A One-and-Half Stage Pedestrian Detector

Ujjwal
Institut VEDECOM
INRIA
ujjwal.ujjwal@inria.fr

Aziz Dziri
Institut VEDECOM
aziz.dziri@vedecom.fr

Bertrand Leroy
Institut VEDECOM
bertrand.leroy@vedecom.fr

François Bremond
INRIA
francois.bremond@inria.fr

Abstract

Pedestrian detection is a specific instance of the more general problem of object detection in computer vision. A balance between detection accuracy and speed is a desirable trait for pedestrian detection systems in many applications such as self-driving cars. In this paper, we follow the wisdom of " and less is often more" to achieve this balance. We propose a lightweight mechanism based on semantic segmentation to reduce the number of anchors to be processed. We furthermore unify this selection with the intra-anchor feature pooling strategy adopted in high performance two-stage detectors such as Faster-RCNN. Such a strategy is avoided in one-stage detectors like SSD in favour of faster inference but at the cost of reducing the accuracy vis-à-vis two-stage detectors. However our anchor selection renders it practical to use feature pooling without giving up the inference speed.

Our proposed approach succeeds in detecting pedestrians with state-of-art performance on caltech-reasonable and ciypersons datasets with inference speeds of ~ 32 *fps.*

1. Introduction

Detection of pedestrians has important applications including *surveillance* and *autonomous vehicles*. High detection accuracy and fast inference are defining expectations from a pedestrian detection technique in the aforementioned applications.

High inference speed is desirable in applications such as *autonomous vehicles* for safety reasons. Essentially high inference speed is a trade-off against high detection accuracy [10]. High detection accuracy is usually associated with *two-stage* detectors such as Faster-RCNN [24] and Mask-RCNN [8] at the cost of inference speed. *One-stage* detectors like YOLO [22] and SSD [15] provide high inference speed at the cost of detection accuracy. In this work, we propose an approach to pedestrian detection which balances the *speed/accuracy* trade-off in pedestrian detectors. We achieve high detection accuracy while attaining a high speed of detection as shown in figure 1.

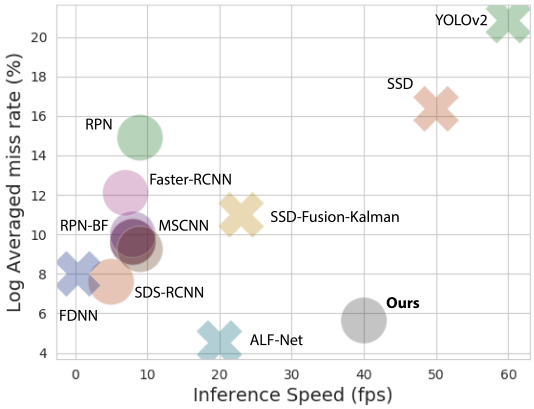

Figure 1. Speed/Accuracy scatter plot of various pedestrian detectors categorized into *one-stage*(**crossed**) and *two-stage* (**circles**) detectors for the caltech-reasonable testing set dataset.

What are the traits which describe the *speed vs. accuracy* behavior of *two-stage* and *one-stage* pedestrian ? Faster-RCNN [24] and SSD [15] are the two earliest representatives of *two-stage* and *one-stage* detectors. Their comparative illustration is shown in figure 2. We describe their basic working below to pinpoint their characteristics which are relevant for our work. It will be seen that their *speed vs. accuracy* behavior stems from their feature handling mechanism.

The working of *two-stage* detectors follows the steps of – a) proposal detection "*first stage or proposal stage*" and, b) object class detection "*second stage or detection stage*".

