

# What Can Help Pedestrian Detection?

Jiayuan Mao<sup>\*†</sup>

The Institute for Theoretical Computer Science (ITCS)  
Institute for Interdisciplinary Information Sciences  
Tsinghua University, Beijing, China  
mjoy14@mails.tsinghua.edu.cn

Yuning Jiang

Megvii Inc.  
Beijing, China  
jyn@megvii.com

Tete Xiao<sup>\*†</sup>

School of Electronics Engineering and Computer Science  
Peking University, Beijing, China  
jasonhsiao97@pku.edu.cn

Zhimin Cao

Megvii Inc.  
Beijing, China  
czm@megvii.com

## Abstract

Aggregating extra features has been considered as an effective approach to boost traditional pedestrian detection methods. However, there is still a lack of studies on whether and how CNN-based pedestrian detectors can benefit from these extra features. The first contribution of this paper is exploring this issue by aggregating extra features into CNN-based pedestrian detection framework. Through extensive experiments, we evaluate the effects of different kinds of extra features quantitatively. Moreover, we propose a novel network architecture, namely HyperLearner, to jointly learn pedestrian detection as well as the given extra feature. By multi-task training, HyperLearner is able to utilize the information of given features and improve detection performance without extra inputs in inference. The experimental results on multiple pedestrian benchmarks validate the effectiveness of the proposed HyperLearner.

## 1. Introduction

Pedestrian detection, as the first and most fundamental step in many real-world tasks, *e.g.*, human behavior analysis, gait recognition, intelligent video surveillance and automatic driving, has attracted massive attention in the last decade [11, 26, 10, 35, 33, 30]. However, while great progress has been made by deep convolutional neural networks (CNNs) on general object detection [24, 19, 7, 14], research in the realm of pedestrian detection remains not as cumulative considering two major challenges.

Firstly, compared to general objects, pedestrians are less

<sup>\*</sup>Equal contribution.

<sup>†</sup>Work was done during their internships at Megvii Inc.

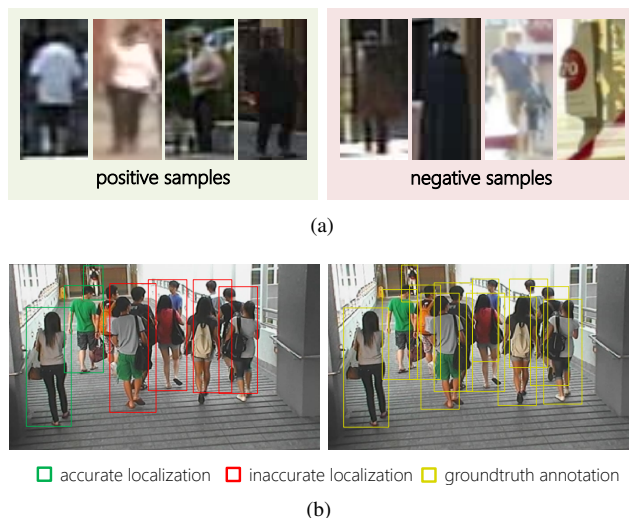


Figure 1. (a) Examples of true pedestrians and hard negative samples of low resolution. Without extra semantic contexts, it is difficult to discriminate between them, even for human eyes. (b) Example of pedestrians in crowded scenes, where CNN-based detectors fail to locate each individual without low-level apparent features.

discriminable from backgrounds. In other words, the discrimination relies more on the semantic contexts. As shown in Figure 1(a), usually appearing in low resolution (less than  $20 \times 40$  pixels), pedestrians together with the cluttered background bring about hard negative samples, such as traffic signs, pillar boxes, and models in shopping windows, which have very similar apparent features with pedestrians. Without extra semantic contexts, detectors working with such low-resolution inputs are unable to discriminate between them, resulting in the decrease in recall and increase in false alarms.

How to accurately locate each pedestrian is the second

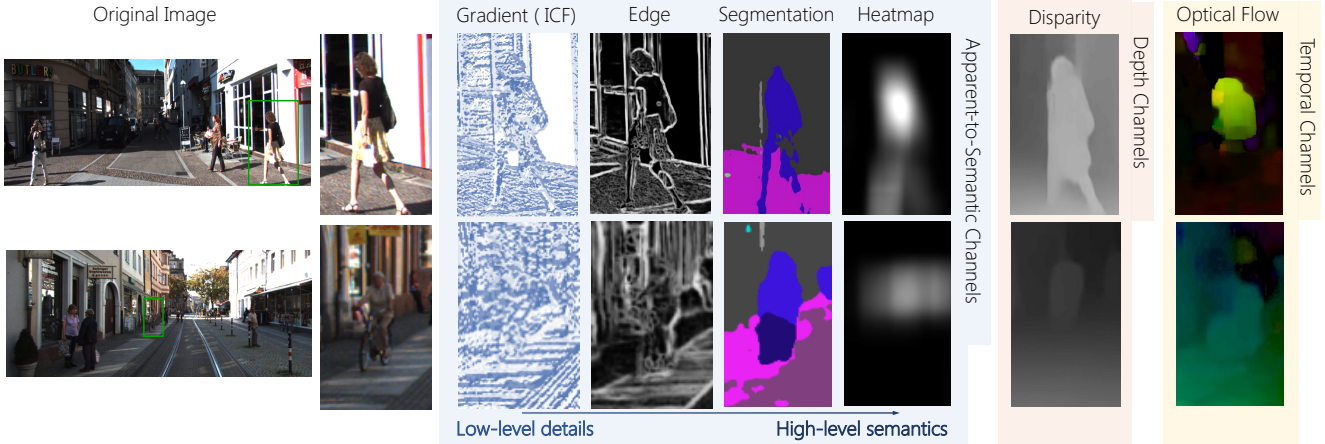


Figure 2. A demonstration of various channel features. Includes: apparent-to-semantic features, temporal features, depth features.

