Mini Project
Report

Study of Pedestrian Detection Using YOLOv2.

Submitted in partial fulfillment of
the requirements for the award of the degree of

Bachelor of Technology
in
Electronics and Telecommunication Engineering

Submitted by

Roll No	Names of Students
111087021	Sumit Popat Lengare
111807066	Parth Praveen Shastri

Under the guidance of
Dr. Aditya Gupta

Abstract

The task of object detection is well studied and used in the fields of driving of Autonomous Vehicles and self-driving cars, but the task of person detection in more emphasized and paid attention upon as the pedestrians on the streets are of a higher priority.In recent years, the development of deep learning based object detection models have seen a great improvement in terms of accuracy and speed, but not both at a time. There has always been a trade off when considered a accurate model and a quick model. But in recent years the team of YOLO and its improvements have achieved extraordinary speed and an acceptable precision, keeping this in mind we try to use this and further reduce this gap. From our end, we employ the YOLOv2 model, a previous state of the art algorithm in the object detection field, in pedestrian detection. We modify some network parameters and try to improve its score in this domain and hence making it work suitably on this task. Experimental results on the INRIA dataset show the improvement in result as compared to vanilla training after the modification the loss function and also employ label smoothing and training on negative examples.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background and Recent Research

The task of pedestrian detection is considered to a subset of the task of general object detection. The object detection task itself is very popular and looked upon in the field of Computer Vision. The applications of pedestrian detection further extend to many useful day-to-day tasks suck as surveillance systems, auto pilot systems, automotive-autonomous driving, driving assistance systems, intelligent transport systems. Despite the great improvement in accuracy in these past few years, the task of pedestrian detection still needs meticulous optimization and scene specific detection. Over the past few years pedestrian detection systems have adopted a variety of measures. Some of these methods focus on the speed of detection yolo[13], [23], while some of these methods are more inclined towards the accuracy of the detection [22], [11]. In the wake of rapid development in software and hardware, there began a wave of deep learning. The discovery of Convolutional Neural Networks has led to an revolution in the computer vision tasks, CNNS are state of the art in many computer vision tasks till date. The deep learning methods are seeing exponential improvement along the path.

When comes to Deep learning many methods have a lot in common, in terms of their pipelines. For most of the deep detection frameworks, they proceed in two phases. In the first phase the high level features are extracted from the pixels of the input image and then using a suitable algorithm the features are segmented according to what we call regions-of-interest. In the second stage, these are then fed to a deep network one by one and low-level features are extracted and then this is treated as a typical classification or a regression task. In this paper we are focused on YOLO which was introduced as a single-shot pipeline rather than the many-staged systems.

1.1.1 Previous Work

In the past few decades the development in research and technology has led to exponential improvement in detection systems, in their accuracy, in their speed. At first feature extraction models like HOG[4] were used, but now a days the deep learning approach is common.But it is safe to say that the discovery of CNNs has been a pivotal point in the detection domain. Since the deep learning entered the field of research,pedestrian detection has been greatly improved in its accuracy. Nevertheless, their running time has been a bit slower, approximately a few seconds every image or even more slowly. In addition, there are several impressive methods employed in the deep network

We can divide the journey of deep learning in object detection in two parts i.e. Multi-shot detectors and Single-shot detectors.

Multi-shot models

Until the introduction of SSD, multi-shot models were the go-to models in object-detection. In these models two steps were observed first was extracting the regions of interest using some traditional algorithms like selective search, and then passing a Convnet detector to extract more features and output the desired probabilities and co-ordinates of the bounding boxes. But before these models a method known as sliding window detector was widely used, the way in which the method worked (as its name, it slided a CNN over the whole image) was computationally inefficient and took a lot of time. So we can say multi-shot models starting from RCNN were an improvement to these models. Although better RCNNs faced similar problems in terms of computation and speed. It extracted $\sim$ 2000 regions of interest which made it high in computation in the next step.

