Neurocomputing 238 (2017) 420–432

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Neural features for pedestrian detection

Chao Li, Xinggang Wang, Wenyu Liu*

School of Electronics Information and Communications, Huazhong Univerisity of Science and Technology, Wuhan, PR China

ARTICLE INFO

Article history:
Received 12 April 2016
Revised 20 January 2017
Accepted 28 January 2017
Available online 8 February 2017

Communicated by Prof. Zidong Wang

Keywords:
Pedestrian detection
Neural features
Fully convolutional network

ABSTRACT

This paper presents a pedestrian detection approach that uses neural features from a fully convolutional network (FCN) instead of features manually designed. We train an AdaBoost detector per layer and compare the performance to find the optimal layer for this task. Combining results of multiple detectors can further improve the performance. In order to adapt the FCN to pedestrian detection task, we fine-tune it with bounding boxes labels. Using neural features generated by fine-tuned FCN, the log-average miss rate (MR) on Caltech pedestrian dataset is 18.79% by a single detector and 16.50% by combining two detectors. We also evaluate the proposed method on INRIA pedestrian dataset and the MR is 11.17% with a single detector and 9.91% through combining two detectors. The improved performance indicates that the proposed neural features are applicable to pedestrian detection task, due to their strong representation.

1. Introduction

Recently, the field of artificial intelligence has achieved significant progress and many relevant applications [1,2] have appeared. Pedestrian detection is a classical and important task in computer vision. There are many real-life applications, such as robotics, video surveillance and autonomous driving. And there has been lots of research on pedestrian detection over the past few years. Generally, a pedestrian detector involves three steps: (1) proposing bounding boxes that potentially contain pedestrians; (2) extracting features of the proposed bounding boxes; (3) judging whether a person exists within a bounding box using the extracted features. As is pointed out in the investigation of recent progress [3], most progress is due to the improvements in features. The frequently-used features in pedestrian detection include Histograms of Oriented Gradients (HOG) [4], LUV, Local Binary Pattern (LBP) [5], Haar-like features [6], texture [7], DWT [8,9], and entropy [10,11]. Dollár et al. [12] propose channel features, which consist of LUV, gradient histogram and gradient magnitude. The features in [12] are very effective and have been widely adopted in pedestrian detection. Some methods develop different operations on the channel features and achieve improvements in performance [13–15]. In addition, some methods take advantage of additional information such as optical flow [3,16,17] and stereo images [18] to improve the accuracy or efficiency. Since the features play such a vital role, it is significant to seek a more powerful feature with the potential to further optimize the performance.

A novel type of neural network named fully convolutional network (FCN) [19] has been proposed recently. Long et al. [19] remove the final classifier layer of classic convolutional networks and convert all fully connected layers into convolutional layers. The network is able to produce pixel-dense outputs for semantic segmentation task. The outputs of FCN are finer and more structural than traditional convolutional networks. It inspires us that exploiting the layer output of FCN is expected to improve the performance of detection task. Furthermore, FCN is trained in a pixel-to-pixel way for semantic segmentation, so it can be fine-tuned to other data using pixel-level labels. We choose FCN instead of other popular neural networks to extract features and fine-tune the parameters of FCN with a pedestrian dataset. We name the features produced by FCN as “Neural Features”. The neural features have stronger representation than hand-crafted channel features since the former are learned automatically from data. Besides, neural features are sophisticated and discriminative enough to judge whether the window contains a person. So we utilize the neural features for training detectors.

Pedestrian detection is a canonical case of object detection and the benchmark datasets are available. Besides, there have been many methods proposed to solve this detection problem, making it convenient to compare the performance of our method with other approaches for demonstrating the effectiveness of our proposed neural features.

In this paper, we have three main contributions. (1) We combine neural features from FCN with a traditional classifier, AdaBoost. The strong representation capacity of neural features

* Corresponding author.
E-mail address: liuwy@hust.edu.cn (W. Liu).

http://dx.doi.org/10.1016/j.neucom.2017.01.084
0925-2312/

boosts the pedestrian detection performance. (2) We propose a weakly supervised learning method to fine-tune the neural features. We change bounding boxes annotations to pixel-level labels to train the FCN models which are originally trained with semantic segmentation dataset. (3) Our method fuses the detection results of multiple detectors trained on neural features of different layers. The complementarity of these detectors contributes to the improvement of detection accuracy.

The rest of this paper is organized as follows. Section 2 gives a brief review of related work on pedestrian detection. Section 3 introduces the details of our neural features and its application in pedestrian detection. Section 4 evaluates the proposed approach on two public benchmark datasets. Finally, Section 5 gives the conclusion and future work.

2. Related work

Since the Viola–Jones framework [20], many fine hand-crafted features have been proposed. Dalal and Triggs propose Histogram of Gradients (HOG) [4], which proves to be effective for the detection task and has become a classical descriptor. Based on the HOG feature, some novel methods are proposed [3,13,17,21–26]. Takuya et al. [21] apply PCA to HOG and select a proper subset of PCA-HOG feature. It reduces the dimensionality of the feature vectors without lowering the performance. Felzenswalb et al. [23] propose Deformable Part Model (DPM) and utilize a variant of HOG [4] as feature. DPM is robust to occlusion and pose variations. So far, channel features combined with boosted trees may be the most popular method for pedestrian detection. Dollár et al. [12,25] propose a type of classical channel features that include LUV color channels, gradient magnitude and quantized orientations. With approximation algorithm [27], the channel features computed at a single scale can be used to approximate the features at nearby scales. Since the work [12], many methods have been developed to improve the capacity of the channel features. Some of them focus on the exploitation of filtration on channel features. For example, LDCF [13] derives filters from data to remove local correlations of channels. The filtered features are decorrelated and can improve the accuracy of the detector. Informed Haar [6] designs rectangular templates on pedestrian shape model and the produced Haar-like features are robust against occlusions. In addition, some approaches add more features, such as optical flow, LBP, covariance and so on. For instance, SpatialPooling(+) [17] utilizes these additional features and executes spatial pooling on them to achieve better robustness to noise. Katamari [3] makes use of context information and optical flow. Both two methods take advantage of additional features to enhance representation. Unlike these methods, we utilize neural features from FCN to replace the manually designed channel features.