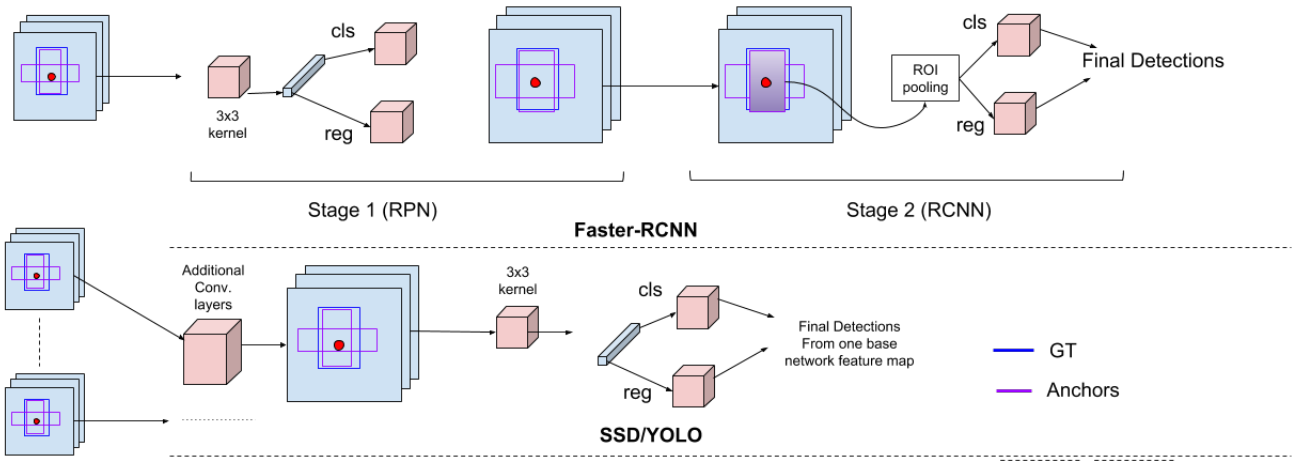

Figure 2. Processing of anchors by Faster-RCNN [24] (**Top**) and by SSD[15]/YOLO[22] (**Bottom**)

The *first stage* uses a convolutional kernel of *fixed size* over a feature map to transform it to another feature map known as the *proposal map*. Hypothetical bounding boxes (*or anchors*) are tiled over the proposal map. Usually multiple anchors with varying scales and aspect ratios are centered at each location in the proposal map. They are called confocal anchors. Feature vector at each proposal map location is used for 2-class (*object vs. no object*) classification and bounding box regression of anchors (*regression done only for anchors classified as objects*. The anchors classified as *objects* after regression are known as proposals. In this setting, all confocal anchors share the same feature vector – a major limitation as confocal anchors cover varying spatial areas. The proposals are then processed by the second stage. In the second stage, feature pooling from proposal regions is employed for isolating feature map representing each proposal region. Feature pooling refers to extraction of features from a sub-region of a feature map. The pooled features (*after flattening*) are fed to a dedicated classifier and regressor to determine the final detected bounding boxes. While the mechanism for generating feature vectors in the first stage is inaccurate, it is adopted in favor of speed. The more accurate but computationally heavier approach of feature pooling is reserved for the second stage, where only object proposals (*often a tiny fraction of total number of anchors* need processing.

One-stage detectors bypass the proposal region generation phase. However, their only stage (*detection stage*) works very similar to the first stage of *two-stage* detectors. For a $C + 1$ class detection problem (*+1 for background*), anchors (also called *priors* in SSD literature) are represented by the feature vector at the location they are centered on. Thus as in the first stage of *two-stage* detectors, all confocal anchors share the same feature vector. Multi-scale training and testing is used in *one-stage* detectors where feature maps from multiple CNN layers are processed as described above, followed by coalescing of final detections. Unlike *two-stage* detectors, there is no proposal detection phase and so number of sub-regions to be processed for final detections is much higher in *one-stage* detectors.

Despite having multi-scale processing and more sub-regions to be processed, *one-stage* detectors have a higher speed. Feature pooling is the major difference between the two classes of detectors. This shows that, feature pooling is a much slower operation than computing convolutions, as the former involves performing copy operations in memory – an intense operation when the number of regions to be processed is large. However, feature pooling is an operation which utilizes the entirety of features within a sub-region of a feature map and hence *two-stage* detectors deliver better detection performance than *one-stage* detectors.

The time and space complexity of feature pooling scales linearly with number of sub-regions to be processed. Thus, to perform detection with high accuracy and high inference speed, it becomes pertinent that the number of sub-regions to be processed is decreased significantly, so as to minimize the detrimental impact of feature pooling on computational time. In this paper, we take cue from the aforementioned observations and propose a pedestrian detection with following characteristics –

1. **Selection of a sparse subset of anchors :** We propose a lightweight mechanism using semantic segmentation to select a small set of anchors in a given image. The proposed semantic segmentation approach requires only groundtruth bounding boxes as *pseudo-segmentation* masks during training. The proposed mechanism –

    - ensures focus on the most probable pedestrian locations. This lowers the possibility of detection of a background region as a *false positive*.

- Has ~ 3 times lesser training parameters than region proposal networks (RPN) [24], promoting faster inference.

2. **Feature pooling :** We have noted that feature pooling in *two-stage* detectors allows for better learning thereby facilitating higher detection accuracy. By selecting a sparse subset of anchors, our proposed approach ensures a low overhead of feature pooling compared to *two-stage* approaches like Faster-RCNN,

3. **Visible and Full pedestrian bounding box detection :** Our proposed approach uses both the visible and full bounding boxes for selection of most relevant anchors. When using only the full-body bounding boxes as in most works, the same IoU can be achieved by several anchors overlapping from different directions. We constrain the criteria for positive pedestrian anchors by involving overlaps with both full-body and visible part of the bounding box. This ensures a relatively consistent feature profile for anchors which includes the relevant pedestrian parts.

We demonstrate the effectiveness of our proposed pedestrian detection system on 2 public datasets – caltech pedestrian dataset [5] and citypersons [29].