The solution to this problem was addressed by Fast RCNN, which introduced a good method known as ROI pooling, which was a special case of SPP(Spatial Pyramid Pooling) to increase efficiency. To further improve this Faster RCNN was proposed which introduced RPN(Region Proposal Network) which is kind of similar to single-shot detectors but still requires more time to compute. Although slow these models are extremely precise which made them popular to use.

Single-shot models

To address this issue of speed Redmon et al 2015 proposed a new and unified algorithm which he called as YOLO - You only look once. This method performed well on the Pascal VOC dataset and started to gain popularity among the researchers, instead of using multiple steps like in RCNN, it used a

single shot system which was unified making it fast. In high level it divided the input image into grid cells where each cell was responsible to predict two bounding boxes. He also proposed a new architecture which worked as a backbone feature extractor *Darknet*. This backbone network did all the feature extraction from the images and final output was a feature map representing the grid cells. But as fast as it was YOLO was less accurate compared to its competitors which raised eyebrows.

So a new detector YOLOv2, simply YOLO version two was introduced, which according to us was a revolutionary detector in the fields of object detection. In this method, the concept of anchor boxes like in Faster RCNN was used to make the model more stable and a new feature extractor was proposed, still it was less accurate in comparision to its counterparts but still it fared well in the trade off.

1.2 Motivation

In the application of deep learning, the structure of the model and its parameters play a pivotal role in getting good results in terms of accuracy. Here we are trying to build on the work of Redmon et al. and carefully and attentively study their work in the field and apply this to pedestrian detection, trying to minimize the trade off between speed and accuracy. We analyze and revise their model thoroughly trying to tweak and improve the results. The field of pedestrian detection is emphasized upon and will help in solving many real world problems.

Chapter 2

Work Done

In this section we will introduce the baseline model in detail and also the improvements done to make it suitable for pedestrian detection.

2.1 Base detector

YOLOv2, an improved version of YOLO , is a detection model with the superior performance applied to the general detection tasks. YOLOv2 could run at the different sizes employing a vanilla as well as the multiscale training technique. Meanwhile it could offer a rather good trade-off between speed and accuracy, being able to outperform advanced techniques like Faster R-CNN, SSD and so on but still run faster than those all. The YOLOv2 network integrates the extraction of the candidate boxes, the feature extraction,the target classification, and the target location into a single deep network, as we have seen in the previous section. That enables end-to-end training and improves the errors in the YOLO model. For achieving an efficient and accurate pedestrian detection, we introduce the general detector, YOLOv2, as the baseline network of our pedestrian detection model, and then make some modifications in the loss function and the parameters of the network, adapting better for pedestrian detection.

2.1.1 Anchor box clustering

The YOLOv2 model adapts to the anchor box method used in the Faster -RCNN model which enables it to predict an offset to selected anchor boxes of different aspect ratios, rather than predict the coordinates in real numbers making the gradients of the model unstable as in YOLO. This improves the results to some extent. The original paper proposed running a clustering

algorithm on all the ground truth boxes in the dataset and selecting the cluster centres as the anchor boxes. We use the famous K-means algorithm as used in the original paper with the distance given by,

$$d(box, centroid) = 1 - IOU(box, centroid)$$

. Here IOU is the IOU metric(Intersection over union). We select the optimal K by looking at graph of number of K's and the mean distance and selecting the elbow point of the graph. Refer figure 2.1.

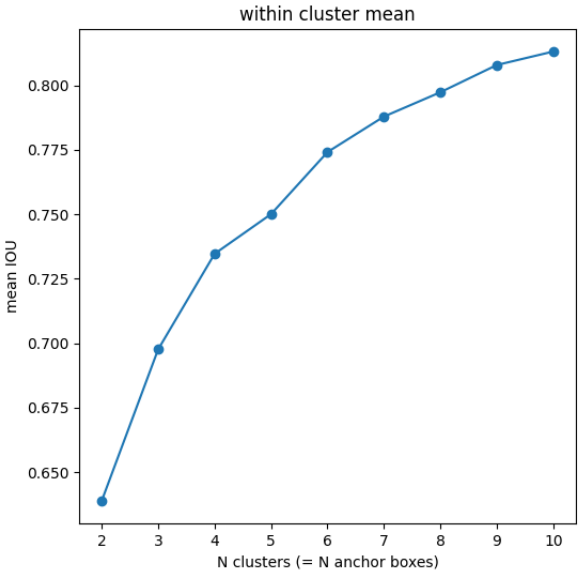

Figure 2.1: We select the elbow point in the graph i.e. $K = 6$ for our model

However, we can select $K > 6$ for improved results at the cost of model complexity.