challenge. Figure 1(b) is one showcase in practical applications where the pedestrians stand close in a crowded scene. As a result, detectors typically fail to locate each individual and hence produce a dozen of false positives due to inaccurate localization. This problem becomes even worse for CNN-based detectors since while convolution and pooling layers generate high-level semantic activation maps, they also blur the boundaries between closely-laid instances. An intuitive alternative to address the problem is to make use of extra low-level apparent features (*e.g.* edges), for the purpose of solving the localization drawbacks by providing detectors with detailed apparent information.

In addition, in many applications, detectors can also benefit from other information, like depth when the camera is equipped with a depth sensor, or temporal information when a video sequence is input. However, it is still unclear how these information can be utilized by detectors, especially CNN-based detectors.

Given the observations above, one question comes up naturally: *what kind of extra features are effective and how they actually work to improve the CNN-based pedestrian detectors?* In this paper, we aim to answer this question and explore the characteristics of different extra features in pedestrian detection task. This paper contributes to:

- Firstly, we integrate extra features as input channels into CNN-based detectors. To investigate three groups of channel features: apparent-to-semantic channels, temporal channels and depth channels, extensive experiments are carried out on the KITTI pedestrian dataset [26].
- Then, we experimentally analyze both advantages and disadvantages of different kinds of channel features. Specifically, we quantify the improvement brought by different channel features and provide insight into the error sources.
- Moreover, a novel network architecture, namely HyperLearner, is proposed to aggregate extra features in a

multi-task learning manner. In HyperLearner, channel features are aggregated as supervision instead of extra inputs, and hence it is able to utilize the information of given features and improve detection performance while requiring no extra inputs in inference. We verify the effectiveness of HyperLearner on several pedestrian detection benchmarks and achieve state-of-the-art performance.

## 2. Related work

Traditional pedestrian detectors, extended from Viola and Jones paradigm [27], such as ACF [9], LDCF [22], and Checkerboards [35], filter various Integral Channels Features (ICF) [10] before feeding them into a boosted decision forest, predominating the field of pedestrian detection for years. Coupled with the prevalence of deep convolutional neural network, CNN-based models [17, 33, 2] have pushed pedestrian detection results to an unprecedented level. In [33], given region proposals generated by a Region Proposal Network (RPN), CNN features extracted by an ROI pooling layer [13] are fed into a boosted forest; while in Cai *et al.* [2], a downstream neural network architecture is proposed to preform end-to-end detection.

Integrating channel features of different types has been proved to be useful in many decision-forest-based pedestrian detectors. Prior work by Park *et al.* [23] embeds optical flow into a boosted decision forest to improve pedestrian detectors working on video clips. CCF [32] uses the activation maps of a VGG-16 [25] network pretrained on ImageNet [16] as channel feature, while Costea and Nedeveschi [8] utilize the heatmap of semantic scene parsing, in which detectors benefit from the semantic information within a large receptive field. However, the problem whether and how CNN-based pedestrian detectors can benefit from extra features still exhibits a lack of study.

### 3. Channel features for pedestrian detection

In this section, we empirically explore the performance boost when extra channel features are integrated into CNN-based detectors.

#### 3.1. Preliminaries

Before delving into our experiments, we first describe the dataset, evaluation metrics and baseline detector we use.

**KITTI dataset** We choose KITTI dataset [26] for channel feature analysis considering its possession of pedestrians of various scales in numerous scenes, as well as the information of adjacent frames and stereo data. KITTI contains 7,481 labeled images of resolution  $1250 \times 375$  and another 7,518 images for testing. The training set is further split into two independent set for training and validation following [5]. The person class in KITTI is divided into two subclasses: pedestrian and cyclist, both evaluated under PASCAL criteria [12]. KITTI contains three evaluation metrics: *easy*, *moderate* and *hard*, with difference in the *min.* bounding box height, *max.* occlusion level, *etc.* Standard evaluation follows moderate metric.

**Faster R-CNN** Our baseline detector is an implementation of Faster R-CNN [24], initialized with VGG-16 [25] weights pretrained on ImageNet [16]. It consists of two components: a fully convolutional Region Proposal Network (RPN) for proposal generation, and a downstream Fast R-CNN (FRCNN) detector taking regions with high foreground likelihood as input.

Since KITTI contains abounding small objects, we slightly modify the framework as [30] and [2]. Specifically, we adjust the number of anchors from 3 scales and 3 ratios to 5 scales and 7 ratios; besides, all `conv5` layers are removed to preserve an activation map of high resolution for both RPN and FRCNN.

We choose Faster R-CNN not only for its prevalence and state-of-the-art performance, but also generality: our observations should remain mostly effective when similar techniques are applied in other CNN-based pedestrian detectors.

#### 3.2. Introduction to channel features

In this section, we introduce the channel features we integrated into the CNN-based pedestrian detector. Based on the type of information they carry, the selected channel features for integration are divided into three groups: apparent-to-semantic channels, temporal channels and depth channels. Figure 2 provides a demonstration of all channels.