Deep learning has revolutionized the computer vision. The methods based on Convolutional neural networks (CNNs) have set up new records in many vision tasks, such as image classification [28–30], object detection [31-33], and semantic segmentation [19]. Barros et al. [30] propose Multichannel Convolutional Neural Network (MCCNN) to recognize multimodal emotional state. They utilize a deep hierarchical feature to deal with spontaneous emotions, and integrate multiple modalities for non-verbal emotion recognition. Their results outperform other methods based on hand-crafted features. Some methods apply the deep learning structure to kernel applications, which use the deep architectures for extracting feature [34,35]. Many deep-learning based approaches appear in pedestrian detection [36–41]. Sermanet et al. [36] pre-train CNN in an unsupervised mode based on convolutional sparse coding for pedestrian detection. JointDeep [37] takes advantage of a deep learning framework to learn four components in pedestrian detection jointly: feature extraction, deformation handling, occlusion handling, and classifiers. To model the complex appearance variations of pedestrian, SDN [38] adds switchable layers in a convolutional neural network. The switchable layers are built with RBM variants. DeepCascade [39] takes advantage of the efficiency of cascade classifiers to improve the detection speed. It runs in real-time at 15 frames per second, which is the fastest deep learning based methods for pedestrian detection. Recently, Hosang et al. [40] train small CifarNet and large AlexNet [28] for pedestrian detection and demonstrate that they are both effective. When training AlexNet, they use the R-CNN ("Regions with CNN features") [42] recipe. They also demonstrate that pre-training on surrogate tasks and utilizing more data are favorable to the performance. DeepPed [41] is built upon the work of [40] and optimizes the stages of detection pipeline to improve the accuracy. Previous deep learning based methods often employ the convolutional neural networks to perform classification. However, it is extremely slow to run CNN detector in a sliding window fashion, so most of these methods use proposals from faster detectors as inputs. Our method merely uses neural network to extract features. We take some layer outputs of FCN as features and adapt them to pedestrian detection by fine-tuning FCN with weakly-supervised labels. We use the neural features to train an AdaBoost classifier. Since the AdaBoost is much faster than the CNN, we run it in a sliding window fashion without proposals.

Inspired by FCN, holistically-nested edge detection (HED) [43] is proposed. It produces an end-to-end edge detection system in an image-to-image training way that is similar to FCN. HED architecture also connects the last convolutional layer in each stage to side output layer, and thus yields multi-scale predictions. The multi-scale results provide a more accurate edge pixel localization and refine the prediction.

3. Neural features

In this section, we give details on how to make use of neural features in pedestrian detection task. In the training stage, FCN neural features of full images are produced and then the positive and negative samples extracted from the feature maps are utilized for training an AdaBoost classifier. In the detection stage, we produce neural features of testing images and construct feature pyramids by approximation, then the trained detector is applied in a sliding window manner. We give detailed explanations for each step in the following subsections, and introduce model fine-tuning and feature dimensionality reduction.

3.1. Extracting neural features from FCN

Convolutional neural networks can serve as feature extractors that produce feature hierarchies due to their multi-layer architecture. One of their advantages is that they produce features from raw pixels automatically rather than by hand-crafted way.

Fully convolutional networks are produced based on Caffe framework [45] by fine-tuning some popular ConvNets such as AlexNet [28] , VGG nets [29] and GoogleNet [46]. Some FCN models are publicly available on model zoo site of Caffe.[1] In this paper, we use three FCN models that are trained on PASCAL VOC segmentation challenge. They are all fine-tuned from the VGG-16 model. We call them "FCN-32s", "FCN-16s" and "FCN-8s" for short and they are illustrated in Fig. 1. In semantic segmentation task, FCN-32s model is firstly generated. Its stride at prediction layer is 32 pixels, which is coarse and limits the scale of detail. In order to refine the spatial precision, pool4 layer produces prediction with a down-sampling factor 16. By fusing the pool4 prediction and

[1] https://github.com/BVLC/caffe/wiki/Model-Zoo.

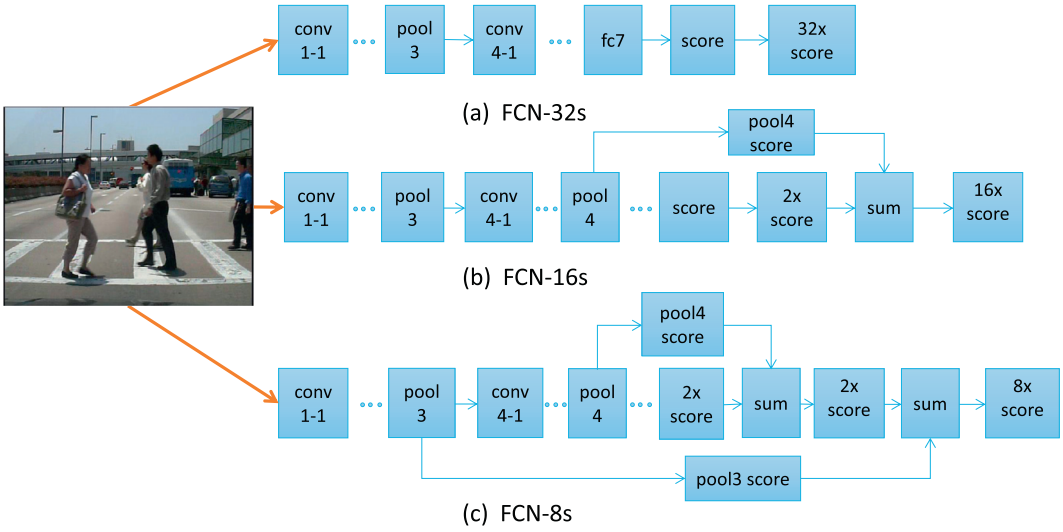


Fig. 1. Comparison of different FCN models. FCN-32s is the base model, and 16s model is generated by integrating *pool*4 layer's score with 32s model's score. FCN-8s model further combines the score of *pool*3 layer with that of 16s model. The input image is from Caltech pedestrian dataset [44]. The "32× score" means a score map that is enlarged by a factor of 32 through deconvolution layer, aiming to conform to the resolution of the original image.

