is presented by FPR and y-axes represents TPR, as shown in Fig. 4.

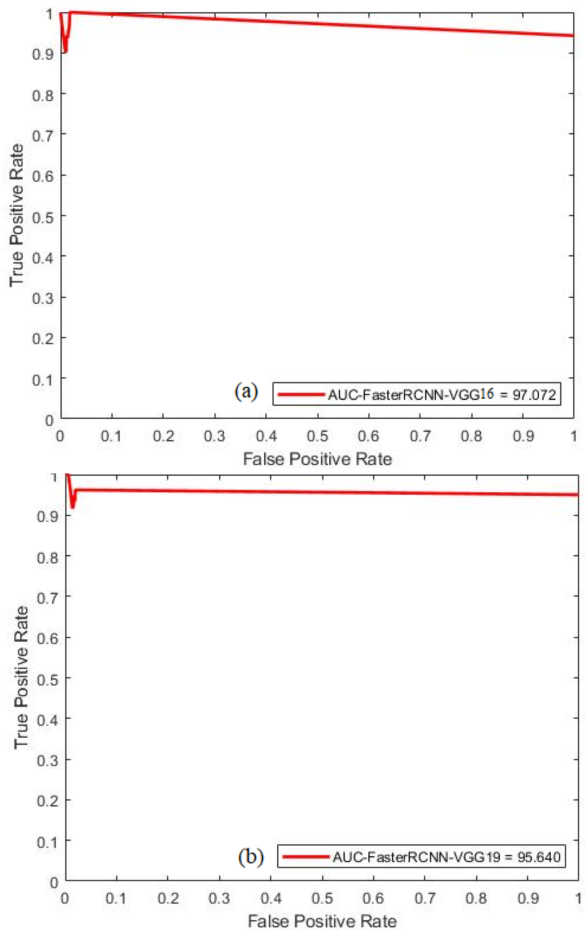

Fig. 4: The ROC curve of our proposed system: (a) for VGG16, (b) for VGG19.

We used VGG16 and VGG19 to train the RPN. The achieved accuracy is 97.31% from using RPN+VGG16 and 97.31% by using RPN+VGG19. This is a very good result because it references that the proposal quality of RPN+VGG19 and RPN+VGG16 are identical in accuracy. However, there is a clear difference in the MR, where we find that RPN+VGG19 better than RPN+VGG16. In Table 1, we summarized the results of the system with ACC, F1-score, and MR when trained by VGG-16 and VGG-19. Zhang et al. [22] used Faster R-CNN to extract features by ZF. After that, they built combination to extracted regions by RPN and K-means clustering and achieved an accuracy of 92.7%, which was tested on the INRIA dataset of pedestrians.

Table 2 and Fig. 5 list experiments result of our proposed model after fine tuning by modified learning rate with transfer weights from Image-Net classification object to detect pedestrians. It can be noted that we used Faster-RCNN algorithm with different epochs to achieve the best result on epoch 20 with 3 sub epochs. It is also possible to say that good results can be obtained when using a small number of data using a transfer learning technique.

Table 2. The performance evaluation of Faster R-CNN with two architectures with Penn-Fudan dataset.

Network	Epochs	ACC	F1	MR
VGG-16	2	95.33%	87.09%	5.0%
	10	97.12%	92.51%	2.65%
	20	97.31%	93.05%	2.37%
	40	97.5%	93.7%	1.98%
VGG-19	2	95.19%	87.08%	4.49%
	10	96.84%	91.87%	2.55%
	20	97.31%	93.17%	1.98%
	40	95.9%	90.12%	1.56%