2. Related Work

We limit our focus to deep learning based pedestrian detection systems. Most contemporary pedestrian detection systems are derived from Faster-RCNN [24], SSD [15] or YOLO [22]. Of these, Faster-RCNN is most commonly used as the basis for building pedestrian detection systems [28, 1, 2, 17, 19, 30, 13], on account of better detection accuracy than *one-stage* detectors. Pedestrian detectors based on *one-stage* detection systems include [27, 16, 6, 7, 23, 11, 21, 18]. Our treatment of related work focuses on *speed/accuracy* trade-off in contemporary pedestrian detectors and delineating details which offer the cue for balancing this trade-off; thereby setting the basis of our contributions. Figure 1 summarizes the relative performance of various pedestrian detectors vis-à-vis speed and accuracy.

***Two-stage* pedestrian detectors :** Approaches extending Faster-RCNN to pedestrian detection include use of tree-based classifiers [28, 25], use of multiple CNN layers [2, 25], use of additional information such as optical flow, segmentation and different color channels [17, 1], use of different networks for processing different scales of pedestrians [13] and novel loss terms such as repulsion loss [26] for improved localization. These extensions improve upon the generic Faster-RCNN detector for pedestrians by an order of $5 - 7\%$. However a comparable improvement in inference speed is not observed. Often these extensions invoke increased system complexity thereby requiring more *floating point operations per second* (FLOPs), which lowers the inference speed. Generally the performance of *two-stage* detectors varies from $8 - 14$ fps, while that of generic VGG16 based Faster-RCNN detector lies in the range of $7 - 10$ fps.

All *two-stage* pedestrian detectors use region proposal network (RPN) for proposal generation. As mentioned before and illustrated in figure 2, the features for proposal generation in RPN are generated without intra-anchor feature pooling. These proposals are often poorly localized [28] and require a further classification and regression stage [28, 1, 2] for improved performance. All the *two-stage* detectors utilize intra-anchor feature pooling after the proposal generation. These pooling operations are carried out over a large number of proposals to minimize the miss-rate. As a result, the inference speed of *two-stage* detectors are limited by the number of processed proposals in addition to system complexity.

One-stage pedestrian detectors : *One-stage* pedestrian detectors are based on SSD [15] or YOLO [22]. The performance of generic SSD and YOLO detectors on pedestrian detection is significantly lower than that of Faster-RCNN. Their extensions to pedestrian detection include multi-step training of SSD [16], use of late fusion of multiple networks to refine the pedestrian candidates generated by SSD [6], recurrent networks for incorporating context [23] and use of skip connections in YOLO [11]. These extensions have improved their performance vis-à-vis their generic counterparts. The recently proposed ALF-net follows the ideas of cascade-RCNN [3], but over SSD [15]. ALF-net achieves an impressive 4.5% miss-rate on caltech-reasonable dataset while operating at ~ 20 fps. This performance is still lower than the performance of generic SSD ($48 - 60$ fps). Other *one-stage* pedestrian detectors [20, 11, 21, 18] report their runtime performance in the range of $20 - 25$ fps which is substantially lower than their generic counterparts. This reduction is primarily the result of added system complexity. For example [6] performs late fusion of multiple CNN networks, each of which operates upon the pedestrian candidates generated by a SSD which is pre-trained to generate pedestrian proposals. At the same time, *one-stage* detectors share the lack of intra-anchor feature pooling which fails to provide as relevant pedestrian features as *two-stage* detectors.

Use of semantic segmentation for pedestrian detection : The use of semantic segmentation in a deep learning setting for pedestrian detection was used in F-DNN [6]. The masks in [6] are predicted by a separate network trained for semantic segmentation and then used as a post-processing