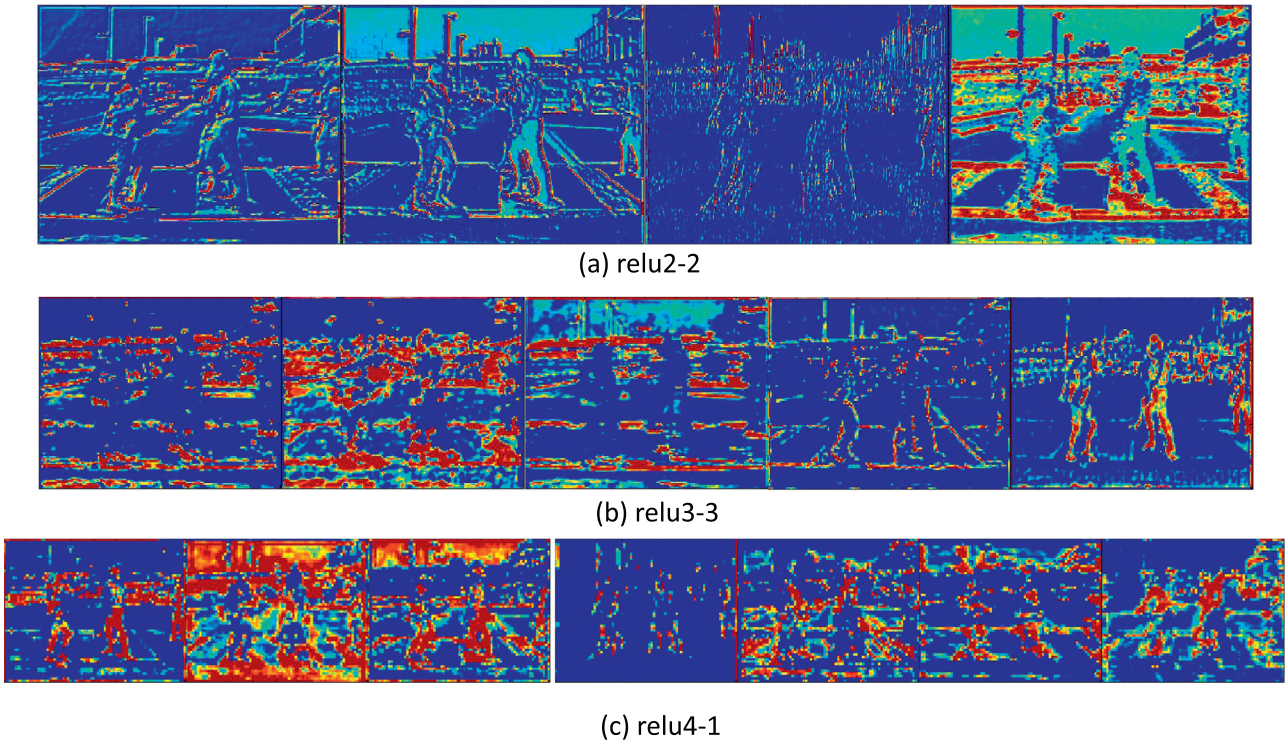


Fig. 2. Visualization of some layer outputs in FCN-32s model. The original color image is the same as Fig. 1. We use heatmap here to present better view. (a) shows *relu*2 – 2 layer output; (b) shows *relu*3 – 3 layer outputs; (c) shows *relu*4 – 1 layer outputs. From low layer to high layer, the channel number increases while the scale is diminishing. Low layer contains more simple structures while higher layer contains more semantic information.

the original score, FCN-32s model turns into FCN-16s model with finer outputs and better performance. Similarly, combing FCN-16s result with the prediction produced by pool3 layer, whose down-sampling factor is 8, can generate a more accurate model called FCN-8s.

Neural features in this paper refer to the layer outputs of FCN. Filters in different layers can extract different features. The low-layer features are usually associated with generic representations for visual problems while the high-layer features are more related to specific task and dataset. Fig. 2 shows some different layer outputs generated by FCN-32s model. We normalize the features within [0,255] and colorize them for a better view in Fig. 2.[2]

As is shown in Fig. 2, the spatial resolution of feature map gradually decreases while the channel number increases along the network. Specifically, the numbers of feature channels in *relu*2 – 2, *relu*3 – 3 and *relu*4 – 1 are 128, 256 and 512, respectively. And

[2] Throughout the paper, we only colorize the feature maps in Fig. 2 and all other images are shown normally.

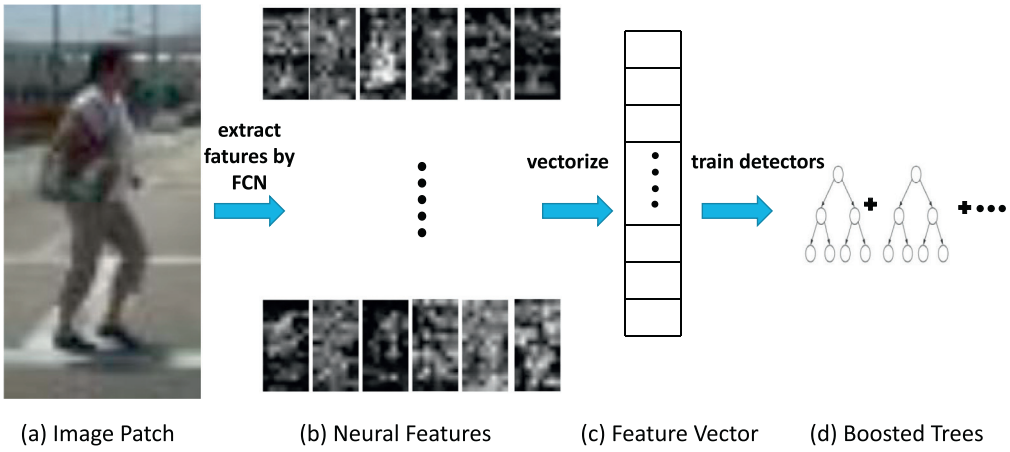


Fig. 3. Pipeline of training pedestrian detector with neural features. (a) A positive image patch extracted from the training images. (b) Neural features of the image patch are computed using FCN. (c) The neural features are vectorized to a fixed-length feature vector. (d) This feature vector is used to train the AdaBoost classifier.

their spatial scales with respect to the original image are 0.5, 0.25 and 0.125. As described above, the outputs of low layers are more relevant with low-level features and details than high layers. We can observe many obvious edge-like lines in $relu2-2$ layer, while in $relu4-1$ layer the response is more semantical.

Among these different layers' feature maps, some features are suitable for detecting the pedestrian while others are not. The key is to find the most appropriate layer output for pedestrian detection. So we need to extract different layer outputs and train classifiers using these features individually. Through comparing the performance of different detectors on the validation set, we can find the best representation for this task.