2.1.2 Detection Algorithm

The YOLOv2 divides the input image into MxN grids, Refer figure 3.1, in our case it is $M = N = 13$, each grid cell will detect an object if the center of that object falls in the cell. Every grid cell will be given B anchor boxes initially. Each ground truth box is associated with a single anchor box in the labels. This is done using the IOU metric (Intersection over union) between each ground truth and anchor. The model will predict offset to this anchor after training. Each box B is associated with a confidence score or objectness

score which determines whether a object is present in that B or not. It is given by,

$$Conf(Object) = P(Object) * IOU(truth, pred)$$

Here $P(Object)$ is simply given as the probability of object in the cell. It is $= 1$ if there is an object in cell and $= 0$ if no object in cell. We scale the confidence by IOU as we do not want our model to be overconfident in predicting the bounding boxes, as YOLO has significant amount of localisation errors.

The output for YOLOv2 is encoded as a $(M \times N \times (B * 5 + 1))$ tensor here the 5 represents the coordinates in $(Conf(Obj), x, y, w, h)$ this format and the 1 represents the probability of the Class

$$Conf(Person) = P(Person|Object)$$

. The output feature map of the feature extractor will be of the given shape. Once the coordinates and other outputs are obtained from the network we employ NMS(Non-Maximal Suppression) which is used to remove the redundancies in the final output.Refer figure 2.2.

2.1.3 The Network Architecture

The architecture of YOLOv2 is shown in 3.2, the design idea of this model is similar to the RPN(Region Proposal Network) in the Faster RCNN. This network removes the fully connected layer in RPN and replaces it with the convolutional layer to predict bounding box co-ordinates and the confidence scores. The network is called *Darknet-19* in the original paper, it is a 24-layered CNN which achieves 72.9% top-1 accuracy and 91.2% top-5 accuracy on ImageNet. This network has 19-Convolutional layers along with 5-maxpool layers, it also uses Batch Normalisation in between. The Darknet-19 backbone architecture is given in Figure 2.5

Our Network takes an image of 416×416 and the corresponding grid cells are 13×13.

2.1.4 The loss function

The original YOLOv2 loss is given same as YOLO, our loss looks like,

$$L = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \Bigg(\lambda \mathbb{1}_i j^{\text{obj}} \big((x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \big) + \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \Bigg) \tag{2.1}$$

