Human Behaviour Recognition Using Deep Learning

Jia Lu, Wei Qi Yan and Minh Nguyen
Auckland University of Technology
Auckland 1010 New Zealand

Abstract

Traditional human behaviour recognition is mostly based on global features of digital images. Nowadays, with the increase of computing power and processing capacity, deep neural networks (DNNs) acquire a high possibility to detect any objects, which have effectively led to a new era of machine learning. In this paper, we investigated a human behaviour recognition using deep learning based on YOLOv3 model. After a number of experiments conducted, our YOLOv3 model had shown to achieve 80.20% of accuracy in human behaviour recognition with the speed of approximate 15 fps using GPU acceleration. Our direct contributions are: (1) data augment and collection, (2) adjusting deep neural network structures, and (3) superior performance in evaluations for our proposed deep learning model.

1. Introduction

In the past few decades, traditional human behaviour recognition is mostly based on the global features of an image, it primarily adopts various static features such as edge features, shape features, statistical features, or transform features to describe human behaviour. Representative features include Haar wavelets [3], Histogram of Oriented Gradient (HOG)[2], Local Binary Pattern (LBP) [26], Edgelet features [23]. Nowadays, with the increase of computation and processing capacity of devices, Deep Neural Networks (DNNs) acquire a high possibility to detect any visual objects, which lead to a new era of machine learning [15][24].

The state-of-the-art technologies of human behaviour recognition is based on the DNNs such as Convolutional Neural Networks (CNN), R-CNN, Fast R-CNN, Faster R-CNN, and Single Shot MultiBox Detector (SSD). The process of a DNNs will not be as same as traditional classifiers [14]. Bounding boxes were assumed at the first time. Thus, DNNs will resample these proposed pixels for each box and apply a high-quality classifier. Moreover, region proposal methods and region-based convolutional neural networks (R-CNNs) are also achieved successes in object detection [19]. Selective search [22] is leading an outstanding achievement in human behaviour recognition competitions based on Faster R-CNN [19][8].

Deep learning can be employed to train either supervised models or unsupervised models [10]. These models can learn a hierarchical structure of features and extract high-level features from lower level [12][9]. CNNs [12] is a type of deep learning models that use trained filters and pooling operations at local area on the original input image; thus, a hierarchical structure of more complex features. Moreover, after a proper regularization during the training [25], CNNs can perform exceptional excellence on the object recognition tasks. Unlike the traditional classifiers, CNNs are not required to preprocess data and generally provide high classification rate; moreover, it shows an invariant to variations such as illumination, pose [13].

Hence, CNNs have been demonstrated as a valid model than any other traditional models in image content, image recognition, detection and retrieval [16][1], with the technology that can help millions of parameters extended into network and a large number of labeled datasets that support the learning process. Deep learning becomes more and more possible to be implemented; moreover, that can be seen as the key factors behind these successes.

In 2015, Ren et al. [19] introduced a Region Proposal Network (RPN) combined with Fast R-CNN in real-time object detection. The algorithm mainly solved two problems: the RPN was trained for the end-to-end which can generate high-quality regions very quickly; it also makes RPN and Fast R-CNNs sharing the convolutional features by using the "attention". By using RPN to replace the selective search method, it merges RPN and Fast R-CNNs into a single network which improves the detection performance in speed. With the trained RPN, region proposal quality was improved; therefore, the overall rate of human behaviour recognition is increased.

This paper is organized as follows. Literature review is presented at Section 2, our method is described in Section 3. Our results are demonstrated in Section 4, the conclusion is drawn in Section 5.

978-1-5386-9294-3/18/$31.00 ©2018 IEEE

2. Related Work

There were a variety of methods for human behaviour recognition including traditional methods which were based on extracted features of images. The state-of-the-art methods of object detection methods by using deep learning which can process and detect a tremendous amount of image data with the minimum latency. In this section, we will mainly focus on the object detection methods by utilizing deep learning.

LeCun et al. [11] implemented the backpropagation (BP) algorithm through a multilayer network to recognize the handwritten zip code; the principle is to reduce the none-used parameters without overly reducing its computing power. In 1998, LeCun et al. proposed a LeNet-5 model [12] which is the formal formation of the CNNs.