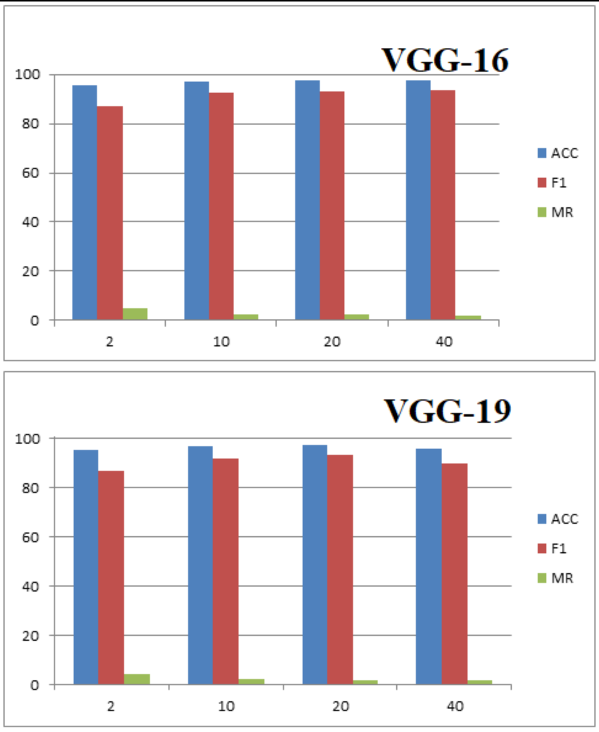

Fig. 5: The performance rating of our proposed system.

We run the experiment result ten times for training and testing with faster R-CNN with and without transfer learning. In which, we toke a pre-trained network on ImageNet. It was used as a beginning step to learn a detected pedestrian mission. Fine-setting a network with learning transfer is it easier and faster than training a network at randomly initialize weights from scratch. Moreover, the average of our proposed took 48 hours without transfer learning and two hours with transfer learning, as displayed in Table 3.

Table3: The Result of Average Time for Training.

Model (Feature extraction)	Time
The proposed system without transfer learning	48 hours (2 days)
The proposed system with transfer learning	2 hours

We run Aggregate Channel Features (ACF) to detect people, which are proposed by Piotr Dollar et al. [27, 28, and 29] on Penn-Fudan datasets. We compared our framework with their system, and the results are listed in Table 4 and Fig. 6. Our framework achieved outperformed their result. It achieved ACC of 97.31%, F1-score of 93.17%, and MR equals to 1.98. Also, Fig. 7 shows the ROC curve of our framework by using ACF.

Table 4. The performance of our framework with ACF.

algorithm	ACC	F1	MR
ACF	92.08%	82.24%	2.12%
Our Faster R-CNN	97.31%	93.17%	1.98%

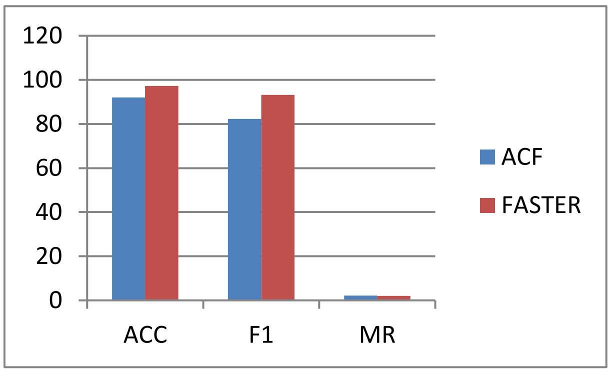

Fig. 6: The accuracy of our framework with ACF in pedestrian detection on Penn-Fudan datasets.

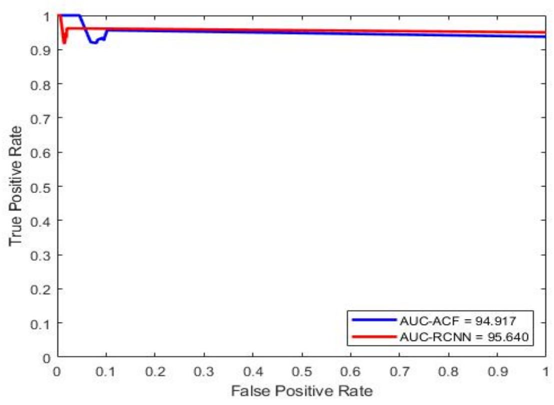

Fig. 7: The ROC curve of our framework with ACF in Penn-Fudan datasets.

5. Conclusion

In the current paper, we detected multiple pedestrian based on Faster R-CNN for extracting the pedestrian and get their localization. We trained Faster R-CNN model with the Penn-Fudan pedestrian dataset. We used a pre-trained VGG16 & VGG19 convolutional architecture. We reached somewhat satisfactory results at in AUC that is achieved 95.6 %, MR =1.92, ACC = 97.3%, and F1-score = 93.17% and also reduced the training time to 2 hours. In future work, we will train the R-CNN with additional data. Also, we are planning to apply the Kernelized Correlation Filter algorithm to tracks of each pedestrian in the image frame. To improve the existed algorithm components, we may look for a more reliable prediction algorithm.