3.2. Training pedestrian detectors using neural features

The neural features produced by FCN are used to train an AdaBoost classifier for pedestrian detection. AdaBoost has been one of the most frequently used classifiers in this field. It consists of a great deal of "weak classifiers", and decision trees here serve as weak classifiers. Though a single tree is weak, the weighted sum of thousands of trees form a powerful classifier.

The training process is illustrated in Fig. 3. Take Caltech pedestrian dataset as an example. Positive and negative window samples are extracted from images and resized to a uniform size, which is 64×32 pixels in the Caltech dataset [44]. FCN takes in these window images and produces differen layer outputs, i.e., the neural features. In practice, we compute the convolutional feature map for an entire image and then crop windows from the shared feature map. It avoids performing a ConvNet forward pass for each window and reduces a great deal of computational cost. We can select one specific layer output to train a detector. The spatial resolution of window features is 16×8, which is a quarter of the size of the original window image. The feature channel number N is consistent with the kernel number of the convolution layer. Then neural features in a window are vectorized to a vector for training decision trees of AdaBoost classifier. In order to find the best feature, we train a detector for every layer output.

In the training stage, positive windows are extracted according to the ground truth. In order to augment the training data, we flip the positive windows horizontally. In the first pass of AdaBoost training, negative windows are extracted from images randomly. In the remaining bootstrapping passes, positive samples keep unchanged while negative samples are accumulated by the hard samples misclassified in previous stages. The hard mining makes the classifier focus on difficult instances, which improves the capacity of classifier.

3.3. Applying pedestrian detectors to neural features

After training detectors, we apply them to neural features extracted by FCN in detection stage. In Fig. 4, we plot the process of performing multi-scale detection. We extract neural features of original scale image and then take advantage of the power law to approximate neural features at nearby scales. Next, detection window slides on these scaled neural features maps. Finally, features extracted from the windows are judged by classifiers to produce predictions.

For a test image, its feature pyramids are constructed according to the power law by resizing the neural features of the original image to different scales. Dollár et al. [25] find that for some types of features in pedestrian images, the ratio of two scales' feature expectation follows a power function of the ratio of the scales. The formula is shown below.

$$f_\Omega(I_{s1})/f_\Omega(I_{s2}) = (s1/s2)^{-\lambda_\Omega} + \varepsilon \tag{1}$$

Ω denotes any low-level shift invariant function that transforms an image I to a channel image $\Omega(I)$. I_{s1} denotes the image I in scale $s1$. The λ here is the parameter of the power law, and the ε is the deviation from the power law. Every type of transformation has its specific λ_Ω.

The key point of the power law is that the finely sampled feature pyramids can be obtained from the coarsely sampled ones by extrapolation. It can spare a great deal of time by avoiding computing features for each scale. With the power law, the construction of feature pyramid can be very efficient. In our task, only the neural features of the original scale image are actually computed using FCN, while the feature of other scales are all approximated by the power law. The λ in the power law can be adjusted and we set it to 0 by default. The number of scales per octave is 8 and the max scale is 1.

We slide the detection window on the feature pyramids of test images. The neural features of windows are sent to the AdaBoost classifier to judge whether the windows contain persons.

3.4. Fine-tuning FCN models on pedestrian dataset

Besides using the original FCN models, we fine-tune FCN with a pedestrian dataset, which makes the model more suitable for pedestrian detection task.

As described above, FCN model is generated by adapting VGG net into the fully convolutional network. Long et al. [19] fine-tune all layers of VGG net, which guarantees a good performance on the PASCAL semantic segmentation task. Since the PASCAL VOC

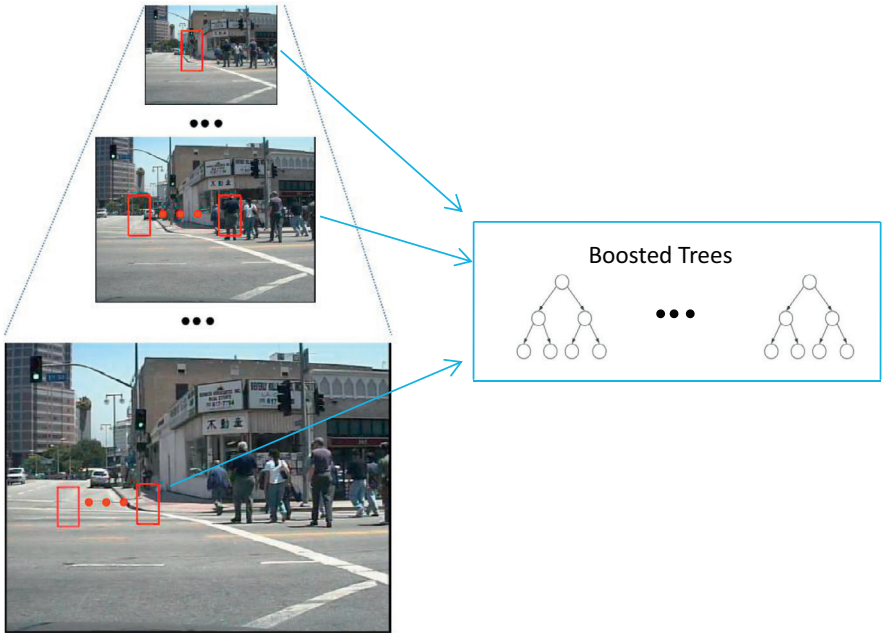


Fig. 4. Pipeline of multi-scale detection. Fixed-size window slides on the image pyramids, and then the neural features of these windows are judged by the boosted decision trees we have trained. In practice, we apply sliding window on the neural features maps rather than the images for efficiency.

dataset is very different from common pedestrian benchmarks, we speculate that the potential of FCN neural features on pedestrian datasets is not fully explored. It is very natural to fine-tune the model parameters with pedestrian datasets.

Through converting fully connected layers to fully convolutional layers, FCN can produce an output corresponding to the spatial structure of the input image. It provides a pixel-to-pixel learning mode whereby we can utilize pixel-level annotations to train a network.

We use the Caltech pedestrian dataset to fine-tune the FCN model since the dataset has a relatively large number of images. We take FCN-32s model as the initial model. In order to train the FCN, we need to produce pedestrian annotations in pixel level. Since the original ground truth of the Caltech dataset [44] is in the form of bounding box, we convert it to label image by setting the bounding box regions to value 1 and the background to 0. A bounding box contains some context around the person, so it includes pixels that don't belong to a person in the generated annotations. We utilize this weak supervision to fine tune the FCN models. There are two classes in our task, so we modify the output number of prediction layer to 2.