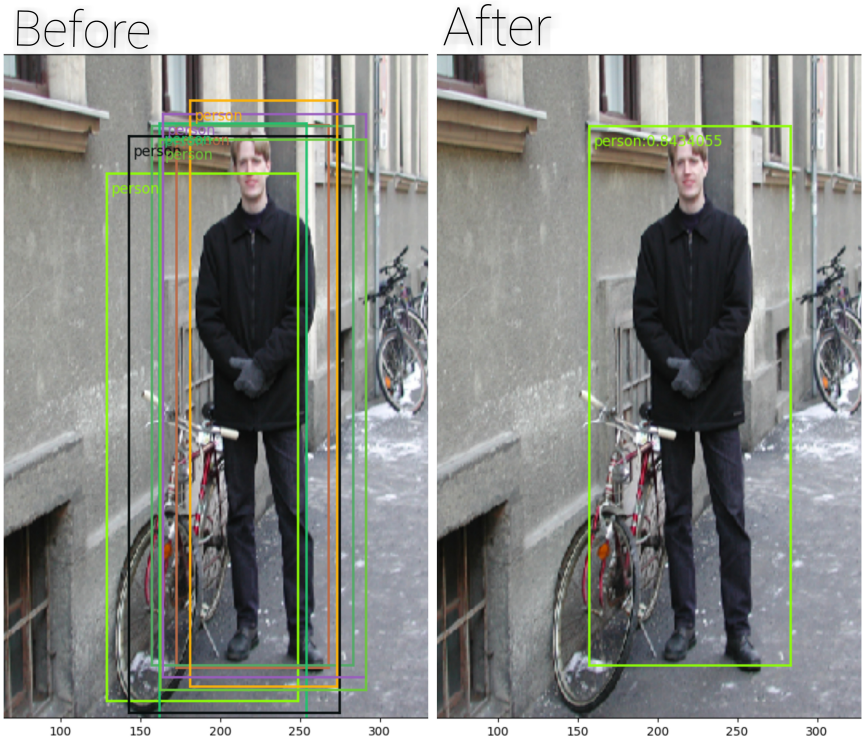

Figure 2.2: The NMS algorithm (Before and After).

Here normal squared-error is used for the coordinate loss as well as the class loss, it is a easy to learn convex function but in our model we use the Binary Cross-entropy loss function as in YOLOv3 also called as the logistic loss. This is used in Binary classification problems normally but we use it here to help the model generalise better. The loss contains only those cells in which the object is present.

Therefore, we give the objectness loss as,

$$\sum_{i=0}^{S^2}\sum_{j=0}^{B} 1_{ij}^{obj} l(C_i, \hat{C}_i) + \lambda_{noobj} \sum_{i=0}^{S^2}\sum_{j=0}^{B} 1_{ij}^{noobj} l(C_i, \hat{C}_i)$$

Figure 2.3: objectness loss

Here $l(C_i, \hat{C}_i)$ represents the Binary crossentropy between the predicted

Confidence C_i, and the true Confidence $\hat{C}_i$, and $\mathbb{1}_i j^{\text{noobj}} = 1 - \mathbb{1}_i j^{\text{obj}}$.

The final loss is given by

$$Loss = L + L^{\text{obj}} \tag{2.2}$$

We also employ a technique known as label smoothing in the class loss to prevent our model from becoming overconfident. It is the same as we have seen in the case of $Conf(Object)$ in the previous section, only instead of multiplying by IOU we can simply decrease the target label from 1 to a lesser value. Here we keep the detection threshold for the confidence to 0.6, and keep the $\lambda noobj$ value to 0.5 and λ value to 5 to focus more on the areas where object is present else the gradient due to the loss of No-Objects will be more. The confidence threshold can be tweaked according to the task at hand.

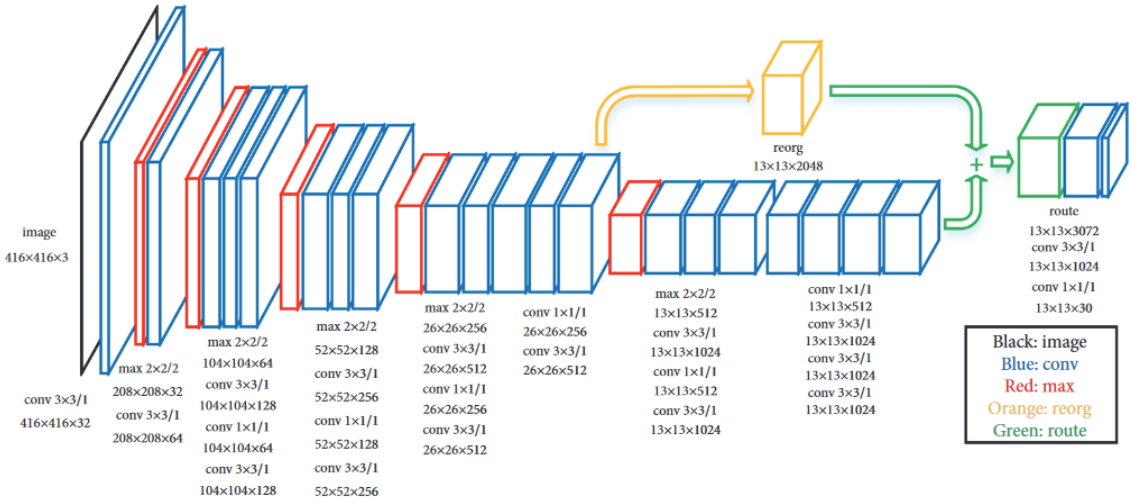

Figure 2.4: The YOLOv2 Architecture.

