Spatial Attention for Pedestrian Detection

Ujjwal[1,2], Aziz Dziri[2], Bertrand Leroy[2], and François Bremond[1]

[1]INRIA Sophia Antipolis, 2004 Route des Lucioles, 06902, Valbonne, France
[2]Institut VEDECOM, 23 bis Allée des Marronniers, 78000 Versailles

[1]

Abstract

Achieving high detection accuracy and high inference speed is important for a pedestrian detection system in self-driving applications. There exists a trade-off between detection accuracy and inference speed in modern convolutional object detectors. In this paper, we propose a novel pedestrian detection system, which leverages spatial attention and a two-level cascade of classification and bounding box regression to balance the trade-off. Our proposed spatial attention module reduces the search space for pedestrians by selecting a small set of anchor boxes for further processing. Furthermore, we present a two-level cascade of bounding box classification and regression and demonstrate its effectiveness for improved accuracy. We demonstrate the performance of our system on 2 public datasets – caltech-reasonable and citypersons; with state-of-art performance. Our ablation studies confirm the usefulness of our spatial attention and cascade modules.

1. Introduction

Detection of pedestrians is fundamental to several important applications such as *video surveillance* and *autonomous driving*. A good detector for these applications is the one providing high accuracy while maintaining high inference speed. There is usually an *accuracy-vs.-speed* trade-off in modern convolutional object detectors [10]. For example, Faster-RCNN [14] achieves better performance at the expense of inference speed, while SSD [13] does the opposite [10]. Pedestrian detection can be seen as a specific instance of the problem of *general-category* object detection and most contemporary pedestrian detectors are derived from Faster-RCNN [16, 2] or SSD. Hence, the *speed vs. accuracy* trade-off extends to modern pedestrian detectors as well. Is it possible to achieve the best of both worlds – high performance and high inference speed ? In this paper we describe a pedestrian detection system which achieves it.

We introduce our proposed pedestrian detector by outlining our major contributions alongwith observations guiding them.

1. **Observation 1**
   It is vital to limit the search space for pedestrians in an image for faster inference. Pedestrians cover a small portion of the image, compared to non-pedestrian background entities. A search space covering all potential pedestrian locations alongwith locations bearing close resemblance to pedestrians (e.g:- vertical structures [17]) is desirable. During training, this provides most relevant samples (*positives* and *hard negatives*) to a classifier for learning. Faster-RCNN as against SSD, uses a region proposal network (RPN) for this purpose. On the other hand, SSD does not limit the search space and rather performs classification and regression across all anchor boxes tiling the image space.
   Contribution 1
   We propose a spatial attention based alternative to RPN for limiting the search space. Our proposed spatial attention mechanism has following major features :
   - Unlike RPN, our spatial attention mechanism does not perform classification and regression of anchor boxes. It is a lightweight module aimed at selecting a small set of anchor boxes. These anchor boxes are then further processed by our detection head. A brief comparison of our anchor handling approach is compared with those of Faster-RCNN and SSD in figure 1.
   - The aforementioned approach mitigates the danger of missing a pedestrian early in the pipeline (e.g:- *a RPN misclassifying a pedestrian as a negative proposal*). In addition, it eliminates a large portion of the image (*consisting of non-pedestrian background entities*) from the search