Szegedy et al. [21] addressed object detection as a regression problem by using Deep Neural Networks. Girshick et al. [6] mentioned that the deformable part models (DPMs) could formulate as a CNNs though the DPMs has 43% of the mAP in PASCAL VOC2007. After this, Girshick et al. [5] proposed Region-based Convolutional Neural Networks (R-CNN) that addresses the problem by using region proposal [20] to get several local regions of an image and then use CNN to get these features of each region. A classifier is also added to the features and determine whether the feature corresponds to a specific object or a background. However, there still has a great deal of disadvantages, such as training is a multi-stage pipeline [4], local regions need to be extracted in advance which occupying a significant amount of disk space; traditional CNN requires a fixed-size input image, but normalization may be truncated/stretched the object, which may lead to loos of information. With a large number of region proposals, most of the region proposals may overlap with each other which makes R-CNN repeatedly extracted features from overlapping parts.

A Spatial Pyramid Pooling network (SPP-Nets) utilizes CNNs to improve the R-CNN by sharing the computation; SPP method canceled the crop/warp image normalization process to solve the information loss and storage problems caused by image distortion and use Spatial Pyramid Pooling (SPP) to replace the last pooled layer before the fully connected layer [7]. The method effectively solves the problem of duplicate calculation of convolution layers and speedup over 100 times than the R-CNN at the test time. SPP-Net also has a number of disadvantages: like R-CNN, SPP-Net also is a multi-stage pipeline during the training and features are needed to be written into the disk [4]. Thus, the Fast R-CNN was proposed to improve the disadvantages of R-CNN [4] and SPP-Nets which speeds up 213 times over the R-CNN at test time and achieves higher mAP (66%) on PASCAL VOC2012. The Fast R-CNN has several advantages, such as training in single-stage and using a multi-task loss; training can be updated into all network layers using the backpropagation through the ROI pooling layers. Unlike R-CNN, the Fast R-CNN maps the region proposal on the features directly so that image only needs to be extracted once, which significantly reduces the time-consuming.

In 2015, Ren et al. [19] proposed a Faster R-CNN method and achieved 73% mAP on PASCAL VOC2007. In the Faster R-CNN, Region Proposal Network (RPN) was pointed out to replace the selective search in Fast R-CNN which shares the convolutional features of the whole image. It also was trained end-to-end to generate region proposal and then uses Fast R-CNN for detection. A deep convolutional network was used to propose regions, and a Fast R-CNN detector utilizes the proposed regions [19]. RPN is to implement the extraction of candidate window by using 3×3 sliding window; each sliding window generates nine anchors and extracts features of these nine anchors for classification and bounding box regression. During the training, cross-boundary anchors are ignored, the highest or higher than 0.7 Intersection-over-Union (IoU) anchors overlap with ground truth box, and each ground truth box may correspond to more than one positive anchor.

Previous methods of human behaviour recognition repurpose classifiers to perform the recognition [17]. Thus, "you only look once" (YOLO) was proposed, YOLO solves the problems of object detection as a regression problem based on an end-to-end network; the output is completed from the input of original images to the position (bounding boxes coordinates) and class (class probabilities) of an object. The main feature of YOLO network is that YOLO can meet the real-time requirements which achieve 45 frames per second (FPS). It also uses the full image as the context information which reduces the false positive in the background. The YOLO algorithm is described as: first; it will divide the image into a fixed $\times$ grid. If the center of sample object falls on the corresponding grid, and the grid will respond to the regression of this object position; second, each grid prediction contains object position and its confidential information which is encoded as a vector; finally, the network output layer is the corresponding result for each grid. However, YOLO model also has some vulnerabilities such as utilizing sum-squared error to implement the loos function which makes convergence become worse and cannot align with maximizing average precision correctly; the loss function treats the different size of bounding boxs error as the same. Therefore, the errors of small bounding box will have a significant impact on the process of network optimization which will reduce the accuracy of the detection. Because each grid cell can only predict two boxes and limits with only one class, a group of small objects cannot predict very well.

YOLOv2 was proposed in 2016 [18], which improves the accuracy (achieved 76.8% mAP in 67 FPS and 78.6%

mAP in 40FPS). With batch normalization on all the convolutional layers, YOLOv2 achieves up to 2% improvement in mAP; a multiscale training method, YOLOv2 can adjust the input size according to the speed and precision requirements. YOLOv2 also is proposed, a joint training algorithm can train object detectors for both detection and classification.