Fig. 5 shows the visual comparison between FCN model outputs and those of fine-tuned model. There are 21 classes in PASCAL VOC semantic segmentation task, so the number of the channel of FCN prediction layer is 21. The 16th channel is "person" class as is shown in Fig. 5. For the fine-tuned model, we also show the "person" channel of its final 2-channel output. The first two rows show that the fine-tuned FCN model gives a higher response to the low-resolution pedestrians than the original model does. It means that the neural features of fine-tuned model are superior to the original model in detecting small-size pedestrians. The third row illustrates an interesting phenomenon. The back of a motorcyclist obtains a strong response in fine-tuned model while little response in the original model. It demonstrates that the fine-tuned FCN model can better detect pedestrians in particular appearance. There are some pedestrians with particular appearance in the pedestrian dataset, and the fine-tuned model can learn how to recognize them. Though we only compare the last layer output of the two models, we can infer that the middle-layer outputs of the fine-tuned FCN model are also more related to the pedestrian data than the original model.

3.5. Feature dimensionality reduction

Neural features usually have a relatively high dimension which results in high space complexity, hence we apply the commonly used Principal Components Analysis (PCA) [47] algorithm to decrease the dimension of features. PCA can yield more discriminative, robust and compact representation than such standard descriptors as PCA-HOG [21] and PCA-SIFT [48].

We collect thousands of neural feature maps randomly. Every point in a feature map is a vector and we choose 2000 points stochastically in each feature map. We apply eigenvalue decomposition to the covariance matrix of these collected feature vectors, and the acquired eigenvectors form the eigenspace of neural features. Then we select top N eigenvectors and take them as the projection matrix of PCA. Given the projection matrix, neural features can be projected to a more compact representation. We precompute the eigenspace offline and save the top N eigenvectors as projection matrix.

4. Experiments

In this section, we evaluate our approach for pedestrian detection on two common datasets. We build our detection work based on Dollár's framework [49]. Neural features are extracted using the publicly available Caffe platform [45] on a single NVIDIA Titan X GPU with 12 GB memory.

4.1. Datasets

Our approach is evaluated on two datasets, one is Caltech pedestrian dataset and the other is INRIA pedestrian dataset [4]. We use the log-average miss rate (MR) over nine points between 10^{-2} and 10^{0} false positives per image to measure the result. It is calculated by the Dollár's toolbox [49].

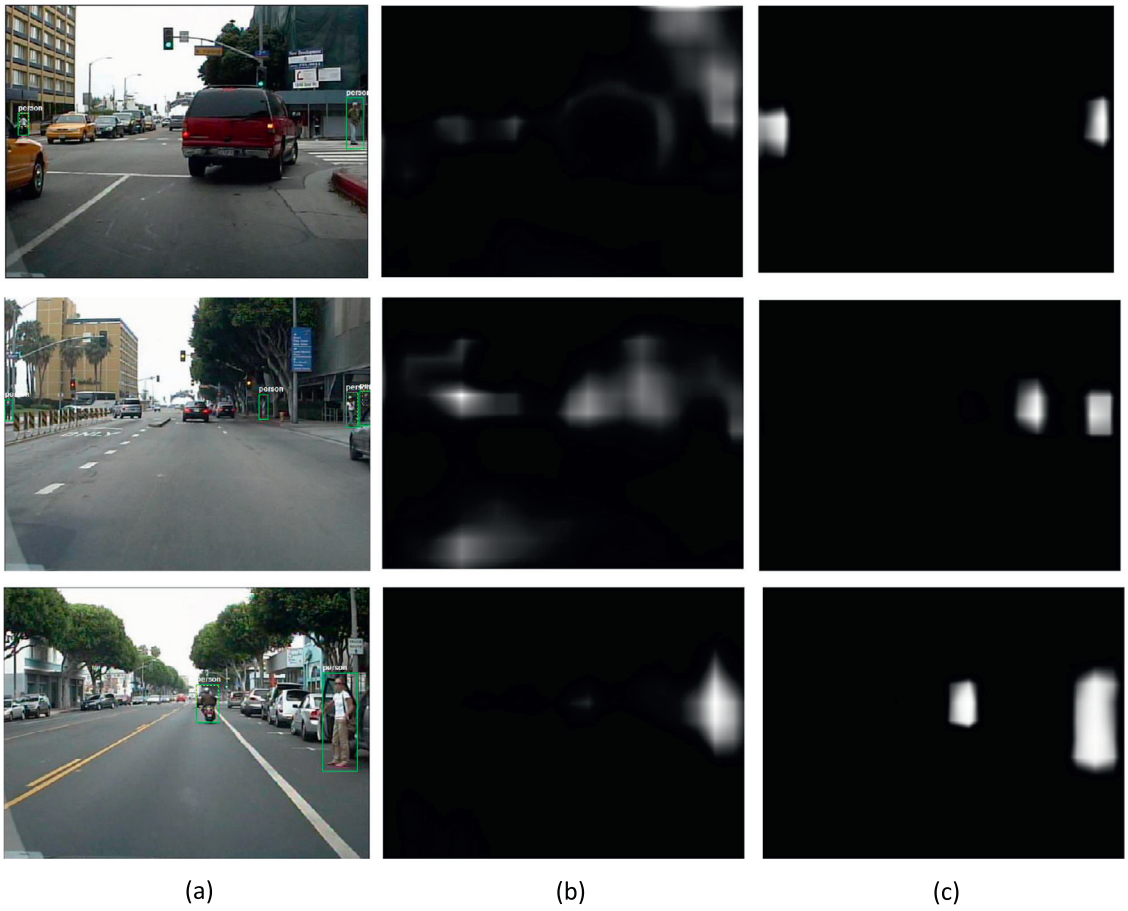


Fig. 5. Visualization of "person" channel of different FCN models outputs. (a) shows three images from Caltech pedestrian dataset. The green bounding boxes are ground truth of pedestrians. (b) shows the visualization of FCN-32s model's "person" channel. (c) shows the visualization of fine-tuned FCN-32s model's "person" channel.