[1]978-1-5386-9294-3/18/$31.00

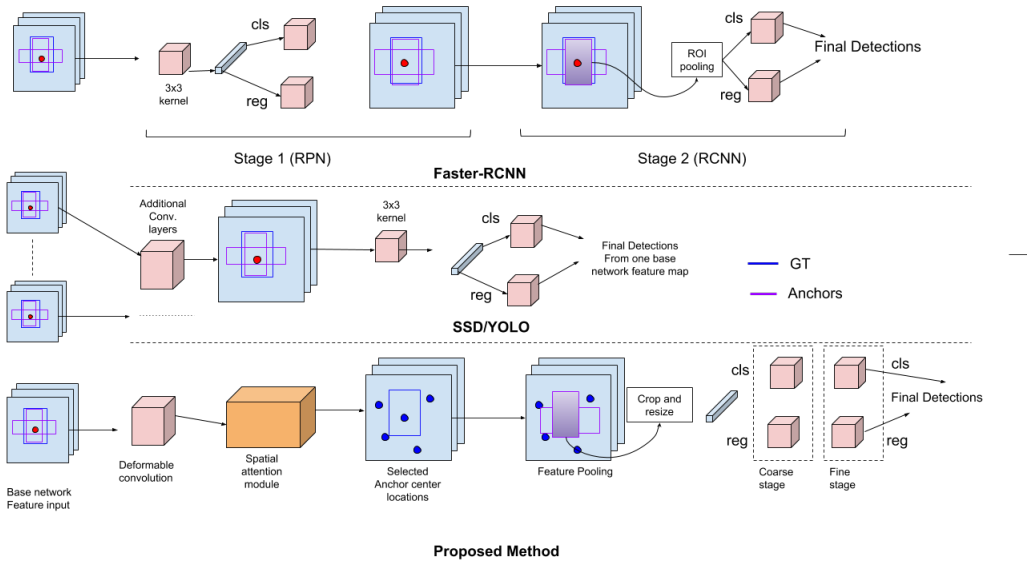

Figure 1. Anchor handling by various object detection techniques compared with our approach based on spatial attention.

space, thereby reducing the risk of false positives.

We describe the spatial attention mechanism in section 2.5. The difference of our method in handling anchors is summarized in figure 1. In Faster-RCNN, at the first stage, a convolutional kernel slides over a feature map and at each location classifies and regresses all the co-located anchors. In the second stage, features are pooled from the regressed anchors and further classified and regressed to get final detections. In SSD and YOLO, anchors are processed in a similar way to the first stage of Faster-RCNN. The process is however carried out individually for several feature maps of the base network. In our work the spatial attention module is used to obtain a set of candidate locations at which all anchors are processed. This reduces the number of anchors to be processed which leads to a reduction in the number of operations during the forward pass.

2. **Observation 2**
   Faster-RCNN pools features from proposal boxes (*in the second stage*), while SSD slides a convolutional kernel over a feature map for final classification and regression. Hence, final classifier and regressor in Faster-RCNN have access to box-specific features covering the whole extent of a proposal object. SSD accesses features in a fixed window at each location of the feature map and hence never accesses the full extent of features of an object (*unless an anchor is smaller than the kernel size*).

   On a flipside of the argument, Faster-RCNN never accesses the surrounding information of the proposal box, while SSD does access the surrounding information due to its sliding kernel.

   Contribution 2
   We propose utilizing deformable convolutional layers [6] along with feature pooling from anchor boxes to access the surrounding information of an object as well as utilizing the anchor-specific features. The deformable convolutional layer feeds the input to our spatial attention. The offsets computed for the deformable layer [6], ensure that features are pooled from non-uniform locations (determined by the offsets) in the input feature map. Thus the output features map (*input to the spatial attention mechanism*) contains features which are adapted to the overall scene structure. Our usage of deformable convolution is explained in section 2.4

3. **Observation 3**
   Use of feature maps from multiple layers of a CNN captures a richer feature diversity than one specific layer. Lower CNN layers are better at detecting small-scale and occluded pedestrians while latter layers better capture large-scale unoccluded instances[3]. While works such as MS-CNN [3], SSD [13] and FPN [12] utilize multiple layers, their approach counts as a late-fusion approach. Objects are detected separately for each selected CNN layer. This causes repetition of detection heads and an increase in training parameters.

   Contribution 3
   We favor an early-fusion approach, where feature maps are combined prior to processing by a single detection head. Owing to pooling operations in CNN architectures, feature maps from multiple layers usually have different sizes. We use àtrous convolution