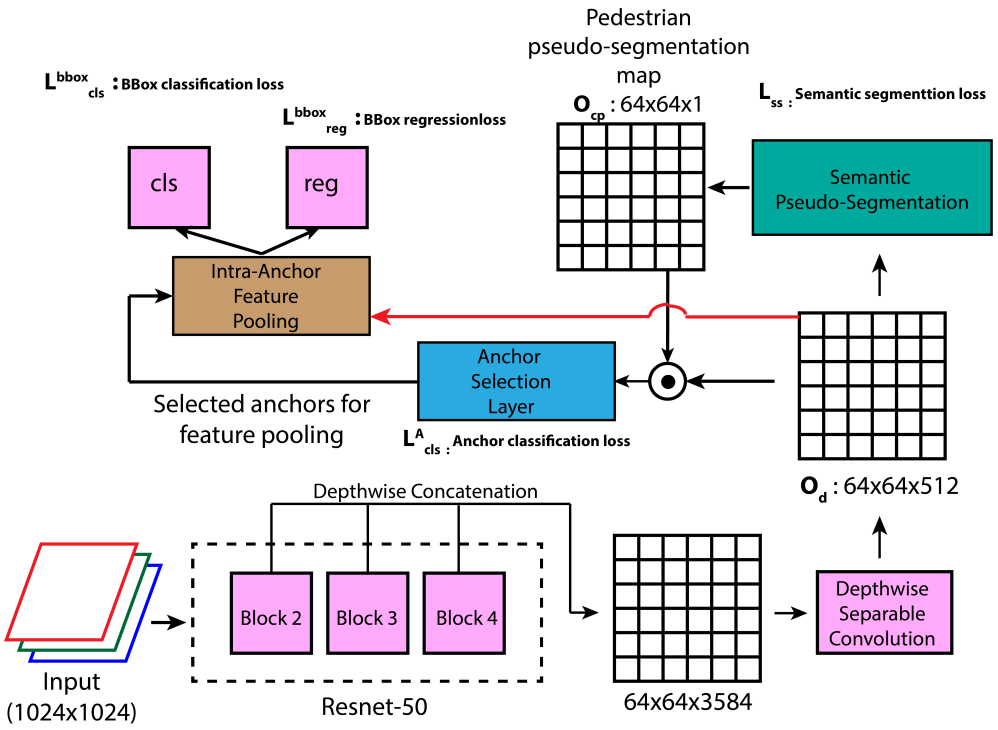

Figure 3. Overview of the proposed approach. Semantic segmentation layer is illustrated in figure 5 and the anchor classification layer is shown in figure 7. T he various loss terms contributed by various components are shown besides the component blocks.

cue to remove invalid detections. This however makes it difficult to tune F-DNN for datasets like caltech pedestrians [5] which do not come with groundtruth segmentation masks. This deficiency is answered by SDS-RCNN [1]. SDS-RCNN uses the pedestrian bounding box in the training data to construct a pseudo-segmentation mask. A pixel-wise cross entropy term to classify background from pedestrian instances augments the standard loss function of the RPN, thereby forcing the output feature map of RPN to focus better on pedestrian instances. The RPN features are then used for classification and regression by a second network. This multi-task approach though very promising suffers from the limitations discussed before for *two-stage* pedestrian detectors. Furthermore, semantic segmentation has been used with the objective of improving the detection accuracy without any focus on utilizing it for improving the inference speed. In contrast, our proposed approach shows that the use of semantic segmentation based on pseudo-segmentation masks naturally leads to a mechanism to reduce the number of anchors to be processed by as much as 97%. This is the key to invoke intra-anchor feature pooling without suffering a runtime performance setback.

Use of visible and full body bounding boxes : Public datasets for pedestrian detection, are often provided with full-body as well as the visible-body bounding boxes [5, 29]. Few pedestrian detectors utilize only the full body bounding boxes for training. In [31]; a spatial attention based work addressing detection under occlusionl; visible bounding boxes are used in addition to full-body boxes and body part annotations. Body part annotations are coarsely defined based on the other two bounding boxes. The basic idea to re-weight the intra-anchor feature maps with the 3 attention maps; one for each type of annotation. Though it is outperformed by other detectors [1, 2] on caltech reasonable subset, it indicates major improvements in cases with partial or heavy occlusion. This suggests that the use of visible part of bounding boxes aids in detection under occlusion. Intuitively, an anchor box which overlaps sufficiently well with both the visible and full body bounding box is a better candidate for regression. Regression of such an anchor box is easier owing to more complete information about the pedestrian inherent in it. Comparatively, complete dependence on full-body or visible bounding box may lead to a wide variance in the information inherent in the anchor.

Our approach utilizes the ideas of SDS-RCNN [1] but without using a RPN. We prune most of the anchors away and use a combination of visible and full body bounding boxes to select positive and negative anchors which are feature pooled for final classification and regression. This approach therefore leverages the best of both worlds – *two-stage* (intra-anchor feature pooling favoring detection accuracy) and *one-stage* (reduced computations favoring inference speed).

Figure 4. (**Top**: Original images. **bottom**: $\boldsymbol{O_{cp}}$ from spatial attention. $\boldsymbol{O_{cp}}$ has been resized to original image size with bilinear interpolation for better visibility.

3. Proposed Approach

The proposed approach consists of – a) an *anchor selection stage*, called a 0.5-stage and, b) a *detection stage*, called a 1-stage. The *anchor selection stage*, selects a set of anchors for feature pooling. Unlike the first stage of Faster-RCNN, our approach does not perform bounding box regression when selecting anchors. Due to the absence of regression, we illustrate our difference from Faster-RCNN by referring to our first stage as a 0.5-stage. The 0.5-stage coupled with the second stage of classification and bounding box regression makes our approach a 1.5-stage pedestrian detection.