In this paper, our work is different from the existing ones. Our contributions are to use data augmentation for the pre-processing, we adjust the YOLOv3 structure in our proposed model. Thus, it shows a positive result for human behaviour recognition.

3. Our Method

At present, there have a large number of public datasets available for pedestrian detection. In this study, we used the publicly available datasets which include Weizmann datasets, UT-Interaction datasets and the INRIA datasets for for human behaviour recognition. For the Weizmann datasets, each video clip contains only one participant. On the contrary, for the UT-Interaction datasets, each video clip contains multiple participants. The INRIA datasets have various viewing angels which are including front, back and various sides. In this project, we utilize three different datasets to implement our research. In this study, there were 1198 samples in total. Moreover, 1078 samples were utilized for training and 120 samples for the cross-validation. Moreover, we use data augmentation for the pre-processing to increase data samples, we flip the samples from horizontal and random zoom range set to 0.2.

In this paper, we implemented both YOLOv2 and mainly focused on YOLOv3-based object detection for human behaviour recognition in videos. In YOLOv3, it integrates advanced algorithm such as residual network and upsampling. Moreover, YOLOv3 improves the network depth into total 53 layers; it carries out predictions on three different sizes of feature maps. To be exact, it is to perform 32, 16 and 8 times downsampling on the input image before prediction; the network downsamples the input image until the first detection layer, which uses a feature map with a stride of 32 for detection. Then, the layers are sampled twice and connected to the feature map of the previous layer with the same feature map dimensions. Another detection is performed on the layer with stride 16, and the same upsampling process is repeated. The last detection was performed at the layer of stride 8. The bounding box for prediction is shown in eq.(1).

$$\begin{cases} b_x = \sigma(t_x) + c_x \\ b_y = \sigma(t_y) + c_y \\ b_w = p_w e^{t_w} \\ b_h = p_h e^{t_h} \end{cases} \tag{1}$$

where b_x,b_y,b_w,b_h represent the predicted central coordinates x,y,w (width),h(height) and t_x, t_y ,t_w, t_h stand for predicted coordinates. c_x and c_y are the top left corner of an image cell. p_w and p_h are the bounding box width and height, $\sigma(\cdot)$ is used to constrain its possible offset range. A Sigmoid function is used to predict the central coordinates which force the output values to be compressed between 0 and 1 to keep the center in the predicted grid unit. In our experiment, we adjust the deep neural network layers. For the traditional YOLOv3, it contains 53 layers in total. Moreover, we adjust the YOLOv3 layer by decreasing its convolutional layers to reduce the GPU process pressure. For our deep neural network, we only have 45 layers in total which shown on Fig. 1, the left side is the original YOLOv3 network, and the right side is the modified YOLOv3 network, we merely decrease half of the 8×layers to 4 × layers. In our experiment, we use the adjusted network to train our model and also utilized it to recognize the simple human behaviors such as walk, run, skip, jack and jump.