2.1.5 Dataset

INRIA Pedestrian Dataset.To perform the following experiments, we recourse to the INRIA Pedestrian Dataset, a commonly accepted, multiscales dataset with a certain challenge which is often used to evaluate the performance of the pedestrian detection techniques. The INRIA Pedestrian Dataset is created in the research work for detecting the erect pedestrian in images and videos. It is subdivided into two patterns:(1)raw images with the appropriate annotations and(2)positive images normalized into 64x128pixel with

Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	128	1×1	28×28
Convolutional	256	3×3	28×28
Maxpool		$2 \times 2/2$	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Maxpool		$2 \times 2/2$	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	1000	1×1	7×7
Avgpool		Global	1000
Softmax			

Figure 2.5: The Darknet-19 architechture.

the raw negative images. We employ the train set and the test set to train and validate our models, respectively,which are contained in the raw images with the appropriate annotations. In this dataset, only the upright persons whose height are greater than 100 are signed in per image. However,the annotation may be incorrect. Sometimes the part of the bounding box labeled can be inside or outside the object,whose influence can be ignored.The INRIA Pedestrian Dataset contains a train set and a test set. The train set has 614 positive images, with 1237 pedestrians. While the test set has 228 positive images, with 589 pedestrians. Images in dataset have the complex background with an obvious light change. The pedestrians, with different degrees of occlusion, wearing different costumes, have many kinds of scales and changing postures.

2.2 Experimental Evaluation

We conduct an experimental campaign to evaluate the performance of our detection model. We train our model on an Intel corei7 8^th gen CPU. for the CNN computations we use the NVIDIA MX250 GPU, which is mediocre in

Figure 2.6: YOLO divides image into 13×13 grid in our case.

its limits and all the results are achieved by training the models for < 100 epochs on the INRIA dataset as we have mentioned above.

We used Gradient Descent algorithm along with the Adam optimizer with initial learning rate of $3 \times 10^{-}4$, with a piecewise decay of 1/3, and 5/3 after every 25 epochs or 1.25k iterations. We also used extensive data augmentation like random flipping, random cropm of about 90% of the image, random hue , random saturation 1.5×, color jitter etc. The default image size that we trained on is 416×416. We did not use Multi-scale training due to lack of resources.

We train 3 types of models: (1)the baseline YOLOv2 (2)the YOLOv2 model based on the Binary crossentropy loss as in YOLOv3 (3)the model in (2) but using label smoothing and also using negative examples in the INRIA dataset. We train on the negative examples to show our model the background scenes where there are no objects so that it will reduce the False Positive Boxes i.e. The predicted boxes in which there are no objects present.We test our models on both standard Non max-suppression as well as a new technique called as soft NMS, both techniques are used for the removal of redundancies but the latter favours dense detection more by scaling the scores of the redundant boxes rather than eradicating them, so if pedestrians are present close together our model can robustly detect them.

The evaluation metric used is Mean-Average-precision a widely used metric for object-detection tasks. The results observed are in tabular form in 2.1

The Models	mAP-50 (std. NMS)	mAP-50 (soft NMS)	FPPI
YOLOv2(baseline)	77.1%	78.2%	1.11
YOLOv2+BCE loss	80.1%	82.6%	1.6
YOLOv2+BCE+lbl sm (proposed)	82.3%	85%	0.792

Table 2.1: The results are evaluated using mAP-50 for both standard and soft NMS.FPPI denotes the false positives per image.