4.1.0.1. Caltech pedestrian dataset. This is a large-scale, challenging dataset and has been serving as a standard benchmark for pedestrian detection. Collected from a vehicle driving through streets in an urban region, the Caltech dataset [44] consists of nearly ten hours of videos. There are a total of 350,000 bounding boxes and 2300 unique pedestrians annotated in the dataset. The sampling rate is 30 fps and the resolution of every frame is 640× 480. We use set00-04 as training set, set05 as validation set, and the remaining set06-10 as test set. In our experiment, we extract one sample out of every 10 frames in the training set and the validation set, and the sample interval is 30 for the test set. The performance is measured under the "Reasonable Set" [44] in which the height of pedestrians is taller than 50 pixels.

4.1.0.2. INRIA pedestrian dataset. INRIA pedestrian dataset [4] is among the most popular and oldest datasets for pedestrian detection. It has various backgrounds and high-quality annotations of pedestrians. Though the quantity is small, the diversity of images brings a stronger generalisation capacity than some other datasets [3]. Since the resolution of images in INRIA dataset is not fixed, we resize all images to 500× 500 pixels, which is the default input size of FCN model. The resolution of persons in INRIA is relatively large, so we set the extraction window size to 128 × 64 pixels.

4.2. Experiments with different models and layers

The output of every layer in FCN is a specific feature, so we can get hierarchical features efficiently by running the network. We utilize these different features to train AdaBoost classifiers for seeking the feature layer with the best performance. According to experience and our preliminary experiments, the performance variation of different layers is consistent among different stride versions of FCN models. So we choose FCN-32s model to compare the performance of different feature layers for simplicity.

Table 1
Log-average miss rate (MR) on the Caltech validation set with FCN-32s model when varying training feature. The "Size" means the scale of feature map about the original image.

Layer	Channel number	Size	MR(%)
$relu2-2$	128	1/2	62.26
$relu3-1$	256	1/4	36.88
$conv3-2$	256	1/4	32.51
$relu3-2$	256	1/4	31.04
$conv3-3$	256	1/4	32.59
$relu3-3$	256	1/4	**22.44**
$conv4-3$	512	1/8	48.51
$relu5-3$	512	1/16	78.24

Table 1 shows the preliminary results on the Caltech validation set when the feature varies. We use depth 4 decision trees in this experiment. It can be observed that the $relu3-3$ layer achieves the best result among these features. This phenomenon makes sense, because FCN model is trained for the 21-class PASCAL VOC semantic segmentation task, and the higher layers such as $conv4-3$ and $relu5-3$ are more related to the specific dataset and less general. Moreover, the big stride of these layers makes their output plane too coarse to be utilized for detection. At the other

Table 2
Log-average miss rate (MR) of different models' *relu*3 – 3 features on the Caltech validation set. The "tree-depth" refers to the depth of trees we used in the AdaBoost classifier.

	Tree-depth		
	3	4	5
FCN-32s	23.81	22.44	27.88
FCN-16s	22.47	25.58	28.59
FCN-8s	25.40	28.34	27.89

extreme, low layers such as *relu*2 – 2 produce features that are too general and not discriminative enough. The middle layers are therefore a tradeoff between general and semantical, and meanwhile they have suitable strides that can reflect fine space structure of image. So it is reasonable that the *relu*3 – 3 layer yields the best performance.

Now that we have found the best layer for this task, our next target is to find out the most appropriate model from the three stride versions of FCN. Table 2 shows the performances of detectors trained with *relu*3 – 3 layer features of various FCN models. It shows that the MR is almost at the same level among different models, especially the FCN-16s model and FCN-32s model. FCN-8s performs best in the original semantic segmentation task [19] due to its finer predictions. But in our task, FCN-8s model achieves a slightly higher MR than other models. The reason may be that the *relu*3 – 3 layer in FCN-8s is more relevant to the original PASCAL VOC semantic segmentation dataset. In FCN model, *relu*3 – 3 layer is linked to pool3 layer. As shown in Fig. 1, the pool3 layer in FCN-8s model connects with a prediction layer named "pool3 prediction", which has been fine-tuned to directly yield predication for semantic segmentation. In contrast, the pool3 layer in FCN-32s or FCN-16s only connects with layer *conv*4 – 1 and does not connect with any prediction layer directly, which makes it more general compared with the same layer in FCN-8s.

We also study the influence of tree depth of AdaBoost on the performance. We find that when the depth is three or four, the MR is low, while increasing the tree depth to five appears to damage the performance. It indicates that the depth-5 tree is too complex for our task and may result in overfitting. Since deeper trees need more data [13], it is appropriate to use depth-3 or depth-4 tree in our experiment.

Based on the experiments performed on the validation set, we choose FCN-16s tree-depth 3 model and FCN-32s tree-depth 4 model in the rest of the experiments. Their MR on the Caltech test set are 22.07% and 23.59%, respectively.

4.3. Data augmentation on positive training samples

In the above experiments, when training the AdaBoost classifier, we apply horizontal flip on positive samples to augment training data. We can translate or rotate feature maps of positive samples to produce more training samples. In our experiment, we translate the positive sample window maps in x direction as well as y direction. It results in nine map variants in total. These feature variants are shown in Fig. 6. Using different combinations of these feature variants, we can train more detectors that may outperform the original one. For example, we train a classifier using the first two variants in the first row of Fig. 6(b) and named it as "*V*1".

4.4. Dimension reduction

The channel number of *relu*3 – 3 layer is 256, which is relatively large. To reduce the feature dimension, we use PCA as described above. The number of dimension to keep is a compromise between performance and complexity. We select the top N eigenvectors in eigenspace to project the feature, and vary N to find the suitable number of dimension. Results are shown in Table 3. Note that we use the FCN-32s tree-depth 4 model here, and its MR on the Caltech test set is 23.59%.

Table 3
Performance of FCN-32s on the Caltech test set as feature dimension (N) varied. The original 256-dimension feature has a 23.59% MR.

Dimension	10	20	30	34	40	50	60
MR(%)	49.94	28.53	27.76	25.94	25.87	24.92	24.22
Eigen (%)	41.23	55.71	64.38	67.16	70.67	75.37	79.00

The first row in Table 3 denotes the remained dimension after operating PCA. The second row denotes the log-average miss rate (MR) that the detector obtained, and the last row shows the proportion of top N eigenvalues. As expected, increasing the remained dimensionality of feature vector can bring lower MR, but the effect gradually gets less noticeable. It indicates that the first several components of the PCA subspace are critical for expressing useful features for this task while the next ones are getting less and less useful.

4.5. Fusion of detections

In this section, we will use fusion to boost the accuracy of our detections. We employ two strategies to fuse detection results. The first one is fusing the results of detectors trained on different layer features, and the second one is incorporating detectors that are trained on the same layer feature but with different sample variants.