**Apparent-to-semantic channels** This group of channels includes ICF channel [10], edge channel, segmentation channel and heatmap channel. The information in these channels ranges from low-level apparent to high-level semantic.

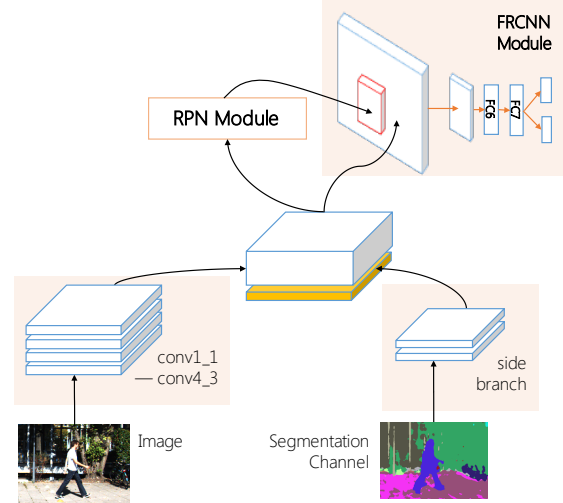


Figure 3. As described in Section 3.3, our Faster R-CNN for channel feature integration. The side branch takes channel features as input and generates channel feature representations before concatenated with `conv4_3`.

The ICF channel is a handy-crafted feature channel composed of LUV color channels, gradient magnitude channel, and histogram of gradient (HOG) channels, which has been widely employed in the decision-forest-based detectors [9, 22, 34]. Containing only colors and gradients within a local patch, ICF channel represents the most low-level but detailed information of an image.

The edge channel is extracted from the second and third layers of HED network [31]. Different with traditional edge detector such as Canny [3], the HED framework produces more semantically meaningful edge maps (see Figure 2). The edge channel is thus considered as a mid-level feature channel containing both detailed appearance as well as high-level semantics.

As in [20, 4], a fully convolutional network (FCN) is trained on MS-COCO dataset [18] to generate the semantic segmentation channel, where each pixel represents the probability of the category (*e.g.*, person and street) it belongs to. The segmentation channel carries higher-level semantic information, while still preserving some detailed appearance features, *i.e.*, the boundaries between objects of different categories. However, two closely-laid instances of same category cannot be distinguished from each other in the segmentation channel without contour of each instance.

Furthermore, to obtain a feature channel with only high-level semantics, we blur the segmentation channel into the heatmap channel. By doing so, the clear boundaries between objects of different categories are also removed and only high-level information of categories remains.

**Temporal channels** The temporal features (*e.g.*, optical flow [1] and motion [29]) have been proved to be benefi-

cial to traditional pedestrian detectors [28, 23] working on videos. To test their effectiveness in CNN-based framework, we extract optical flow channel as representative using temporally adjacent frames.

**Depth channels** With more and more depth sensors employed in intelligent systems such as robotics and automatic driving, the depth information available in these tasks becomes an alternative extra channel feature to boost detectors. Instead of using the sparse point clouds captured by laser radars, we turn to DispNet [21] to reconstruct the disparity channel from stereo images.

### 3.3. Integration techniques

We integrate channel features by creating a new shallow side branch alongside the VGG-16 main stream (see Figure 3). This side branch consists of several convolution layers (with kernel size 3, padding 1 and stride 1) and max pooling layers (with kernel size 2 and stride 2), outputting an 128-channel activation maps of  $1/8$  input size, which is further concatenated with activation map conv4\_3. The concatenated activation map is fed into the RPN and FR-CNN to perform detection.

We experiment different compositions of the side branch: the number of convolution layers and the initial weights (*i.e.*, a random gaussian kernel, or pretrained weights). The technique we employed to pretrain the side branch is to train a Faster R-CNN detector which completely relies on the side branch and initialize the side branch with the weights from this network.

	Model		Pedestrian		
	#Convs	Init. W.	Mod	Easy	Hard
O	N/A	N/A	68.96	73.33	60.43
A	2	random	<b>70.80</b>	<b>78.15</b>	<b>62.16</b>
B	1	random	70.40	75.17	61.92
C	2	pretrained	69.92	77.33	61.65

Table 1. Detection improvement by integrating channel features on KITTI validation set. Model “O” is our baseline detector. “#Convs” means the number of convolution layers in the side branch. “Init. W.” denotes initial weights for the side branch. The input images are not enlarged.

Summarised in Table 1, all integration methods improve the baseline Faster R-CNN detector in KITTI validation set on both classes across all three metrics. Nevertheless, the model with two extra convolution layers outperforms the model with only one extra convolution layer. A pretrained side branch does not perform well when further assembled with the VGG-16 network. When probing the network, we find that the model with pretrained weights tend to “rely” more on the sidebranch, (*i.e.*, activation map produced by side branch has much greater value than the

Model	Recall			
	(0, 80]	(80, 160]	(160, inf]	all scales
Baseline	21.3%	87.6%	96.8%	70.0%
+Segmentation	35.6%	88.2%	96.8%	74.0%

Table 2. Recall comparison at 70% precision between baseline and segmentation channel at different pedestrian heights. The results are based on 1x scale.

main stream). Given the fact that the side branch was pre-trained to perform detection independently, this imbalance may be a cause accounting for the performance degradation. Based on the analysis, we use two convolution layers with random Gaussian initialization in all future experiments.