For experiments outlined in this paper, we use ResNet-50 [9] as base network, where we use à trous convolution on the second, third and fourth resnet blocks to ensure that the feature maps from these blocks are of the same dimension (*with output stride of* 16). Due to the large number of feature channels in the concatenated map, we use depthwise separable convolution on it to reduce the feature dimensionality. Depthwise convolution performs per channel processing – a better strategy when processing feature maps from multiple convolutional layers.

As outlined in section 1, our approach aims to minimize the number of sub-regions to be processed for final detections. Feature pooling can then be employed over these sub-regions for better feature handling vis-à-vis *one-stage* detectors for high detection accuracy. There are two key ideas in our work, which achieve this reduction in the number of sub-regions to be processed. They form the basis for our 0.5-stage and are outlined in the following two subsections.

3.1. Pseudo-Semantic Segmentation

Given the bounding box of a pedestrian, all the pixels lying within the rectangle can be thought to approximately constitute a *pseudo*-segmentation mask. We utilize semantic segmentation of this mask to reduce the number of anchors to be processed. This is significantly different from other techniques using *pseudo*-segmentation mask such as SDS-RCNN [1], MSDS-RCNN [12], GDFL [14] and PAD [32], which limit the usage of *pseudo*-segmentation mask to improve the feature maps and do not harness its usefulness in improving the detection speed. From figure 3, we see that during backpropagation, the gradients from semantic segmentation impact the base network and the depthwise separable convolutional layers. Thus, in our approach semantic segmentation aids in improving feature maps for detection and also detection speed by limiting number of anchors to be processed.

Figure 4 shows some pedestrian probability maps ($\boldsymbol{O_{cp}}$ in **figure 3**) generated by a simple semantic segmentation approach (**figure 5**), which is reminiscent of the *à trous spatial pyramid pooling* (ASPP) module in deeplabv2 [4]. The high selectivity of ASPP for pedestrians indicates that only high probability regions in $\boldsymbol{O_{cp}}$ need be processed for pedestrian detection. This can help eliminating false positive regions early in the pipeline.

Anchors are tiled across a feature map, with multiple anchors centered at each location. For a $N \times N$ feature

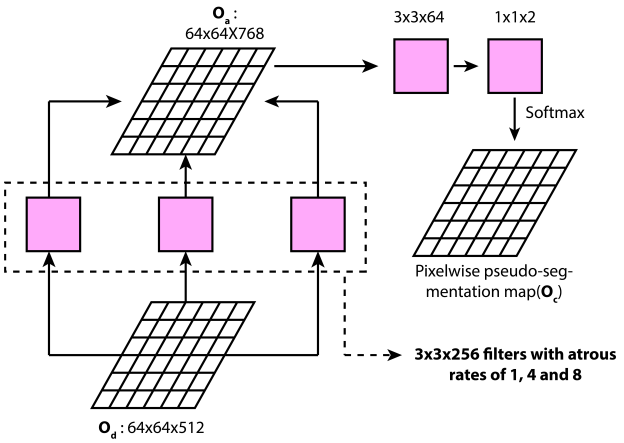

Figure 5. The semantic segmentation layer used in the proposed approach.

Figure 6. An occluded pedestrian with full-body bounding box (**green**) and visible bounding box (**blue**). **Red** anchors are confocal with the **magenta** anchor. The **red** anchors do not overlap well with both the full-body and visible bounding box, while the **magenta** anchor has sufficient overlap with both.

map with N_A anchors per location, the total number of anchor regions is N^2N_A. With a fraction θ ($0 < \theta < 1$), of N^2 eliminated as low-probability regions in $\boldsymbol{O_{cp}}$, only $(1-\theta)N^2N_A$ anchors remain to be processed. In our experiments we conclude that for most images in caltech and citypersons, $\theta \geq 0.7$. However, not all N_A confocal anchors optimally cover a pedestrian and hence a fraction of them can be eliminated from final processing. We achieve this using the anchor classification layer described next.

3.2. Anchor classification layer

To select a subset of N_A confocal anchors at each location; optimally covering a pedestrian, we perform a 2-class anchor classification – *positive anchors* and *negative anchors*. Only the *positive anchors* are subsequently set to the detection stage for feature pooling. This classification is performed over $\boldsymbol{O_h}$ – the schur product of $\boldsymbol{O_{cp}}$ and $\boldsymbol{O_d}$