Chapter 3

Conclusion

After seeing the results we can say that the BCE loss plays an important role in the increase of accuracy, But as you can see in 3.1, the problem of False positives is unresolved, this improves significantly with the inclusion of negative examples along with label smoothing. The mAP also improves by about $+1.5 - 2.0\%$ and by a whooping 7.0% as compared to YOLOv2 baseline model. The soft NMS also helps in improving the accuracy if any dense objects are present. As in our dataset mostly the pedestrians are not in crowded areas this is not as significant. But in many real situations it can help in model accuracy. Finally we can conclude by saying that, addition of negative examples and label smoothing improves the model localization capability increasing the accuracy further and improving our model.

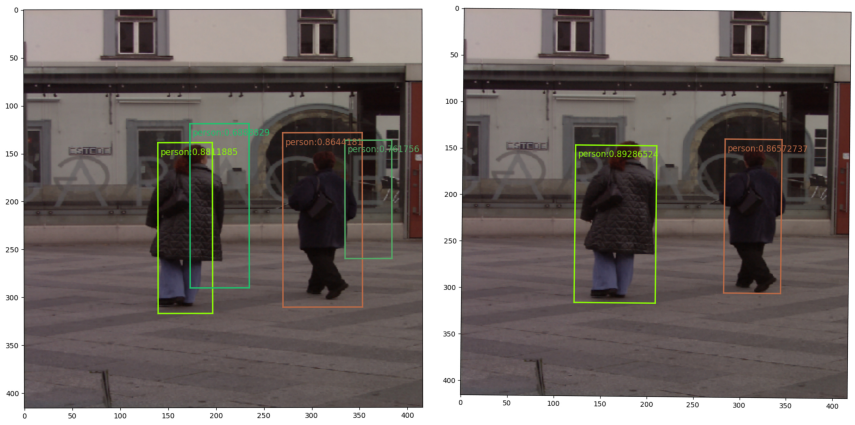

Figure 3.1: The image on the left shows the result of YOLOv2 + BCE, and on the right shows the result of our model.$thresh_{\text{obj}} = 0.6$ in both cases.

Figure 3.2: **Results** on the test set. In the first image the model localizes well.In the second image there is a false positive due to lack of dense images to train on.

Chapter 4

Future Work

This study has focused on improving the task of pedestrian detection using previous datasets. Another approach could be creating our own dataset with high resolution images and properly annotating them, to ensure no localization errors. We have seen the improvement caused due to the inclusion of negative examples for task specific images we had included a portion of negative images, but we can test by including equal amounts of positive and negative images as it will help in reducing the false positives by a significant margin. We can also use Adaptive NMS which adapts according to the density of the image in hand, but this includes a density prediction network within the model hence increasing its complexity, we can work around it to find an optimal solution.

We can also deploy high resolution surveillance systems using this model as its real time capability is noticeable.We can deploy this model using raspberry pie or any other portable computation systems. we can also focus on extending and trying to deploy this on autonomous vehicles or robots.Furthermore we can try and extend this thought on building a robust and efficient automated system.

References

[1] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster,Stronger," in-Proceedings of the 30th IEEE Conference on Com-puter Vision and Pattern Recognition, (CVPR '17),pp.6517–6525,Honolulu, Hawaii, USA, 2017.

[2] R. Benenson, M. Omran, J. Hosang, and B. Schiele, "Ten yearsof pedestrian detection, what have we learned?" inProceedingsof the European Conference on Computer Vision (ECCV '14),vol.8926 ofLecture Notes in Computer Science, pp. 613–627, 2014.,

[3] A. D. Costea and S. Nedevschi, "Word channelbased multiscalepedestrian detection without image resizing and using onlyone classifier," in-Proceedings of the 27th IEEE Conference onComputer Vision and Pattern Recognition, (CVPR 2014),pp.2393–2400, USA, June 2014.,