### 3.4. Comparison and analysis

We conduct experiments on two input scales ( $1x$  and  $2x$ ). Table 3 summarizes the results. For a fair comparison, a controlled experiment in which the original image is used as input of the side branch is also included.

In general, models integrated with extra channel features show improvement over the baseline. The experiment using original image as extra input shows nonobvious improvement, which confirms that the performance gain is indeed attributed to channel feature integration. Among all channel features, ICF channel shows least contribution to the detection performance in both scales. We conjecture the reason is that in deep convolutional networks, CNN features are more discriminative than hand-crafted features like HOG.

Recall the two major challenges for pedestrian detection: hard negative samples and the individual localization. Through detailed analysis, we demonstrate how CNN-based detectors can benefit from extra channel features to overcome these problems.

**$1x$  experiments** In  $1x$  experiments, channels that carry more semantic information show better performance. As shown in Table 3, detectors with segmentation channel and heatmap channel bring most significant improvement to the detector. In accord with our previous hypotheses, the detectors utilize the semantic context provided by extra channel features to discriminate pedestrian of low resolution from hard negative samples.

Table 2 provides the recall comparison at certain precision rate (70%) between models with segmentation channel and the baseline model for pedestrians of different sizes. All pedestrians are divided into four groups based on their heights in pixel. Leading absolute 4% recall rate on average, the detector with segmentation channel performs significantly better in recall for small pedestrians (less than or equal to 80 pixel in height).

**$2x$  experiments** In  $2x$  experiments, model with only high-level semantic information but no low-level apparent features (*i.e.* the heatmap channel) fails to produce consistent

Model	Pedestrian 1x Input			Improvement				Pedestrian 2x Input			Improvement			
	Mod	Easy	Hard	Mod	Easy	Hard	Avg	Mod	Easy	Hard	Mod	Easy	Hard	Avg
Fr-RCNN* [24]	59.29	64.53	53.01	-	-	-	-	71.05	76.00	62.08	-	-	-	-
MS-CNN [2]	68.37	73.70	60.72	-	-	-	-	72.26	76.38	64.08	-	-	-	-
Our Baseline	68.96	73.33	60.43	-	-	-	-	71.21	77.73	62.19	-	-	-	-
+ Original img	68.63	76.61	60.45	-0.33	+3.28	+0.02	+0.99	71.33	76.72	62.17	+0.12	-1.01	-0.02	-0.30
+ ICF	68.40	73.56	60.20	-0.56	+0.23	-0.23	-0.19	71.80	77.40	62.79	+0.59	-0.33	+0.60	+0.29
+ Edge	69.49	76.28	60.89	+0.53	+2.95	+0.46	+1.31	72.34	78.32	63.28	+1.13	+0.59	+1.09	+0.94
+ Segmentation	<b>70.80</b>	<b>78.15</b>	<b>62.16</b>	<b>+1.84</b>	<b>+4.82</b>	<b>+1.73</b>	<b>+2.80</b>	<b>72.54</b>	<b>78.49</b>	<b>63.61</b>	<b>+1.33</b>	<b>+0.76</b>	<b>+1.42</b>	<b>+1.17</b>
+ Heatmap	70.33	78.03	61.75	+1.37	+4.70	+1.32	+2.46	71.39	77.64	62.34	+0.18	-0.09	+0.15	+0.08
+ Disparity	70.03	77.74	61.48	+1.07	+4.41	+1.05	+2.18	71.72	77.52	62.47	+0.51	-0.21	+0.28	+0.19
+ Optical Flow	69.39	77.07	60.79	+0.43	+3.74	+0.36	+1.51	71.13	76.85	62.24	-0.08	-0.88	+0.05	-0.25

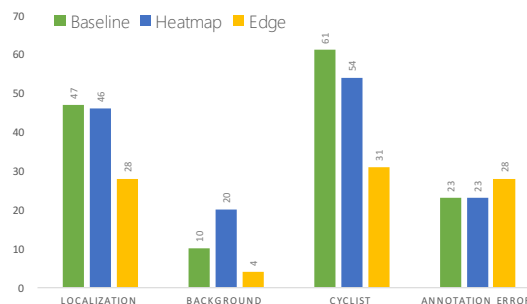
Table 3. Channel features comparison on KITTI validation set. We list improvement across all three KITTI metrics as well as the average. \*: Our reproduced Faster R-CNN with same parameters as in [24]. The baseline is a re-implementation of Faster RCNN pipeline, consisting of slight differences with the basic Faster RCNN (See Section 3.1).

improvement over the baseline model compared to the  $1x$  experiments. Nonetheless, channel features with both high-level semantic and low-level apparent information (edge channel and segmentation channel) outperforms other channels. A possible explanation for this is that when it comes to large input scale, low-level details (*e.g.*, edge) will show greater importance in detection. To further explore this phenomenon, we randomly sampled  $1/4$  of images (about 800) from validation set and collected false positive statistics at 70% recall rate, as shown in Figure 4(a). While in Figure 4(b), we also count top-200 false positives in the validation set and show the fractions of each error source. Not only inhibiting false positives across all categories at a high recall, edge channel also contributes significantly to the localization precision. Integrated with the edge channel, detector lowers localization error rate by absolute 9% and 7% compared with the baseline and the detector with heatmap channel respectively. This proves that channel features with low-level apparent features (*e.g.*, boundaries between individuals and contours of objects) improve localization precision when the input image is of high resolution.