6. REFERENCES

[1] Navneet D, Bill T (2005) Histograms of Oriented Gradients for Human Detection. In: IEEE Computer Society, pp 886-893.

[2] Paul V, Michael J, Daniel S (2003) Detecting Pedestrians using patterns of motion and appearance. In: ICCV, pp 734-741.

[3] Yoav F, Robert E (1997) A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. In: Journal of computer and system sciences, vol 55, pp 119_139.

[4] Haar A (1910) Zur Theorie der orthogonalen Funktionensysteme. In: Mathematische Annalen, vol 69, pp 331-371.

[5] Constantine P, Tomaso P (2000) A trainable system for object detection. In: IJCV, vol 38(1), pp15-33.

[6] Anuj M, Constantine P, Tomaso P (2001) Example-based object detection in images by components. In: TPAMI, vol 23(4), pp 349-361.

[7] Ross G, Jeff D, Trevor D, Jitendra M (2015) Region-based convolutional networks for accurate object detection and segmentation. In: TPAMI.

[8] J. Uijlings, K. van de Sande, T. Gevers, A. Smeulders (2013) Selective search for object recognition. In: IJCV.

[9] Kaiming H, Xiangyu Z, Shaoqing R, Jian S (2015) Spatial pyramid pooling in deep convolutional networks for visual recognition. arXiv:1406.4729v4.

[10] Ross G (2015) Fast R-CNN. In: IEEE International Conference on Computer Vision (ICCV).

[11] Shaoqing R, Kaiming H, Ross G, Jian S (2016) Faster R-CNN: Towards real-time object detection with region proposal networks. arXiv:1506.01497v3.

[12] Karen S, Andrew Z (2015) Very deep convolutional networks for large-scale image recognition. In ICLR.

[13] Daniel K, Michael A, Christian J, Georg L, Heiko N, Michael T (2017) Fully Convolutional Region Proposal Networks for Multispectral Person Detection. In: IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp 243-250.

[14] Liming W, Jianbo S, Gang S, I-fan S (2007) Penn-Fudan Database for Pedestrian Detection and Segmentation. In: ACCV [Online]. Available: https://www.cis.upenn.edu/~jshi/ped_html/.

[15] Kaiming H, Georgia G, Piotr D, Ross G (2018) Mask R-CNN. arXiv:1703.06870v3.