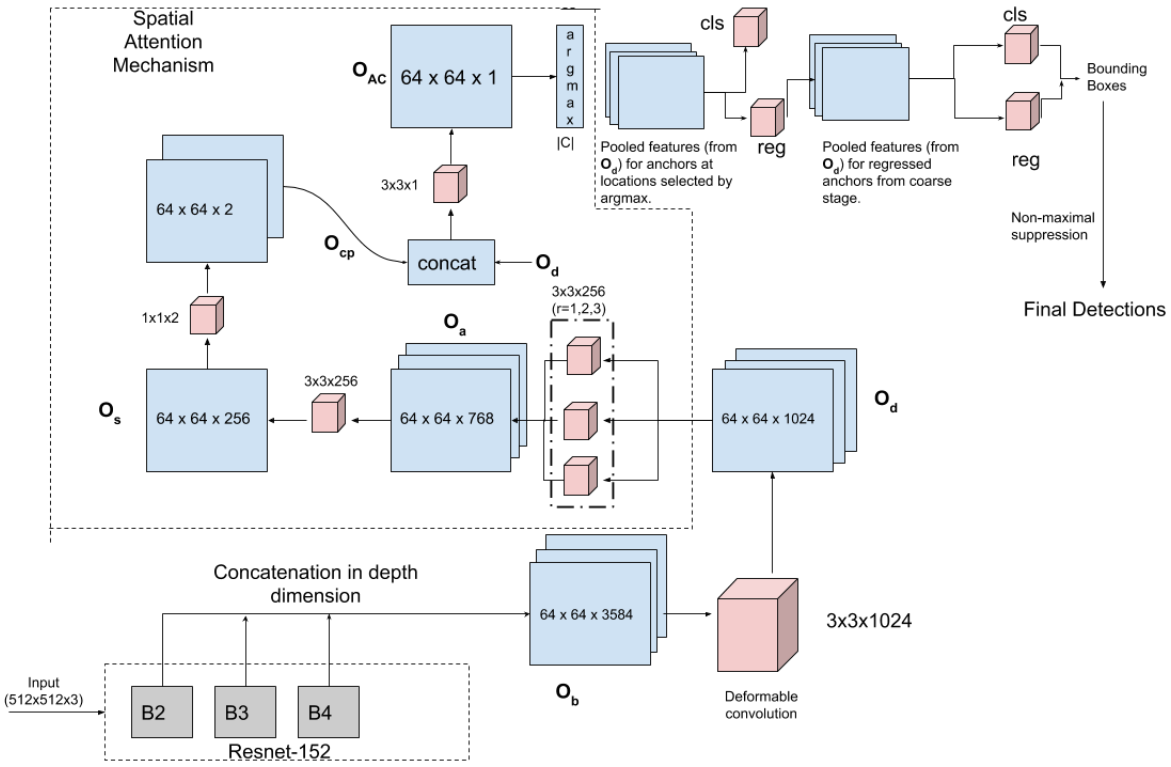

Figure 2. Block diagram of the proposed approach. At the fine stage of classification and regression, we take as input the selected anchors and the regression outputs from the coarse stage. We perform regression on the selected anchors and subsequently pool features from those regressions.

[5] within CNN layers to ensure that feature maps are of the same dimension and concatenate them. This encapsulates a rich feature diversity for processing by a detection head. Our early fusion approach is discussed in section 2.3.

4. **Observation 4**
   A single classification-regression head performs worse than multiple such heads [4]. Anchor boxes have a predefined configuration while pedestrians may appear at arbitrary locations. It is thus possible that an anchor may have only a small overlap with a pedestrian. This corresponds to a partial view of a pedestrian. Multiple heads for classification and regression such as in a cascaded setting as proposed in cascade-RCNN [4], successively refine the bounding box and thus build up a better understanding of a pedestrian from its partial view.
   Contribution 4
   In cascade-RCNN [4], 4 cascade steps are used; each optimized for a different intersection-over-union (IoU) of a detection with the groundtruth. We utilize only 2 cascade steps and our approach does not need a hyper-parameter specification of the target IoU as in [4]. More precisely, we do not model these steps for detections with successively increasing IoU of detections as in [4].

We demonstrate the performance of our proposed single-stage detector over two public datasets – a) caltech reasonable [8] and, b) citypersons [18] dataset.