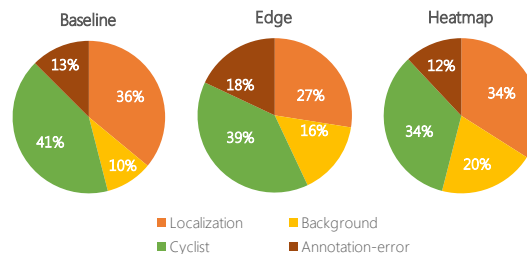
Besides, We witness noticeable improvement in  $1x$  when optical flow is integrated into the detector. Park *et al.* [23] also proved this effectiveness in decision-forest-based detectors with a detailed analysis. For the disparity channel, the results are very similar to the results of heatmap channel. To have an insight into this, we should notice that the relative value in a disparity map also serves as a “segmentation-like” channel (see Figure 2), while the absolute value has only limited effects compared to the deep convolutional features and the predefined anchors.

#### 4. Jointly learn the channel features

As observed above, integrating channel features into the network can boost our detector working on images of both low resolution and high resolution. With these channel fea-



(a) False positive sources at 70% recall rate



(b) Top-200 false positives sources

Figure 4. False positive analysis for baseline, edge channel and heatmap channel at  $2x$  scale. All false positives are categorized into four types: localization error, background classification error, cyclist classification error, and annotation error. Localization error is defined as non-matched detection bounding boxes which overlap with a groundtruth but  $iou < 0.5$ , while background error has no overlap with any groundtruth box. Cyclist error happens when a bounding box match cyclist groundtruth. Annotation error occurs when detection “matches” a *de facto* groundtruth which, however, is not annotated.

tures, we can narrow most of the gap between resolutions without introducing heavy computational cost brought by enlarging the input image, and push state-of-the-art forward.

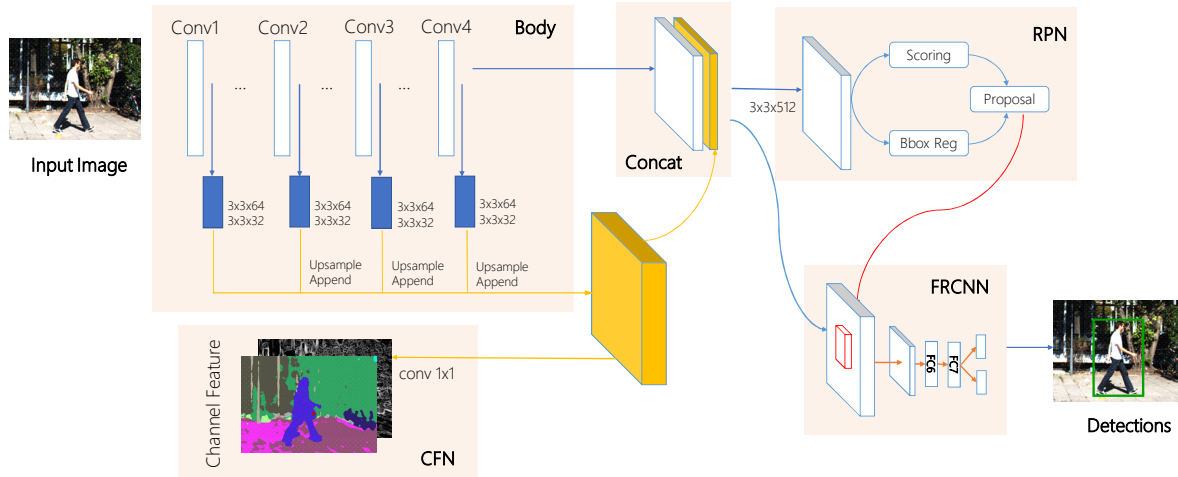


Figure 5. The proposed HyperLearner, which consists of 4 components: body network, channel feature network (CFN), region proposal network (RPN) and Fast R-CNN (FRCNN). HyperLearner learns representations of channel features while requiring no extra input in inference. Refer to Section 4.1 for details.

However, a brute-force integration method is computationally expensive with respect to the basic Faster R-CNN, given that the input channel feature usually requires extra computational cost. While many of the channel features comes from neural networks (*e.g.*, semantic segmentation and edge), it is natural to think of “teaching” our neural-network both channel features generation and detection. In the following section, we propose a new network structure to address the issue in a multi-task learning manner, namely, HyperLearner.

#### 4.1. HyperLearner

The HyperLearner framework is illustrated in Figure 5. As shown, our system consists of four components: the body network for activation map generation, a channel feature network (CFN), a region proposal network (RPN) and a Fast R-CNN (FRCNN) network for final detection task.

From the very left, the entire image is forwarded through multiple convolution layers to generate the hierarchical activation maps. We first aggregate activation maps and make them into a uniform space, namely aggregated activation map. Aggregating activation maps from multiple level has been proved to be useful and important in many computer vision tasks [15, 31] for its ability to collect rich hierarchical representations. This aggregated map is then fed into the channel feature network (CFN). CFN is a feed-forward fully convolutional network (FCN) for channel feature prediction. Unlike Faster R-CNN, RPN and FRCNN do not only take the output of the last convolution layer ( $\text{conv4}_3$ ) as input. Instead, the aggregated activation map is also fed into the RPN, as well as FRCNN. By sharing the same aggregated activation map, the RPN and FRCNN are able to benefit from the representations CFN learned.