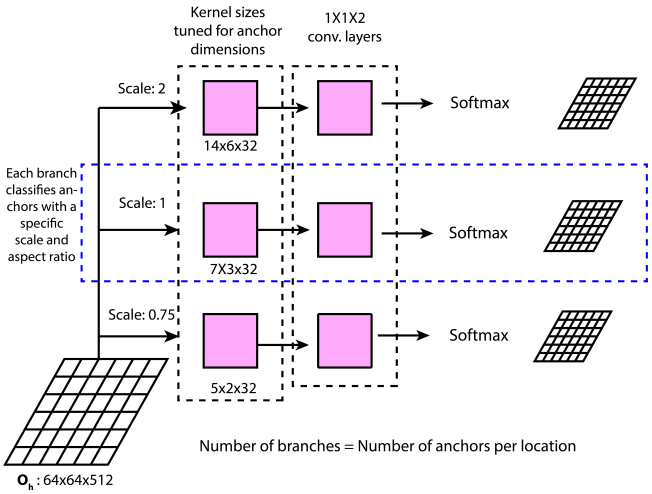

Figure 7. The anchor classification layer. For illustration it is assumed that all anchors have been generated by a base anchor of size 64×64 and have an aspect ratio of 0.41 (*width/height*). The anchor at scale 1 then corresponds to a box of size $\sim 100 \times 41$. For a feature stride of 16, a kernel of size 7×3 will cover the corresponding area of this box in the feature map. For other scale values, the kernel size can be similarly defined.

(broadcasted across the channel dimensions).

At this point, a total of N^2N_A anchors need to be classified. Instead of using costly feature pooling over the N^2N_A anchors, we directly use convolutional kernels for classification. A set of N_A sibling classification branches are set up. Each classification branch serves the classification of anchors with a specific scale and aspect ratio. The i^{th} classification branch is constituted of a convolutional layer with 32 filters of size $h_i \times w_i$, followed by a $1 \times 1 \times 2$ convolutional layer, which is then followed by a softmax operation to determine the probability of an anchor to be *positive*. $h_i \times w_i$ is determined by the configuration used for generating anchors. An example is shown in figure 7. The key idea in our anchor classification layer is that of anchor specific kernel sizes. The kernel corresponding to a classification branch has a size which matches the size of the anchor corresponding to the branch. Thus the kernel is able to cover the entirety of the anchor features. This shows that the idea of using anchor specific kernel sizes is a key idea allowing for fast and accurate anchor classification without using any pooling operations.

An anchor overlapping well with both the full body and visible part of the bounding box encapsulates information about the pedestrian and the occlusion and is thus more useful than other anchors. This is illustrated in figure 6, where only the magenta anchor overlaps sufficiently well with both full-body and visible part of the bounding box. Thus, during training for an anchor ψ, its IoU with the full-body bounding box B_F and visible bounding box B_V is computed and the class of the anchor is determined as *pos-*

itive or *negative* as follows

$$class(\psi) = \begin{cases} \textit{positive}, & \text{IoU}(\psi,B_V) \geq \alpha; \text{IoU}(\psi,B_F) \geq \beta \\ \textit{negative} & \text{otherwise} \end{cases}$$

Behavior during training and testing : During training, we utilize both positively and negatively classified anchors, while only *positively* classified anchors are used during testing. Let n_{pos} be the number of *positive* predicted anchors during training. Then the number of negative anchors used is defined as follows :

$$n_{neg} = min(5 \times n_{pos}, M - n_{pos}) \quad (1)$$

where, M is a hyperparameter and is usually limited by the memory limit of the computing device. During testing phase, we use only the *positive* anchors for classification and regression.

This difference in behavior is preferred so that during training, a large range of samples encompassing *positive* and *negative* examples are seen by classifier and regressor for robust learning.

We implemented our proposed approach on top of tensorflow object detection api [10]. We used stochastic gradient descent (SGD) with a momentum value of 0.9 as an optimizer. Our initial learning rate was set to 0.01 with gradients clipped to a value of 10.0. This warm-up phase lasted for $10K$ iterations after which the learning rate was decreased by a factor of 10 after every $30K$ iterations. Data augmentation was used for training in the form of – a) random horizontal flipping, b) random brightness adjustment and c) random contrast adjustment. We upscaled all images using bilinear interpolation to a fixed size of 1024×1024.

3.3. Loss Function

Our loss function can be written as

$$L_{total} \triangleq L_{ss} + L_{cls}^{A} + L_{cls}^{bbox} + L_{reg}^{bbox} + L_{2}^{reg} \quad (2)$$

where,

1. L_{ss} : Average pixelwise cross entropy term for semantic segmentation.
2. $L_{cls}^{A\ i}$: Cross entropy term for anchor classification. It is a sum of N_A cross-entropy terms, where N_A is the number of confocal anchors.
3. L_{cls}^{bbox} : Average cross entropy term for bounding box classification.
4. L_{reg}^{bbox} : Average smooth L1-loss term as defined in [24].
5. L_{2}^{reg} : Average of regularization terms elsewhere in the network.

4. Experiments, Results and Analysis

4.1. Datasets

We used caltech-reasonable [5] and citypersons [29] for validating the performance of the proposed approach. In the citypersons dataset's evaluation protocol, there are 4 distinct categories of evaluation – *pedestrian*, *rider*, *sitting person* and *person (other)*. We cluster these 4 sub-categories are into one and refer to it as *pedestrians*.