2. Proposed Approach

Notations We denote àtrous rate as r. Unless otherwise stated, $r = 1$ (*standard convolution*). Padding of a feature map is denoted by P. Unless otherwise stated, $P = 0$. All paddings are *zero-paddings*. A convolutional layer's configuration is denoted in the form $filter\_height \times filter\_width \times num\_filters$. All filter strides in our work are 1. Input image and feature map dimensions are denoted in the form $height \times width \times num\_channels$. It is implicitly assumed that in implementations an additional leading dimension of *batchsize* is used.

2.1. Outline

The overall block diagram of our proposed approach is shown in figure 2. We now describe the individual components of our system in the subsequent subsections.

2.2. Input

Our input (I) is a batch of RGB images ($512 \times 512 \times 3$).

2.3. Base Network

We use a ResNet-152 [9] network pretrained on the imagenet dataset [7] as the base network for feature extraction. We employ à trous convolution with *rate*=2 at the last convolutional layer of blocks $2, 3$ and 4 of the ResNet-152, to ensure that the output feature maps from these layers are of the same dimensions; each with an output stride of 8. These feature maps are concatenated depthwise to obtain a feature map for further processing. We call it $\boldsymbol{O_b}$ ($64 \times 64 \times 3584$).

2.4. Deformable convolution

Deformable convolution [6] has been shown to sample features from non-uniform locations in a feature map. Compared to standard convolution, it is able to better learn an object's structure. Following this, we use a deformable convolutional layer ($3 \times 3 \times 1024; P = 1$) with $\boldsymbol{O_b}$ as the input. We denote its output as $\boldsymbol{O_d}$ ($64 \times 64 \times 1024$).

2.5. Spatial Attention

Our spatial attention module is aimed at being a lightweight but robust alternative to RPN. Unlike RPN, our module does not perform classification or bounding box regression to generate proposals. We describe the components of our proposed spatial attention module below

Semantic fully convolutional layer
We utilize a simple semantic segmentation approach to focus on the most probable pedestrian locations. 3 parallel branches of convolutional layers ($3 \times 3 \times 256; P = 1$) with $r =$ 1,2 and 3 respectively; process $\boldsymbol{O_d}$. The outputs of the 3 branches are depthwise concatenated to obtain a feature map $\boldsymbol{O_a}$. Another convolutional layer ($3 \times 3 \times 256; P = 1$) processes $\boldsymbol{O_a}$ to obtain $\boldsymbol{O_s}$. $\boldsymbol{O_s}$ is in turn is fed to a $1 \times 1 \times 2$ kernel which performs per-pixel prediction over 2 classes – *pedestrians* and *background*; resulting in $\boldsymbol{O_c}$ ($64 \times 64 \times 2$). In the lack of accurate segmentation masks in datasets like $caltech - reasonable$, we utilize the bounding box as an pseudo segmentation mask [2]. This mask provides the groundtruth for per-pixel prediction. We take the channel of $\boldsymbol{O_c}$, corresponding to pedestrian prediction and refer to it as $\boldsymbol{O_{cp}}$. Some examples of $\boldsymbol{O_{cp}}$ are shows in figure 3.

Anchor Center Location Inference
We infer a set of locations (C) on the feature map $\boldsymbol{O_d}$ as described below. $\boldsymbol{O_{cp}}$ and $\boldsymbol{O_d}$ are concatenated in the depth dimension. The resulting feature map is processed by a convolutional layer ($3 \times 3 \times 1; P = 1$) to obtain $\boldsymbol{O_{AC}}$. We then select C as follows :

$$C = argmax_{|C|}(\sigma(\boldsymbol{O_{AC}})) \tag{1}$$

where $\sigma(.)$ is the sigmoid function and $argmax_{|C|}(\boldsymbol{O_{AC}})$ is a function representing top $|C|$ locations in $\boldsymbol{O_{AC}}$. The parameter $|C|$ is a hyperparameter in our system. To facilitate backpropagation, we define the gradient of $argmax(.)$ as the identity function, which passes the gradients coming from the top layers to the bottom layers without modification. Anchors centered at C, are known as the candidate anchors in our work.

All the proposal anchors centered at locations defined by C, are subsequently used for classification and bounding box regression.