**Aggregated activation map** The body network takes the raw image, of shape  $3 \times H \times W$ , as its input, and outputs several activation maps. In our experiments, the body network is a VGG-16 [25] network (without  $\text{conv5}_1$  to  $\text{conv5}_3$ ) initialized with the weights pretrained on ImageNet [16]. We extract the activation maps from layer  $\text{conv1}_2$ ,  $\text{conv2}_2$ ,  $\text{conv3}_3$  and  $\text{conv4}_3$ . Due to the pooling layer in the network, these maps are of different size and number of channels. We add two convolution layers after each map and keep their numbers of output channels same (32 in all our experiments). The high-level maps are then upsampled to the same size as the first activation map. Finally, they are concatenated together to form the aggregated activation map.

**Channel Feature Network (CFN)** The CFN directly takes the aggregated activation map to generate the predicted channel feature map through a fully convolutional structure. This map is typically of the same shape as the raw image. For example, the predicted channel feature may be a semantic segmentation map of several categories, or an edge detection map like HED Network [31].

**Region Proposal Network (RPN) and Fast-RCNN (FRCNN)** We build the RPN and FRCNN using the same structure as proposed in [24]. RPN and FRCNN now take both last convolutional activation map in the VGG16 network ( $\text{conv4}_3$ ) and the aggregated activation map from the body network as the inputs. The proposals generated by RPN are then fed into FRCNN to perform final detection.

#### 4.2. Training Details

**Loss Function** During the training phase, besides the raw image and groundtruth bounding boxes for standard Faster

R-CNN framework, the HyperLearner also takes a channel feature map as its supervisor, which is typically generated by another CNN (e.g., semantic segmentation and edge). To address the channel feature learning, we introduce a new pixel-level loss. Denote the feature map predicted by the CFN as  $C_{x,y}$ , and the supervisor map as  $S_{x,y}$ . The loss is computed by:  $\frac{1}{H \times W} \sum_{(x,y)} \ell(S_{x,y}, C_{x,y})$ ,

where  $H$  and  $W$  represents the size of the feature map and  $\ell$  is a loss function for a single pixel. In binary probabilistic maps, like edge map, cross-entropy loss is used, given by:  $\ell(p, q) = \beta_{x,y}(-p \log q - (1-p) \log(1-q))$ , where  $\beta$  is a weight function to balance the positive labels and negative labels. If  $S_{x,y} > 0.5$ ,  $\beta = 1 - |S_+|/|S|$ ; otherwise,  $\beta = |S_+|/|S|$ , where  $|S_+| = \sum \mathbf{1}[S_{x,y} > 0.5]$ . For multi-class probabilistic maps, like segmentation map, cross-entropy loss is used. For other tasks, MSE loss is used.

The final loss for the network is thus computed by:  $\mathcal{L} = \mathcal{L}_{\text{CFN}} + \lambda_1 \mathcal{L}_{\text{RPN}_{\text{cls}}} + \lambda_2 \mathcal{L}_{\text{RPN}_{\text{bbox}}} + \lambda_3 \mathcal{L}_{\text{FRCNN}_{\text{cls}}} + \lambda_4 \mathcal{L}_{\text{FRCNN}_{\text{bbox}}}$  where the last four component remains the same as Faster R-CNN [24]. In all our experiments, we set all  $\lambda_i = 1$ .

**Multi-stage training** The aggregated activation map acts as an important role in the framework, which must be carefully trained. We employs a pragmatic multi-stage training methods, making the whole training process splitted into four stages.

In the first stage, only CFN is optimized. In detail, we fix parameters of all pretrained convolution layers in the body network (conv1\_1 to conv4\_3), and drop all RPN and FRCNN layers to train the CFN. In the second stage, we fix the whole body network (including the convolution layers for aggregating activation maps) and CFN, and train only RPN. Then in the third stage, body network, CFN and RPN are all fixed; only FRCNN component is optimized. While in the final stage, all layers are jointly optimized.

Acrossing all stages, in the optimization of the FRCNN, we treat region proposals coordinates from RPN as fixed value and do not back-propagate the gradient.

## 5. Experiments and results

The performance of HyperLearner is evaluated across multiple pedestrian datasets: KITTI [26], Caltech Pedestrian [11], and Cityscapes [6]. The datasets we chose cover most of the popular ones in pedestrian detection task.

One may also notice that our body network an implementation of HyperNet proposed in [15]. Thus, we implement a control experiment where the CFN is removed as a typical HyperNet setting. That is, the body network keeps its side branches for aggregated activation map, but it does not learn from any extra supervision.

Model	1x input			2x input		
	Mod	Easy	Hard	Mod	Easy	Hard
Fr-RCNN* [24]	59.29	64.53	53.01	71.05	76.00	62.08
MS-CNN [2]	68.37	73.70	60.72	72.26	76.38	64.08
Baseline	69.80	74.37	61.20	71.73	77.84	62.30
HyperNet	69.72	76.91	61.10	72.23	77.96	63.43
+Segmentation	71.15	79.43	62.34	72.35	79.17	62.34
+Edge	71.25	78.43	62.15	72.51	78.51	63.24

Table 4. Results on KITTI validation set, the model HyperNet refers to the HyperLearner without CFN. Evaluation follows moderate metric in KITTI.

\*: Fr-RCNN follows setting as [24] while baseline model is Faster-RCNN with slightly different parameters. See also Table 3.