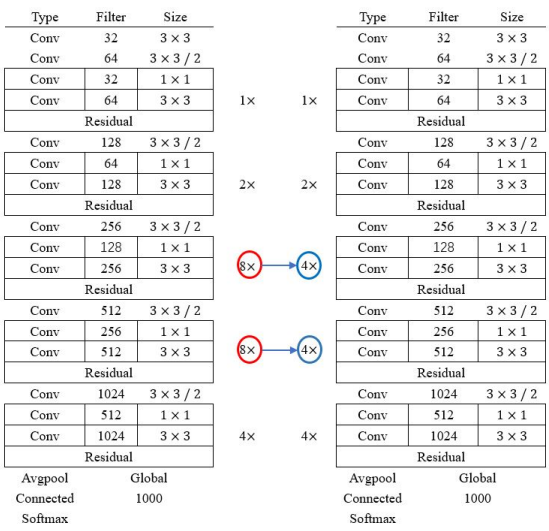


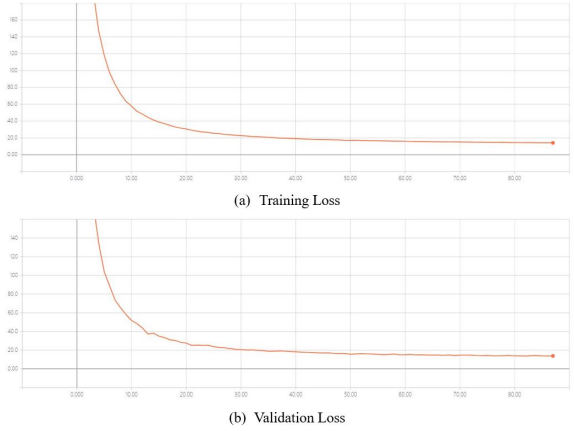
Figure 1. Modified YOLOv3 network

In this study, we set the batch equals 32 and subdivision is 8 for each iteration, we send 32 samples into the network and all these samples were divided into 8 times for network training. For YOLOv3, we set the learning rate as 0.0001. Also, the number of epochs is 70. In YOLOv2, it contains 19 convolutional layers and five max-pooling layers. We utilize data augmentation in this study to increase our training data. For our experiments, we use GPU (GTX970) acceleration to speed up our training process so as to reduce time consumption.

4. Results and Analysis

There are several standard evaluations in information retrieval which are accuracy, recall and precision. Fig. 2 is a result of human behavior recognition using YOLOv3 and

Figure 2. The result of human behavior recognition by using YOLOv3.

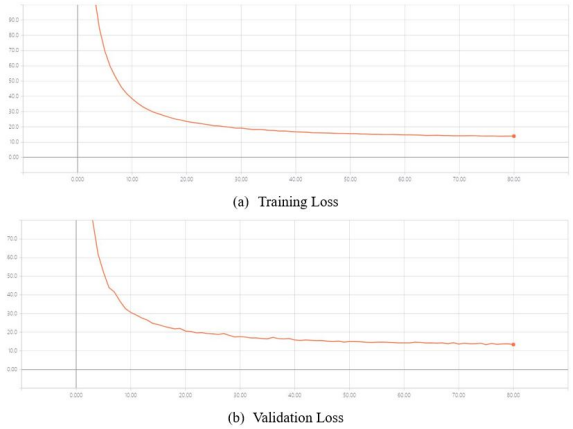


(a) Training Loss

(b) Validation Loss

Figure 3. The training and validation loss with 0.001 learning rate in YOLOv3.

generated based on the test video frames which includes (a) walk, (b) skip, (c) run, (d) jack and (e) jump. For all the experiments, our batch size was set to 32 and subdivision were set to the 8. The reason is that deep learning requires GPU acceleration, but increasing the batch size will consume more GPU memory. Fig. 3 shows the validation loss with 0.001 learning rate. In this experiment, YOLOv3 learning rate was set to 0.001; for the training loss is 14.11 and validation loss is 13.84. After 87 epochs, it stopped (early stopping) in case of accuracy reduction and avoid overfitting. Fig. 4 shows the loss with 0.0001 learning rate, the study was stopped at 82 epochs. Training loss dropped to 13.91 and validation loss dropped to 13.39. Moreover, 14.56 training loss was obtained if we set learning rate as 0.00001 and training stopped at 63 epochs. Table 1 shows the different hyperparameters which will affect the result in



(a) Training Loss

(b) Validation Loss

Figure 4. The training and validation loss with 0.0001 learning rate in YOLOv3.

Table 1. The results of Yolov3 with different learning rates

Learning Rates	1×10^{-3}	1×10^{-4}	1×10^{-5}
Training Loss	14.11	13.91	14.56
Validation Loss	13.84	13.39	14.41
Test Accuracy	73.75%	80.20%	70.00%
Number of Epochs	87	82	63

both training and testing stages. Increasing batch size will require more GPU consume.

After 87 epochs, it stopped (early stopping) in case of accuracy reduction and avoid overfitting. Fig. 4 shows the loss with 0.0001 learning rate, the study was stopped at 82 epochs. Training loss dropped to 13.91 and validation loss dropped to 13.39. Moreover, 14.56 training loss was obtained if we set learning rate as 0.00001 and training stopped at 63 epochs.

Table 1 shows that different hyperparameters which will affect the result in both training and testing stages. Increasing batch size will require more GPU consume. In our experiments, we only adjust the learning rate (LR). Therefore, decreasing the learning rate can obtain less loss, but the network may fall into a local optimum. On the contrary, if the learning rate gets too big, the loss will stop falling and repeatedly oscillate at some point. When we set the learning rate to 0.0001, training loss gets only 13.91, and validation loss comes to 13.39 which were the best result in these three different values of LR. Moreover, it obtains 80.20% of accuracy in test video frames. During the GPU acceleration, detection can be achieved in 15 fps.