2.6. Classification and Bounding Box Regression

In our work we perform classification and bounding box regression in two steps – *coarse* and *fine-grained*. For both steps, features pooled from the anchors located at C are used for classification and regression.

Coarse-Step
We pool features from the proposal anchors in $\boldsymbol{O_d}$. For pooling we crop the features inside the anchors and resize them to a fixed size followed by max-pooling by a 2×2 kernel with stride of 2. In our implementation the fixed size is 14×14. These features are fed to two sibling fully connected layers for classification and regression respectively.

Our regression function predicts – s_x (horizontal scale factor), s_y (vertical scale factor), t_x (horizontal translation) and t_y (vertical translation). Given an anchor with center (x_c, y_c), width as w and height as h, it is transformed as follows:

$$\begin{aligned} x_c^{new} &= x_c + t_x \\ y_c^{new} &= y_c + t_y \\ w^{new} &= w \times s_x \\ h^{new} &= h \times s_y \end{aligned} \tag{2}$$

In equation 2, variables on the left hand side are the transformed variables after regression.

Fine-grained Step All the anchors classified as pedestrians by the coarse-step classification, are transformed according to equation 2. The process of coarse-step classification and regression is then repeated using a different set of classification and regression layers, resulting in the final set of bounding boxes detected as pedestrians on the image.

In our work, since the fine-step of classification and regression, pools features from the regressed anchors of the coarse-step, it extracts features from an improved location.

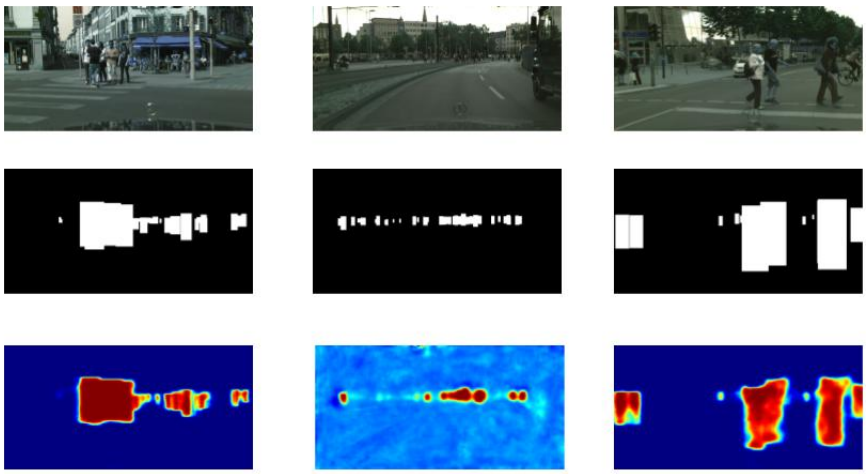

Figure 3. (**Top**: Original images, **middle** : groundtruth pseudo segmentation mask and **bottom**: $\boldsymbol{O_{cp}}$ from spatial attention. $\boldsymbol{O_{cp}}$ has been resized to original image size with bicubic interpolation.

This helps in an improved prediction of bounding box coordinates. It is possible to cascade multiple such steps. However, in our implementation use of 2 steps was found sufficient for good results. Addition of subsequent steps increases the number of feature pooling operations – thereby decreasing inference speed.

3. Training

3.1. Loss Function

During training, all anchors overlapping with an $IoU > 0.5$ are selected as positive anchors, the remaining being negative anchors. In our proposed approach there are a total of 5 loss terms as enumerated below :

1. Spatial Attention loss (L_{att}) : This refers to the pixelwise cross-entropy between $\boldsymbol{O_c}$ and the groundtruth pseudo segmentation mask, averaged across all pixels. This cross-entropy is computed for 2 classes – *pedestrian* and *background*.

2. Coarse-step classification loss L_{cls}^{coarse} : The coarse stage classification loss is also a cross-entropy loss for the classification of a proposal anchor as *pedestrian* or *background*.