### 5.1. KITTI Dataset

We evaluated the performance of HyperLearner with two kinds of feature supervision: edge and semantic segmentation. These two kinds of channel features have been proved to be effective when directly integrated into the Faster R-CNN framework (see Section 3.3). The results on the validation set of KITTI dataset is illustrated in the Table 4.

In experiments on 1x scale, we notice great performance improvement when our HyperLearner is jointly learned from an edge detection network or a semantic segmentation network compared to the Faster R-CNN baseline and the HyperNet. The quantitative analysis is consistent with the experiments in Section 3.3 where we directly integrate them as an extra input into the network through a branch network.

In experiments on 2x scale, HyperLearner as well as HyperNet make clear improvement. Based on former analysis, when the input image is of high resolution, the introduction of channel features with low-level details could benefit the detector. In HyperNet setting, side branches of the body network act as an multi-level feature extractor, and therefore such kind of improvement is expected.

As a transfer learning application, HyperLearner successfully boost a CNN-based detector using features learned by other networks with different architecture and trained for other tasks. From another perspective, HyperLearner offers an alternative way to perform feature learning in such CNNs and showed noticeable improvement. Based on the results in Table 4 and 5, it is safe to conclude that HyperLearner actually utilizes the extra supervision from channel features to generate a better hyper-feature extractor, especially for the detection task.

### 5.2. Cityscapes dataset

The Cityscapes dataset [6], is a large-scale dataset for semantic urban segmentation which contains a diverse set of stereo video recordings from 50 cities. It consists of

2,975 training and 500 validation images with fine annotations, as well as another 20,000 training images with coarse annotations. The experiments are conducted on the fine-annotated images. Compared with former standard datasets, Cityscapes possesses meticulous detection labeling (pixel-level), as well as fine semantic segmentation labeling.

As mentioned, the Cityscapes dataset provides pixel-level semantic segmentation labeling, so instead of using segmentation model pretrained on MS-COCO dataset, we directly address the multi-task learning by employing pixel-level segmentation labels as supervisor (*i.e.*, our HyperLearner jointly learns pedestrian detection and semantic segmentation). During training, we only use segmentation labels for “person”. As shown in Table 5, we also witness significant improvement over the Faster R-CNN baseline and HyperNet.

### 5.3. Caltech dataset

The Caltech dataset [11] is also a commonly used dataset for pedestrian detection evaluation. It consists of 2.5 hours 30Hz VGA video recorded from a vehicle traversing the streets of Los Angeles, USA. Detection results are evaluated on a test set consisting of 4024 frames.

Zhang *et al.* [34] conducted a detailed survey and provided a refined groundtruth labeling on Caltech dataset. Our experiments is completely based on this new labeling (both training and testing). HyperLearner achieves state-of-the-art performance on the test set. Figure 7 shows the detailed comparison of HyperLearner, the Faster R-CNN baseline and other methods.

## 6. Summary

In this paper, we integrated channel features into CNN-based pedestrian detectors, specifically, ICF channel, edge channel, segmentation channel and heatmap channel (apparent-to-semantic channel); optical flow channel (temporal channel); disparity channel (depth channel). Our quantitative experiments show semantic channel features can help detectors discriminate hard positive samples and negative samples at low resolution, while apparent channel features inhibit false positives of backgrounds and improve localization accuracy at high resolution.

To address the issue of computational cost, we propose a novel framework, namely HyperLearner, to jointly learn channel features and pedestrian detection. HyperLearner is able to learn the representation of channel features while requiring no extra input in inference, and provides significant improvement on several datasets. From another point of view, HyperLearner offers an alternative way to perform feature learning in HyperNet-like CNNs in a transfer learning manner.

Model	540p input		720p input		Improvement	
	Speed	AP	Speed	AP	540p	720p
Baseline	130ms	74.97	240ms	86.89	-	-
HyperNet	140ms	74.30	250ms	86.67	-0.53	-0.22
Jointsegmap	140ms	<b>77.22</b>	250ms	<b>87.67</b>	<b>+2.25</b>	<b>+0.78</b>

Table 5. Results on Cityscapes validation set. The speed column shows the time each model needed to perform detection on a single image. The speed is tested on single NVIDIA TITAN-X GPU. We use all segmentation polygons labeled “person” to generate bounding boxes for the pedestrian detection task. Following the standard in Caltech dataset [11], all persons with (pixel-level) occlusion greater than 0.5 or of height less than 50 pixels are ignored. Furthermore, all polygons labeled “cyclist” or “person group” are also ignored.

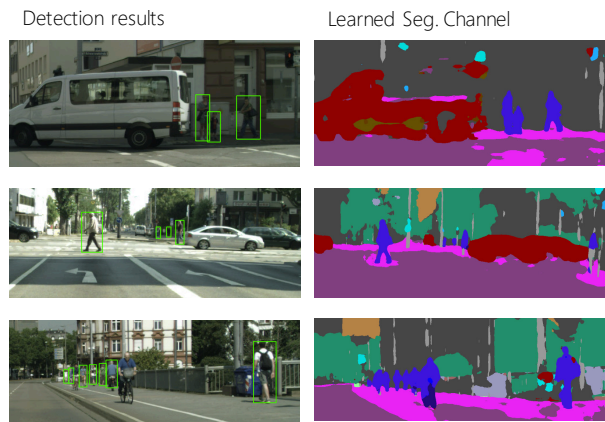


Figure 6. Results of HyperLearner on Cityscapes validation set. The left column shows our detection result, while the right column demonstrate CFN’s output learned from segmentation labeling.