4.5.1. Fusing detection results of different layers

As mentioned above, different layer outputs denote different level features. Since we can train a specific classifier using one layer feature and obtain the corresponding detection result, it may improve performance if we combine these results in a certain way. Besides the best single detector *relu*3 – 3, we look for other layers that are diverse and valuable.

Detection results consist of detected bounding boxes and confidence scores. We give different weights to the bounding boxes scores produced by different detectors. In our experiment, the weight of score generated by *relu*3 – 3 detector is fixed to one and the weight of the other one detector's score is adjustable. After gathering these detections, we use non-maximal suppression (NMS) to get the final result. NMS is usually used to suppress multiple detections that are overlapped. For each pair of overlapped bounding boxes, the bounding box with a lower score will be suppressed if the ratio of the overlapping part that takes up the union is larger than a threshold θ. Therefore the fusion mechanism has to tune two parameters, one of which is the overlap threshold θ and the other the weight of the weaker detector score ω. The two parameters are optimised by grid search on the Caltech validation set.

We try to use some layers to fuse with *relu*3 – 3 detections. Results on the Caltech test set are reported in Table 4. We use FCN-32s model in this experiment. The first row shows MR of corresponding detectors, and the second row contains the MR fused with *relu*3 – 3 result. We observe that fusion on detections from different layers indeed improves the performance. It indicates that the results of detectors trained with diverse layers are complementary to a certain extent.

4.5.2. Fusing detection results of the same layer

The second strategy of fusion is combining the detection results of detectors that trained with the same layer neural feature.

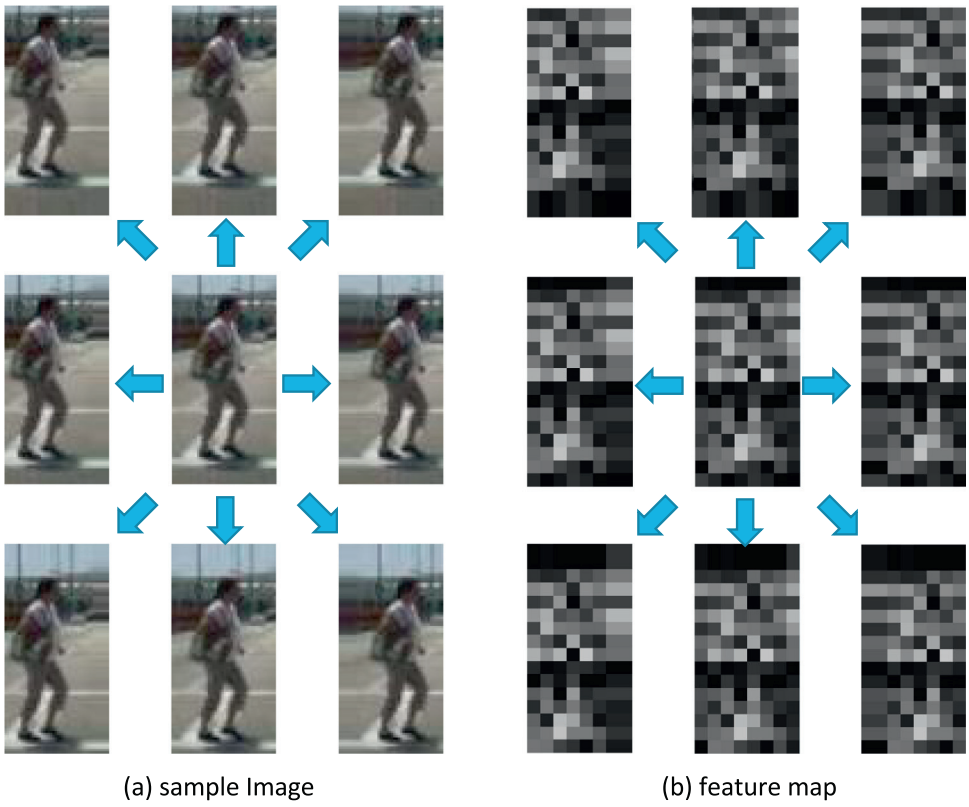


Fig. 6. Variants of positive sample. (a) shows nine variants in sample image. The image has been resized to 64× 32. The shift distance is four pixels, and arrows in the figure indicate the shift direction. (b) shows the feature maps of variants. The shift distance is one unit, which corresponds to four pixels in the original image.

Table 4
Log-average miss rate (MR) of fusing *relu*3 – 3 detector with other layers detectors on the Caltech test set, the MR of *relu*3 – 3 detector is 23.59%. We use FCN-32s model here.

	*relu*2 – 2 detector	*conv*3 – 2 detector	*conv*4 – 3 detector
Original detection	44.26	25.95	48.24
Fused with *relu*3 – 3	23.28	21.75	22.86

Table 5
Log-average miss rate (MR) of detectors on the Caltech test set.

	Norm	*V*1	Fused
FCN-16s	22.07	25.34	20.31
FCN-32s	23.59	25.77	21.85

Although using the same layer feature, the AdaBoost classifiers are trained with diverse positive samples variants, which leads to different detectors. The positive samples variants are illustrated in Fig. 6. We train a detector named V1 using the first two variants of positive samples and then fuse it with the original detector. The MR of these detectors on test set is shown in Table 5. We refer to the original detectors as "Norm", which are trained in the norm data augmentation way (flip the positive sample). The parameters θ and ω are optimised on the Caltech validation set.

It is observed that fusing multiple detectors that are trained with different feature variants, decreases the MR by about 2 % compared with the Norm detectors. Feature map variants have different biases in translation direction, so the detectors trained with them learn the ability to handle these variants. Fusion can exploit their complementarity and achieve a considerable improvement in performance.

4.6. Fine-tuning FCN models

We fine-tune the FCN models with Caffe framework on set00-05 of Caltech pedestrian dataset. Since the amount of data is critical for training neural networks, we reduce the sample interval to three for getting more training images. FCN models are fine-tuned in sequence. We firstly fine-tune the FCN-32s model and then the FCN-16s, and lastly the FCN-8s. These fine-tuned models are referred to as "FCN-Caltech". The FCN-Caltech-32s model is initialized from FCN-32s model.

4.6.0.1. Implementation details. The default parameters settings in FCN training are publicly available. The learning rate is 1e-10; the momentum is 0.99; weight decay is 0.0005 and batch-size is 1. We follow the default settings in our experiments. The original output number of prediction layer is 21 for PASCAL VOC semantic segmentation task, and we change it to 2 so that the network can be fine-tuned for predicting two classes (person and background).