3. Coarse-step regression loss L_{reg}^{coarse} : For regression we use the smooth-L1 loss as used in [14]. Regression is performed only for proposal anchors classified as pedestrians. The regression loss is $L_1^{smooth}(p_i, p_i^*)$, where p_i denotes the transformation of the anchor box variables (s_x, s_y, t_x, t_y), while p_i^* denotes the same for the groundtruth box. These transformations are described in equation 3.

$$\begin{aligned} p_{t_x} &= log(\frac{t_x - x_c}{w_a}) \\ p_{t_y} &= log(\frac{t_y - y_c}{h_a}) \\ p_{s_x} &= log(s_x) \\ p_{s_y} &= log(s_y) \end{aligned} \tag{3}$$

   w_a and h_a are the width and height of the anchor respectively. The scaling is done assuming the scale of the anchor box as 1. Faster-RCNN during the RCNN stage, starts with proposal boxes which are already regressed. In our approach, there is a bigger difference between anchor and groundtruth bounding box dimension. We found that regression using s_x and s_y instead of width and height as in [14] lead to more stable training.

4. Fine-grained step classification loss L_{cls}^{fine} : Same as the formulation of L_{cls}^{coarse}.

5. Fine-grained step regression loss L_{reg}^{fine} : Same as the formulation of L_{reg}^{coarse}

6. Regularization losses L_R : We use l_2 regularization in all our convolutional layers. L_R is the sum of all the regularization terms in our detector.

The total loss function for training our detector is

$$L_{total} = \frac{\alpha L_{att}}{N} + \frac{L_{cls}^{coarse} + L_{reg}^{coarse} + L_{cls}^{fine} + L_{reg}^{fine}}{N_a} + \lambda L_R \tag{4}$$

In equation 4, α is a constant scaling factor, while N_a is the minibatch size (*total number of anchors*) and N is the

Test Set	LAMR
Caltech1x (trained on caltech10x)	4.83
Caltech1x (trained on citypersons + caltech10x)	3.79
Citypersons-val (trained on citypersons train)	11.58

Table 1. Performance of the proposed approach on caltech1x and citypersons-validation set. ($|C| = 350$).

Method	Caltech1x	Citypersons-val
RPN-BF[16]	9.6	N/A
MSCNN[3]	10	N/A
SDS-RCNN[2]	7.6	N/A
GDFL[11]	8	N/A
SSD (Resnet-152)[13]	11	N/A
Rep-Loss[15]	4	13.1
Ours	**3.79**	**11.58**

Table 2. Comparative performance of the proposed approach with other pedestrian detectors.

batchsize of images. λ is the regularization factors (set to 0.00005 in all our experiments). Pedestrian bounding boxes in training data contain several background elements as well. Our experiments with $\alpha = 1$, suggest that it causes small-scale pedestrians to be missed, especially with large-scale pedestrians around. Since L_{att} is obtained by averaging over all the pixels, large-scale pedestrians contribute more to it. Scaling L_{att} by small value of α is found by us to be a better approach for improved detections. In all our experiments, $\alpha = 0.8$.

3.2. Implementation Details

In our implementation, we obtain the best results with $|C| = 350$. We use anchors with two aspect ratios $(0.41, 2.41)$ and 6 scales $(0.25, 0.5, 0.75, 1, 2, 4)$ resulting in 12 anchors at each location. All anchors have been generated with a base anchor size of 64×64. Thereby, $12 \times 60 = 720$ anchor features are the input to our coarse-stage classification.

For training, we utilized the stochastic gradient descent algorithm. We did a warm-up phase where the learning rate ($lr = 0.003$) for first $10K$ steps, after which lr was reduced by a factor of 0.5 after every $10K$ steps. Data augmentation was used in our training experiments with random horizontal flip and random brightness and contrast adjustments. The batchsize used in our experiments was 2. Our proposed system was implemented in TensorFlow [1] and was run on a single NVidia V100 GPU with 32 GB of GPU memory.

4. Experiments and Results

4.1. Datasets

We experimented with the caltech-reasonable [8] and citypersons [18] datasets. For caltech-reasonable, we utilized the improved annotations provided by [17] for the $10x$ training set (42782 images) and for the $1x$ testing set (4024 images). For the citypersons dataset we used the original annotations from the authors [18]. For citypersons we trained on the training set (2975 images) and tested on the validation set (1525 images).