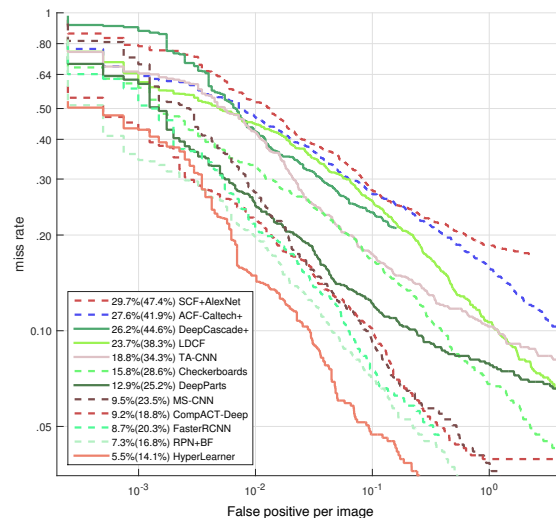


Figure 7. Detection quality on Caltech test set (reasonable,  $MR_{-2}^N(MR_{-4}^N)$ ), evaluated on the new annotations [34]. We achieve state-of-the-art results on both evaluation metrics.



## References

- [1] S. S. Beauchemin and J. L. Barron. The computation of optical flow. *ACM computing surveys (CSUR)*, 27(3):433–466, 1995. [3](#)
- [2] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. *arXiv preprint arXiv:1607.07155*, 2016. [2](#), [3](#), [5](#), [7](#)
- [3] J. Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986. [3](#)
- [4] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014. [3](#)
- [5] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun. 3d object proposals for accurate object class detection. In *Advances in Neural Information Processing Systems*, pages 424–432, 2015. [3](#)
- [6] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. [7](#)
- [7] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. 05 2016. [1](#)
- [8] A. Daniel Costea and S. Nedeveschi. Semantic channels for fast pedestrian detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2360–2368, 2016. [2](#)
- [9] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1532–1545, 2014. [2](#), [3](#)
- [10] P. Dollár, Z. Tu, P. Perona, and S. Belongie. Integral channel features. 2009. [1](#), [2](#), [3](#)
- [11] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 304–311. IEEE, 2009. [1](#), [7](#), [8](#)
- [12] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. [3](#)
- [13] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015. [2](#)
- [14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. [1](#)
- [15] T. Kong, A. Yao, Y. Chen, and F. Sun. Hypernet: Towards accurate region proposal generation and joint object detection. *arXiv preprint arXiv:1604.00600*, 2016. [6](#), [7](#)
- [16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. [2](#), [3](#), [6](#)
- [17] J. Li, X. Liang, S. Shen, T. Xu, and S. Yan. Scale-aware fast r-cnn for pedestrian detection. *arXiv preprint arXiv:1510.08160*, 2015. [2](#)
- [18] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014. [3](#)
- [19] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. Reed. Ssd: Single shot multibox detector. *arXiv preprint arXiv:1512.02325*, 2015. [1](#)
- [20] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015. [3](#)
- [21] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. *arXiv preprint arXiv:1512.02134*, 2015. [4](#)
- [22] W. Nam, P. Dollár, and J. H. Han. Local decorrelation for improved detection. *arXiv preprint arXiv:1406.1134*, 2014. [2](#), [3](#)
- [23] D. Park, C. L. Zitnick, D. Ramanan, and P. Dollár. Exploring weak stabilization for motion feature extraction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2882–2889, 2013. [2](#), [4](#), [5](#)
- [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. [1](#), [3](#), [5](#), [6](#), [7](#)
- [25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. [2](#), [3](#), [6](#)
- [26] A. G. P. L. R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. [1](#), [2](#), [3](#), [7](#)
- [27] P. Viola and M. J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004. [2](#)
- [28] S. Walk, N. Majer, K. Schindler, and B. Schiele. New features and insights for pedestrian detection. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pages 1030–1037. IEEE, 2010. [4](#)
- [29] H. Wang, M. M. Ullah, A. Klaser, I. Laptev, and C. Schmid. Evaluation of local spatio-temporal features for action recognition. In *BMVC 2009-British Machine Vision Conference*, pages 124–1. BMVA Press, 2009. [3](#)
- [30] Y. Xiang, W. Choi, Y. Lin, and S. Savarese. Subcategory-aware convolutional neural networks for object proposals and detection. *arXiv preprint arXiv:1604.04693*, 2016. [1](#), [3](#)
- [31] S. Xie and Z. Tu. Holistically-nested edge detection. In *Proceedings of IEEE International Conference on Computer Vision*, 2015. [3](#), [6](#)
- [32] B. Yang, J. Yan, Z. Lei, and S. Z. Li. Convolutional channel features. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 82–90, 2015. [2](#)

- [33] L. Zhang, L. Lin, X. Liang, and K. He. Is faster r-cnn doing well for pedestrian detection? *arXiv preprint arXiv:1607.07032*, 2016. [1](#), [2](#)
- [34] S. Zhang, R. Benenson, M. Omran, J. Hosang, and B. Schiele. How far are we from solving pedestrian detection? In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 2016. [3](#), [8](#)
- [35] S. Zhang, R. Benenson, and B. Schiele. Filtered channel features for pedestrian detection. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1751–1760. IEEE, 2015. [1](#), [2](#)