We implemented YOLOv2 based on the Python3.6 with TensorFlow to training and testing, combined with MATLAB 2018 for the user interface, and data pre-processing including data augmentation. Fig. 5 shows the training loss for the YOLOv2. In this experiment, learning rate was not fixed, we adjust different learning rate for every ten iterations and learning rate mainly are 1e-3, 1e-4 and 1e-5.

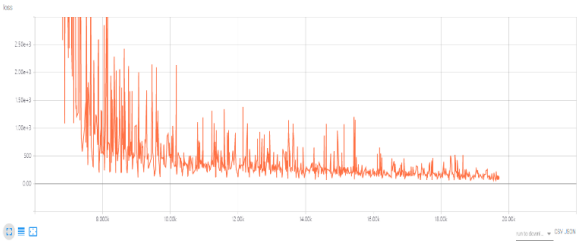


Figure 5. The training loss in YOLOv2.

Figure 6. The results of pedestrian detection by using YOLOv2.

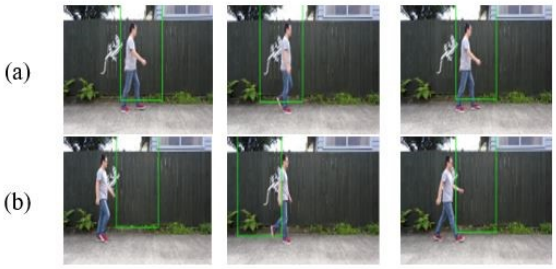


Figure 7. The test results by using YOLOv2.

Training stage contains 40 epochs; we also set the batch size to 32 in case of lack of GPU memory. Fig. 6 shows the YOLOv2 detection; three images were correctly detected; three images contain with one pedestrian but cannot be detected very well. Both images were coming from public INRIA dataset. Fig. 7. shows the result of detection by utilizing test video frames, (a) the frames were correctly recognized; (b) the frames were incorrectly recognized. Moreover, there is only 53% accuracy of the test video in 165 frames.

In the recognition experiment, we also use Weizmann dataset as the training set. Moreover, there are 1900 samples in total; the dataset consists with the training set (1710 samples) and validation set (190 samples). Moreover, there are five different behaviors shown in Fig. 8, which include walk, skip, run, jack and jump. Table 2 shows that the best precision of human behavior recognition is jack which

Figure 8. The results of human behavior recognition which are correctly classified.

Table 2. Classification precision for behavior recognition

Precisions	Walk	Skip	Run	Jack	Jump	mAP
YOLOv3	0.97	0.88	0.97	1.00	0.9	0.95
Modified YOLOv3	0.94	0.94	0.97	1.00	0.97	0.97

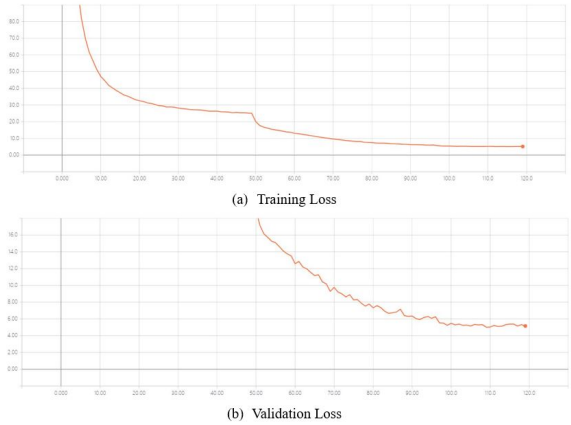
Table 3. Error rates of behavior recognition

Errors Rates	Walk	Skip	Run	Jack	Jump	mER
YOLOv3	0.129	0.167	0.029	0.0	0.114	0.088
Modified YOLOv3	0.108	0.108	0.091	0.0	0.029	0.067

achieved 100%. However, for skip, it achieved lower precision in our experiment. Moreover, we believe that the reason to cause this problem because of skip activity, the frames may be similar to another jump activity. Moreover, in our experiment, after modified the YOLOv3 structure, it also has a high precision rate for all activities, jack activity achieved the precision of 100%, and we acquired a decent result for the four behaviors, and the precision was increased. For both models, we have achieved really high mAP, the reason is that we only have five different classes for behavior recognition. Table 3 shows the error rate for our human behavior recognition; according to the Table 3, our modified YOLOv3 model has the error rate less than that of the YOLOv3 model in human behavior recognition. In our experiment, the error rate of jump activity only has 0.029. The jack activity has no error in both models. Fig. 9 shows the (a) training and (b) validation loss of our modified network. And in this experiment, the learning rate is set to 0.001 at the fine-tuning stage, learning rate will be automatically reduced in case of converge problem. Moreover, we pre-trained the network before 50 epochs; after fine-tuning the network, we see that training loss is able to achieve 5.14; validation loss is able to achieve 5.15 at 121 epochs.