	Caltech-Reasonable		**CityPersons**	
	Train	**Test**	**Train**	**Val**
Images	42,782	4024	2,975	500

Table 1. Summary of dataset size of caltech-reasonable [5] and citypersons [29] dataset.

4.2. Hyperparameter settings

Unless and otherwise mentioned, we use parameter settings as mentioned henceforth. Following [28, 1, 2], we use anchors with an aspect ratio of 0.41 (*average aspect ratio of bounding boxes in caltech and citypersons datasets*). We use 6 anchor scales ($\{0.25, 0.5, 0.75, 1, 2, 4\}$), generated from a base anchor of size $(64, 64)$. For anchor classification layer (sec 3.2), $\alpha = 0.3$, $\beta = 0.5$, $M = 2000$.

4.3. Results and Analysis

Table 2 summarizes the LAMR of our proposed approach on caltech-reasonable and citypersons datasets. Table 2 summarizes the comparative performance of our proposed approach with other pedestrian detectors vis-á-vis accuracy and speed on caltech-reasonable and citypersons datasets. We achieve state-of-art performance on both caltech-reasonable and citypersons datasets. Our best results on caltech-reasonable is obtained by first pre-training our model on citypersons training set, followed by fine-tuning it on caltech-reasonable training set. This amounts to 0.77% improvement in LAMR. From table 2, we see that our improvements on citypersons is much higher than on caltech-reasonable. The input to our model is 1024×1024, which results in higher distortion for caltech-reasonable images (640×480) than citypersons images (2048×1024). This lower distortion gives better training to our model and explains higher improvements on the citypersons dataset.

With an optimized implementation, our approach performs inference at $\sim$ 32 fps, on input images of size 1024×1024, which is ~ 1.5 times faster than the next best speeds ($\sim$ 20 fps) achieved by [16]. It is to be noted that we report our inference speed only for the inference evaluation and do not factor in the time for any display or disk access. It is further notable that the speeds reported in [16] are for images of size 480×640, while for us all images are of size 1024×1024. This shows a major speedup attained by our approach vis-á-vis other competitive methods.

Method	Stages	LAMR		Speed
		caltech-reasonable (test) (w/o CP pre-training) (CP pre-trained)	citypersons (val) (trained only on CP)	
Faster-RCNN [24]	2	12.10	15.4	7
SSD [15]	1	17.78 (16.36)	19.69	48
YOLOv2 [22]	1	21.62 (20.83)	NA	60
RPN-BF [28]	2	9.6 (NA)	NA	7
MS-CNN [2]	2	10.0 (NA)	NA	8
SDS-RCNN [1]	2	7.6 (NA)	NA	5
ALF-Net [16]	1	4.5 (NA)	12.0	20
Rep-Loss [26]	2	5.0 (4.0)	13.2	-
Ours	1.5	**4.76 (3.99)**	**8.12**	**32**

Table 2. Performance comparison of the proposed method with other methods for caltech-reasonable test set and citypersons validation set. The speed figures are in *frames per second*.

The above improvements in inference speed are largely possible through a conjunction of semantic segmentation layer and anchor classification layer. The semantic segmentation layer provides high quality segmentation maps, which assist in selecting a small set of locations at which all anchors are classified. The anchor classification layer then further reduces this count by classification. As a result, the number of anchors used during testing phase is proportional to the number of pedestrians in the image.

5. Ablation Studies

5.1. RPN vs. 0.5-stage

LAMR	
RPN	**0.5-stage**
15.18	**8.12**

Table 3. Ablation study of RPN vs. 0.5-stage on the citypersons(validation) dataset.

To know the impact of our 0.5-stage, we replace it with standard RPN layer of Faster-RCNN [24], with the same anchor parameters as reported in section 4.2, and using top 300 proposals for processing. From table 3 shows that the 0.5-stage has a huge impact on performance. This performance boost speaks jointly of the improvements brought about by semantic segmentation and anchor classification layer. Moreover, our 0.5-stage has computational advantages over RPN as shown below.

In the RPN, for a feature map with 512 channels, and 6 confocal anchors, the classifier will admit $512 \times 2 \times 6$ parameters for *foreground/background* classification. The RPN regressor will admit $512 \times 4 \times 6$ parameters for 4 quantities regressed for each anchor box. This leads to a total of $512 \times (4 + 2) \times 6 = 18432$ trainable parameters. This does not include the parameter count of proposal filters and feature projection layer in RPN [24]. In contrast, the union of segmentation and anchor classification modules admit a total of 7048 training parameters in our approach. Thus, compared to a basic RPN with no extra proposal filter and feature projection layers, our approach admits 2.61 times less parameters than RPN. This makes learning easier for our system on smaller pedestrian datasets like citypersons [29] (2975 training images).