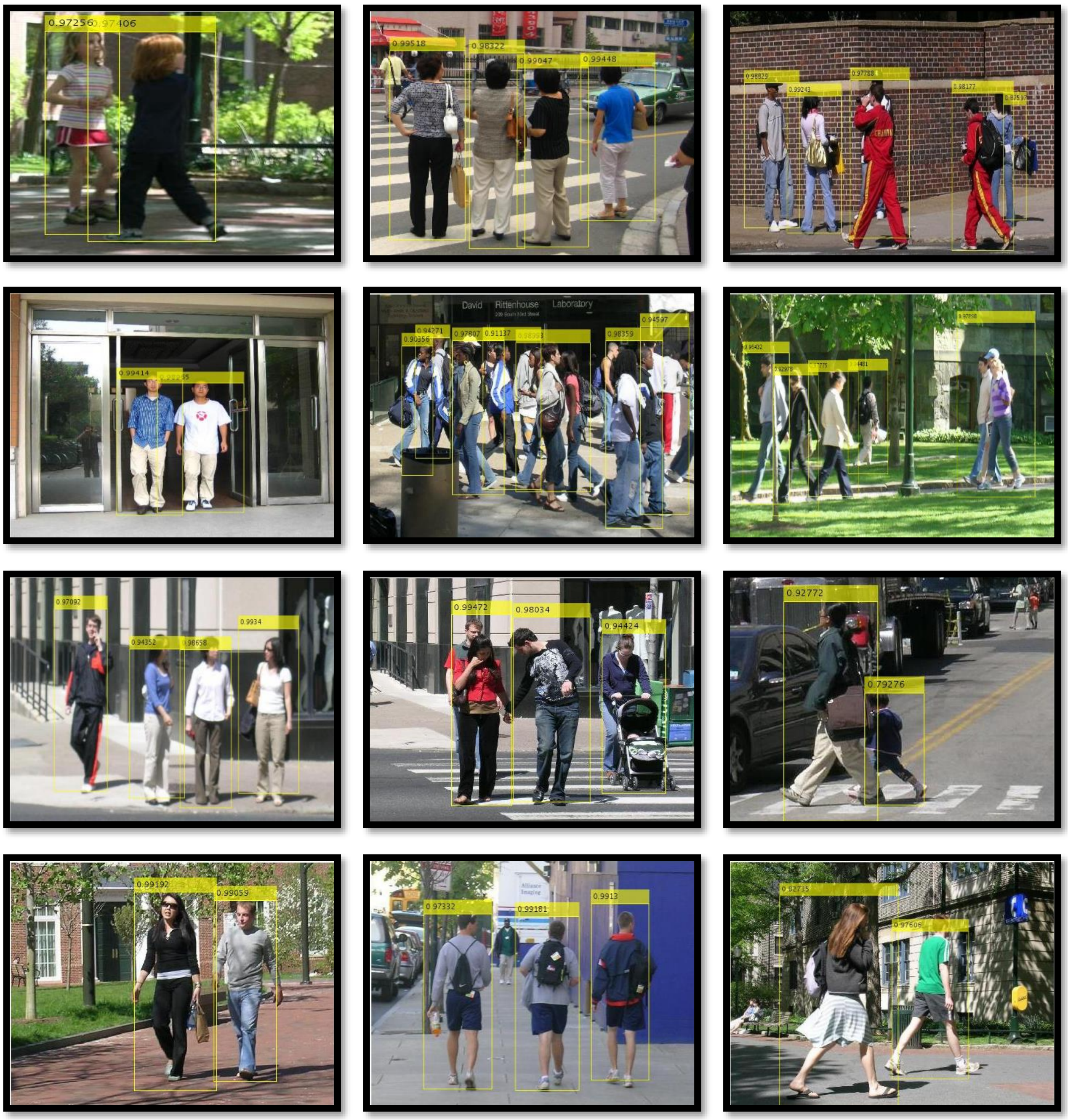

Fig. 8: Some experimental results of pedestrian detection on the Penn-Fudan pedestrian Dataset using the Faster R-CNN system.

[16] Tsung Y, Piotr D, Ross G, Kaiming H, Bharath H, Serge B (2017) Feature Pyramid Networks for Object Detection. In: CVPR, pp 2117-2125.

[17] ImageNet. http://www.image-net.org.

[18] He K, Xiangyu Z, Shaoqing R, Jian S (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp 770-778.

[19] Russakovsky O, Deng J, Su H (2015) ImageNet Large Scale Visual Recognition Challenge. In: International Journal of Computer Vision (IJCV). Vol 115, Issue 3, pp 211-252.

[20] Lawrence Z, Piotr D (2014) Edge boxes: Locating object proposals from edges. In: European Conference on Computer Vision (ECCV).

[21] Alex K, Ilya S, Geoffrey E (2012) ImageNet Classification with Deep Convolutional Neural Networks. In: Image-net large scale visual recognition (ILSVRC).

[22] Hui Z, Yu D, Shurong N, Shuo Y, Yonghua Z, Chen D (2017) Pedestrian detection method based on Faster R-CNN. In: 13th International Conference on Computational Intelligence and Security.

[23] Yann L; leon B; Yoshua B; Patrick H (1998) Gradient-Based Learning Applied to Document Recognition. In: IEEE.

[24] Matthew Z; Rob F(2013)Visualizing and Understanding Convolutional Networks. In: arXiv:1311.2901v3 [cs.CV].

[25] Christian S; Wei L; Yangqing J, Pierre S; Scott R; Dragomir A; Dumitru E; Vincent V, Andrew R;(2015) Going Deeper with Convolutions. In: CVPR.

[26] Fawcett, Tom (2006)An introduction to ROC analysis. In: Pattern Recognition Letters.

[27] Piotr D; Ron A; Serge B; Pietro P(2014) Fast Feature Pyramids for Object Detection. In: IEEE.

[28] Piotr D; Christian W; Bernt S; Pietro P (2012) Pedestrian Detection: An Evaluation of the State of the Art. In: IEEE.

[29] Piotr D; Christian W; Bernt Sc; Pietro P 2009) Pedestrian Detection: A Benchmark. In: IEEE.