[4] N. Dalal and B. Triggs, "Histograms of Oriented Gradients forHuman Detection," inProceedings of the 2005 IEEE ComputerSociety Conference on Computer Vision and Pattern Recognition(CVPR '05),pp.886–893,SanDiego,CA,USA,2005.,

[5] R. Appel and W. Kienzle, "Crosstalk cascades for frame-ratepedestrian detection," inProceedings of the European Conferenceon Computer Vision (ECCV '12),pp.645–659,2013.,

[6] J. Yan, X. Zhang, Z. Lei, S. Liao, and S. Z. Li, "Robust multiresolution pedestrian detection in traffic scenes," inProceedingsof the 2013 IEEE Conference on Computer Vision and PatternRecognition,pp.3033–3040,2013.,

[7] W. Ouyang and X. Wang, "Joint Deep Learning for PedestrianDetection,"inProceedings of the 2013 IEEE International Confer-ence on Computer Vision, pp. 2056–2063, IEEE, Sydney, NSW,Australia, 2014.,

[8] P. Doll éar, S. Belongie, and P. Perona, "The fastest pedestriandetector in the west," inProceedings of the British Machine VisionConference, pp. 1–11, BMVA Press, 2010.[15] X. Zeng, W. Ouyang, and X. Wang, "Multi-stage contextualdeep learning for pedestrian detection," inProceedings of the2013 IEEE International Conference on Computer Vision,pp.121–128, IEEE, Sydney, NSW, Australia, 2013.,

[9] C. Wojek, S. Walk, and B. Schiele, "Multi-cue onboard pedes-trian detection," inProceedings of the 2009 IEEE Conference onComputer Vision and Pattern Recognition, pp. 794–801, 2009.,

[10] L. Zhang, L. Lin, X. Liang, and K. He, "Is faster R-CNN doingwell for pedestrian detection?" inProceedings of the EuropeanConference on Computer Vision,pp.443–457,2016.,

[11] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towardsreal-time object detection with region proposal networks,"IEEETransactions on Pattern Analysis and Machine Intelligence,vol.39,no. 6, p. 1137, 2017.,

[12] X.Du,M.El-Khamy,J.Lee,andL.S.Davis,"FusedDNN:adeepneural network fusion approach to fast and robust pedestriandetection," inProceedings of the 2017 IEEE Winter Conferenceon Applications of Computer Vision (WACV), pp. 953–961, 2016.,

[13] J.Redmon,S.Divvala,R.Girshick,andA.Farhadi,"You only look once: unified, real-time object detection," in Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779–788, IEEE, Las Vegas, NV, USA,2016.,

[14] T.Kong,A.Yao,Y.Chen,andF.Sun,"HyperNet:towardsac-curate region proposal generation and joint object detection,"inProceedings of the 2016 IEEE Conference on Computer Visionand Pattern Recognition (CVPR), pp. 845–853, IEEE, Las Vegas,NV, USA, 2016.,

[15] S. Xie and Z. Tu, "Holistically-Nested Edge Detection," inPro- ceedings of the 2015 IEEE International Conference on Computer Vision.,

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database.InComputer Vision and Pattern Recognition, 2009. CVPR2009. IEEE Conference on, pages 248–255. IEEE, 2009,

[17] S. Ioffe and C. Szegedy. Batch normalization: Acceleratingdeep network training by reducing internal covariate shift.arXiv preprint arXiv:1502.03167, 2015,

[18] J. Redmon. Darknet: Open source neural networks in c. <http://pjreddie.com/darknet/¿, 2013–2016.

[19] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: To-wards real-time object detection with region proposal net-works. arXiv preprint arXiv:1506.01497, 2015

[20] Navaneeth Bodla, Bharat Singh, Rama Chellappa, andLarry S Davis. Soft-nms: improving object detection withone line of code. InICCV, 2017.

[21] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid poolingin deep convolutional networks for visual recognition. InECCV, 2014.

[22] R. Girshick, "Fast R-CNN," inIEEE International Conference onComputer Vision (ICCV), 2015.

[23] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. E. Reed.SSD: single shot multibox detector.CoRR, abs/1512.02325,2015.