Impact of anchor classification layer : To quantify the impact of anchor classification layer (sec 3.2), we removed it from our pipeline and selected all anchors at top 300 locations in $\boldsymbol{O_{cp}}$ (resulting in number of anchors as $300 \times 6 = 1800$). This led to a LAMR of 16.43% on citypersons validation set. This major jump in LAMR is thought to be the result of major class imbalance, which is caused when all 6 anchors at top 300 locations are selected, with very few of them actually overlapping sufficiently well with the pedestrians.

6. Conclusion

We propose a novel pedestrian detection scheme with an emphasis on achieving high detection accuracy and inference speed simultaneously. Our approach relies on semantic segmentation and selection of a small set of anchors which enables intra-anchor feature pooling without sacrificing inference speed and accuracy. Experimental analysis shows that our proposed approach achieves state-of-art performance across caltech-reasonable and citypersons datasets at an impressive inference speed of ~ 32 fps.

References

[1] G. Brazil, X. Yin, and X. Liu. Illuminating pedestrians via simultaneous detection & segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4950–4959, 2017.

[2] 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.

[3] Z. Cai and N. Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6154–6162, 2018.

[4] 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. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[5] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE transactions on pattern analysis and machine intelligence*, 34(4):743–761, 2012.

[6] X. Du, M. El-Khamy, J. Lee, and L. Davis. Fused dnn: A deep neural network fusion approach to fast and robust pedestrian detection. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 953–961. IEEE, 2017.

[7] L. Fang, X. Zhao, and S. Zhang. Small-objectness sensitive detection based on shifted single shot detector. *Multimedia Tools and Applications*, pages 1–19, 2018.

[8] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7310–7311, 2017.

[11] W. Lan, J. Dang, Y. Wang, and S. Wang. Pedestrian detection based on yolo network model. In *2018 IEEE International Conference on Mechatronics and Automation (ICMA)*, pages 1547–1551. IEEE, 2018.

[12] C. Li, D. Song, R. Tong, and M. Tang. Multispectral pedestrian detection via simultaneous detection and segmentation. *arXiv preprint arXiv:1808.04818*, 2018.

[13] J. Li, X. Liang, S. Shen, T. Xu, J. Feng, and S. Yan. Scale-aware fast r-cnn for pedestrian detection. *IEEE transactions on Multimedia*, 20(4):985–996, 2018.

[14] 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.

[15] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[16] W. Liu, S. Liao, W. Hu, X. Liang, and X. Chen. Learning efficient single-stage pedestrian detectors by asymptotic localization fitting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 618–634, 2018.

[17] J. Mao, T. Xiao, Y. Jiang, and Z. Cao. What can help pedestrian detection? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3127–3136, 2017.

[18] V. Molchanov, B. Vishnyakov, Y. Vizilter, O. Vishnyakova, and V. Knyaz. Pedestrian detection in video surveillance using fully convolutional yolo neural network. In *Automated Visual Inspection and Machine Vision II*, volume 10334, page 103340Q. International Society for Optics and Photonics, 2017.

[19] L. Neumann, A. Zisserman, and A. Vedaldi. Relaxed softmax: Efficient confidence auto-calibration for safe pedestrian detection. 2018.

[20] W. Ouyang, H. Zhou, H. Li, Q. Li, J. Yan, and X. Wang. Jointly learning deep features, deformable parts, occlusion and classification for pedestrian detection. *IEEE transactions on pattern analysis and machine intelligence*, 40(8):1874–1887, 2018.

[21] Q. Peng, W. Luo, G. Hong, M. Feng, Y. Xia, L. Yu, X. Hao, X. Wang, and M. Li. Pedestrian detection for transformer substation based on gaussian mixture model and yolo. In *2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, volume 2, pages 562–565. IEEE, 2016.

[22] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[23] J. Ren, X. Chen, J. Liu, W. Sun, J. Pang, Q. Yan, Y.-W. Tai, and L. Xu. Accurate single stage detector using recurrent rolling convolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5420–5428, 2017.

[24] 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.

[25] L. B. Ujjwal, Dziri Aziz and B. Francois. Late fusion of multiple convolutional layers for pedestrian detection. 2018.

[26] X. Wang, T. Xiao, Y. Jiang, S. Shao, J. Sun, and C. Shen. Repulsion loss: detecting pedestrians in a crowd. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7774–7783, 2018.

[27] F. Yang, H. Chen, J. Li, F. Li, L. Wang, and X. Yan. Single shot multibox detector with kalman filter for online pedestrian detection in video. *IEEE Access*, 2019.

[28] 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.

[29] S. Zhang, R. Benenson, and B. Schiele. Citypersons: A diverse dataset for pedestrian detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3221, 2017.

[30] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li. Occlusion-aware r-cnn: detecting pedestrians in a crowd. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 637–653, 2018.

[31] S. Zhang, J. Yang, and B. Schiele. Occluded pedestrian detection through guided attention in cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6995–7003, 2018.

[32] X. Zhao, S. Liang, and Y. Wei. Pseudo mask augmented object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4061–4070, 2018.