For both the datasets, we utilized only the reasonable subset annotations (height$\geq$ 50 pixels, occlusion$\leq$ 0.35) for training. We use *log-averaged miss-rate* (**LAMR**) [8] as the evaluation metric.

Two-step (cls+reg)	Deformable Convolution	Early Fusion	LAMR
✓	✓	✓	4.83
✓	✓	–	6.22
✓	–	✓	5.81
–	✓	✓	5.67
–	–	–	7.19

Table 3. Summary of ablation studies on caltech-1x dataset. These ablations were performed for $|C| = 350$ and directly training the system on caltech10x-train.

Method	Inference Speed (fps)
RPN-BF[16]	7
MSCNN[3]	8
SDS-RCNN[2]	5
SSD (VGG16)[13]	48
Rep-Loss[15]	N/A
Ours	11

Table 4. Comparative performance of the inference speed of various detectors. For our proposed detector, the inference speed corresponds to $|C| = 350$.

4.2. Results

4.2.1 Detection Accuracy

Our best results were obtained with $|C| = 350$, by training our system first on caltech10x training set followed by fine-tuning on the citypersons training set. Table 1 summarizes the results on caltech-reasonable and citypersons (validaton) datasets for different choices of training set. In table 2, we provide a comparison of our best numbers with several other pedestrian detectors. Table 2 shows that our system outperforms the other methods on both *caltech-reasonable* and *citypersons* validation sets. Methods such as [3, 15, 2] perform upscaling of input images prior to feeding them to their systems. Upscaling has been shown in [3, 15] to improve performance by around $0.4 - 1.7$ LAMR points. Upscaling input images to higher sizes is likely to further improve the performance of our system.

Figure 4. Some detections (**Left**) on the validation set of caltech-1x dataset compared with corresponding groundtruth (**Right**).

|C|	LAMR	
	Caltech-1x	**Citypersons-val**
30	8.59	15.3
50	8.41	14.73
60	8.05	14.21
100	6.69	13.95
350	**3.79**	**11.58**

Table 5. Effect of varying the hyper-parameter $|C|$. The miss-rate for caltech-1x is based on model pre-trained on citypersons-train followed by fine-tuning on caltech-1x. The miss-rate for citypersons-val is based on model trained on citypersons-train.

|C|	**Detection Speed (fps)**
30	60
60	36
90	24
100	20
350	**11**

Table 6. Impact of varying the hyper-parameter $|C|$ on inference speed.

4.2.2 Detection Speed

Our system operates at an inference speed of $\sim 11 fps$ under a batchsize of 2 images, for $|C| = 350$. This speed is comparable with the performance of most Faster-RCNN based systems. As table 6 shows, the inference speed decreases with an increase in the value of $|C|$. This is expected since the number of feature pooling operations is proportional to the value of $|C|$. Table 6 in conjunction with table 5 details the relative loss in detection accuracy for an increase in inference speed by decreasing the value of $|C|$. Even at $|C| = 100$, the performance obtained is quite competitive with the detection speed being at 20 fps. This shows that the proposed system offers considerable flexibility in terms of detection speed while maintaining competitive detection performance. We also observe from table 6 that, for lower values of $|C|$, the inference speed can be as high as 60 fps, which is higher than the speed of SSD [13] (with VGG16 as base network) (table 5). Feature pooling is a slower operation and is avoided by SSD for that purpose. However, for lower values of $|C|$, even with intra-anchor feature pooling, our proposed system performs fewer operations compared to SSD. This is on account of the fact that, SSD performs multi-scale computations which effectively increase its computations compared to our approach. However, for higher values of $|C|$, there are significantly more pooling operations which results in inference speeds more similar to other faster-rcnn based approaches.