5. Conclusion and Future Work

In this paper, we demonstrated a deep learning-based detection technique to achieve the pedestrian detection. The YOLO model as a deep learning method was implemented in this study; it can achieve a real-time detection. However, a GPU acceleration during the training and testing by using deep learning is required to reduce the time-consuming.



(a) Training Loss

(b) Validation Loss

Figure 9. The training and validation loss by adjusting the network.

Moreover, with the different hyperparameters which can affect the results; thus, a proper hyperparameter should be chosen carefully to fine-tune the model. Future work should be to extend the YOLO detection method. Deep learning shows the abilities to detect objects and classify each object into the right class. Thus, we will implement this to get more advanced which not only to detect pedestrians but also to achieve higher performance of human behaviors recognition by using deep learning methods.

References

[1] P. Agrawal, R. Girshick, and J. Malik. Analyzing the performance of multilayer neural networks for object recognition. In *ECCV*, pages 329–344, 2014.

[2] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE CVPR*, volume 1, page 886893, 2005.

[3] D. Gernimo, A. Lpez, D. Ponsa, and A. D. Sappa. Haar wavelets and edge orientation histograms for onboard pedestrian detection. In *Iberian Conference on Pattern Recognition and Image Analysis*, volume 2, pages 418–425, 2007.

[4] R. Girshick. Fast R-CNN. *IEEE ICCV*, pages 1440–1448, 2015.

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE CVPR*, pages 580–587, 2014.

[6] R. Girshick, F. Iandola, T. Darrell, and J. Malik. Deformable part models are convolutional neural networks. In *IEEE CVPR*, pages 437–446, 2015.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on PAMI*, pages 1904–1916, 2015.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[9] G. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[10] S. Ji, W. Xu, M. Yang, and K. Yu. 3D convolutional neural networks for human action recognition. *IEEE Transactions on PAMI*, 35:221 – 231, 2014.

[11] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541 – 551, 1989.

[12] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278 – 2324, 1998.

[13] Y. LeCun, F. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *IEEE CVPR*, 2004.

[14] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *ECCV*, pages 21–37, 2016.

[15] J. Lu, J. Shen, W. Yan, and B. Bacic. An empirical study for human behavior analysis. *IJDCF*, 9(3):11–27, 2017.

[16] A. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: An astounding baseline for recognition. In *IEEE CVPR Workshop*, pages 806–813, 2014.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. *IEEE CVPR*, 2016.

[18] J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. *IEEE CVPR*, page 65176525, 2017.

[19] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on PAMI*, 39(6):1137 – 1149, 2017.

[20] K. Sande, J. Uijlings, T. Gevers, and A. Smeulders. Segmentation as selective search for object recognition. In *IEEE ICCV*, pages 1879–1886, 2011.

[21] C. Szegedy, A. Toshev, and D. Erhan. Deep neural networks for object detection. In *International Conference on Neural Information Processing Systems*, pages 2553–2561, 2013.

[22] J. Uijlings, K. Sande, T. Gevers, and A. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104:154171, 2013.

[23] B. Wu and R. Nevatia. Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In *IEEE ICCV*, volume 1, pages 90–97, 2005.

[24] W. Yan. Introduction to intelligent surveillance: Surveillance data capture, transmission, and analytics (Springer: London), UK, 2017.

[25] K. Yu, W. Xu, and Y. Gong. Deep learning with kernel regularization for visual recognition. In *Advances in Neural Information Processing Systems*, pages 1889–1896, 2009.

[26] Y. Zheng, C. Shen, R. Hartley, and X. Huang. Pyramid center-symmetric local binary/trinary patterns for effective pedestrian detection. In *ACCV*, volume 1, pages 281–292, 2010.