4.3. Ablation Studies

In table 3 we summarize the ablation studies in our experiments. The use of deformable convolution assists in improved $\boldsymbol{O_{cp}}$ and provides a boost of ~ 0.9 LAMR points. The early fusion of multiple layers is responsible for a gain of $\sim 1.4\%$ in LAMR, thereby verifying the effectiveness of using multiple layers. To study the role of two-stage classification and regression, we switched to single-step classification and regression, which lowered our LAMR by 1.04%. Most of this loss in LAMR was observed for pedestrians with occlusion close to 0.35% (upper limit of occlusion in caltech-reasonable), and was owing to insufficient bounding box overlap. This is expected, as an anchor may not overlap fully with a pedestrian under high occlusion conditions. Hence, it is difficult to estimate the full bounding box by the partial view of a pedestrian.

In table 5 we show the effect of varying the hyper-parameter $|C|$ which decides the number of anchors to be

selected. Higher values of $|C|$ lead to improved performance, but at the cost of inference speed as it leads to more number of feature pooling operations.

4.4. Spatial Attention : Comparison with RPN

It is not straightforward to quantitatively compare spatial attention with RPN. Spatial attention module does not perform any bounding box regression and does not select all anchors at any specified location. Thus, although RPN and spatial attention in our work – both aim to reduce the search space for pedestrians, the underlying mechanisms are different. Traditionally RPN is evaluated by measuring the IoU of its proposal outputs with the groundtruth to compute the LAMR. In our work since no regression is performed on the anchors by the spatial attention, comparison using IoU is not fair.

For a fairer comparison, we trained Faster-RCNN with ResNet-152 on caltech10x and compared its performance on caltech1x with our approach for the configuration in the last row of table 3. In this case, the only difference between Faster-RCNN and our approach remains in our use of spatial attention. We gain 0.42% of performance over standard faster-RCNN (table 7). This basic comparison serves to show that the performance of the proposed spatial attention module is comparable to the RPN performance.

Faster-RCNN	**Ours**
12.02	**11.6**

Table 7. Comparison between spatial attention and RPN on the basis of LAMR

5. Conclusions

We propose utilizing spatial attention for pedestrian detection. We show that spatial attention can be a potential alternative to RPN. It allows selecting a sparse set of anchor proposals, which limit the computations performed during detection, leading to higher inference speeds. Processing a small set of anchors allows feature pooling from anchors to be performed without the expense of processing time. In nutshell, our proposed system combines the characteristic traits of two-stage (e.g:- Faster-RCNN) and single-stage (e.g:- SSD, YOLO) detectors. Our proposed system achieves state-of-art performance across two major pedestrian detection benchmark datasets. In future, we aim to extend similar ideas to general category object detection problems.

References

[1] M. Abadi et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.

[2] G. Brazil, X. Yin, and X. Liu. Illuminating pedestrians via simultaneous detection & segmentation. *arXiv preprint arXiv:1706.08564*, 2017.

[3] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. In *European Conference on Computer Vision*, pages 354–370. Springer, 2016.

[4] Z. Cai and N. Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, pages 6154–6162, 2018.

[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 40(4):834–848, 2018.

[6] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. *CoRR, abs/1703.06211*, 1(2):3, 2017.

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. Ieee, 2009.

[8] P. Dollar et al. Pedestrian detection: An evaluation of the state of the art. *IEEE transactions on pattern analysis and machine intelligence*, 34(4):743–761, 2012.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *CVPR*, pages 7310–7311, 2017.

[11] C. Lin, J. Lu, G. Wang, and J. Zhou. Graininess-aware deep feature learning for pedestrian detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 732–747, 2018.

[12] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017.

[13] 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. Springer, 2016.

[14] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[15] X. Wang, T. Xiao, Y. Jiang, S. Shao, J. Sun, and C. Shen. Repulsion loss: detecting pedestrians in a crowd. In *CVPR*, pages 7774–7783, 2018.

[16] L. Zhang, L. Lin, X. Liang, and K. He. Is faster r-cnn doing well for pedestrian detection? In *European Conference on Computer Vision*, pages 443–457. Springer, 2016.

[17] S. Zhang, R. Benenson, M. Omran, J. Hosang, and B. Schiele. Towards reaching human performance in pedestrian detection. *TPAMI*, 40(4):973–986, 2018.

[18] S. Zhang, R. Benenson, and B. Schiele. Citypersons: A diverse dataset for pedestrian detection. In *CVPR*, volume 1, page 3, 2017.