4.6.0.2. Ground truth generation. FCN model is trained using pixel-level label, which provides every pixel's category. We transfer the original bounding box ground truth to label images. The person region is set to be 1 and the background region is 0.

We fine-tune the FCN-32s model using Caltech pedestrian training set. It takes about 33 h on a GPU for 180 k iterations. The training process is terminated when the loss converges and the visual

Table 6
Log-average miss rate (MR) of fine-tuned models on the Caltech test set.

Model	FCN-Caltech-32s	FCN-Caltech-16s	FCN-Caltech-8s
MR(%)	**20.02**	20.69	20.68
MR(Large-model-size)	19.30	19.40	**18.79**

Table 7
Log-average miss rate (MR) of FCN-Caltech-32s models on the Caltech test set. The three models are trained using different batch sizes.

Batch size	1	10	20
Iteration number	180 k	18 k	9 k
MR(%)	20.02	21.50	21.73

Table 8
Log-average miss rate (MR) of fusing detection results of FCN-Caltech models on the Caltech test set.

Model	MR(%)	Fused MR(%)
FCN-Caltech-32s	20.02	17.72
FCN-Caltech-8s	20.68	
FCN-Caltech-32s	20.02	**16.50**
FCN-Caltech-8s(Large)	18.79	

output of network barely changes. And then, we initialize the FCN-Caltech-16s model with the 32s model we have fine-tuned. The base learning rate is 1e-13 by default. The FCN-Caltech-16s model has a new prediction layer produced from pool4 layer. Then the FCN-Caltech-8s model is fine-tuned as the same training mode and initialized from the FCN-Caltech-16s model we have trained. Training the 16s and 8s model takes about another 26 h, respectively.

Table 6 shows the results of FCN-Caltech models on the Caltech test set. We use the *relu*3 – 3 layer output of these models as neural features to train detectors. The three fine-tuned models have similar performance. The FCN-Caltech-32s model obtains the lowest MR 20.02%, which is 2 % lower than the original best result 22.07% produced by FCN-16s model. It indicates that fine-tuning is effective and can adapt models to our task and data.

To verify that the batch size one is suitable for our task, we change the batch size to 10 and 20 while the total numbers of training images are kept unchanged, and the results are shown in Table 7. All the models are fine-tuned from FCN-32s model. It can be noted that the three fine-tuned models get similar performance and the model trained with batch size one actually works better, so we keep the batch size to one in our experiments.

4.7. Employing a large template

In this experiment, we attempt to use a large template when training the pedestrian detector. We enlarge the pedestrian template to 96 × 48 pixels instead of 64 × 32, so the spatial resolution of corresponding neural features becomes 24 × 12. With larger template, the neural features of a window become finer in spatial resolution, which makes it more discriminative. Since the template has been 1.5 times larger than the original one, the sliding window in detection stage is also enlarged. We must simultaneously enlarge the test images in order to find the small-size pedestrians.

The results are shown in the second row of Table 6. Though using the same neural features, the detector trained with large template can improve the performance of the former one. For example, there is a noticeable gain in FCN-Caltech-8s, which nearly decreases 2 % in MR. It indicates that using an appropriate large model size can boost the performance.

Table 8 contains the detailed results of fusing detection results of two FCN-Caltech models using NMS. Combining the FCN-Caltech-32s and FCN-Caltech-8s detection results can decrease the MR to 17.72%. When taking the detectors trained with large template into the consideration scope of fusion, we can further reduce the MR to 16.50%. The two fused detectors are: (1) the normal size one trained using FCN-Caltech-32s features, with MR being 20.02%; (2) the large template one trained using FCN-Caltech-8s features, with MR being 18.79%. Fusing two detectors brings a considerable improvement in performance than single detector.

Table 9
Log-average miss rate (MR) of different models on INRIA test set.

Model	FCN-32s	FCN-16s	FCN-8s
MR(%)	**14.14**	15.90	15.29

4.8. Comparison with other methods

We compare the performance of our method with other popular detection approaches on the Caltech dataset [44]. The detection results of other approaches are acquired from Caltech pedestrian detection dataset web.[3] The log-average miss rate (MR) is calculated by the same criterion as our method. The evaluation results are shown in Fig. 7.

Using the original FCN models, we get a 22.07 % log-average miss rate (MR) on this dataset by a single detector, outperforming the channel features methods, such as ACF [25] and LDCF [13]. Notice that the ACF [25] and LDCF [13] both use 2.5 times training data as much as ours. They sample image every 4 frames from Caltech training set while our interval is 10 frames. Our method also outperforms some CNN based approaches, such as Joint-Deep [37], SDN [38] and SCF+AlexNet [40]. Katamari [3] and SpatialPooling+ [17] are comparable to the performance of our basic method "NeuralFeatures", but both of them use many features such as HOG, LBP, spatial covariance, optical flow, etc. Through fusing two detectors' results, "NF Fusion" can decrease the MR to 20.31 % and outperform [3] and [17].

After fine-tuning the FCN models on Caltech pedestrian dataset, there is a considerable improvement in performance. "Fine-tuned NF" achieves a 20.02 % MR (not shown in Fig. 7) using a normal template and 18.79 % MR using a large template, outperforming approaches mentioned above. By fusing detections of the two fine-tuned models, "Fine-tuned Fusion" further decreases the MR to 16.50 %.

4.9. Results on INRIA pedestrian datasets

The image number of INRIA pedestrian dataset is much smaller than that of the Caltech dataset [44]. We evaluate the performance of different models on INRIA pedestrian dataset and show them in Table 9. It can be noticed that FCN-32s model is the best performer and we choose it as the default model in following experiments.

As for the dimensionality reduction, similar PCA method is applied to INRIA dataset [4]. A distinct effect is reflected in Table 10. When N is 32, we can obtain a good trade-off between accuracy and complexity. The MR of 32 dimension is lower than that of other dimensions shown in table. The reason may be that the components after 32 are potentially noisy for this task, especially for the 40–50 components.

Table 11 contains some fusion results on INRIA test set. Note that the log-average miss rate (MR) of *relu*3 – 3 detector is 14.14 %. We observe that the fusion brings a more impressive gain in performance than the Caltech dataset [44] in Table 4, especially for

[3] http://www.vision.caltech.edu/Image_Datasets/CaltechPedestrians.

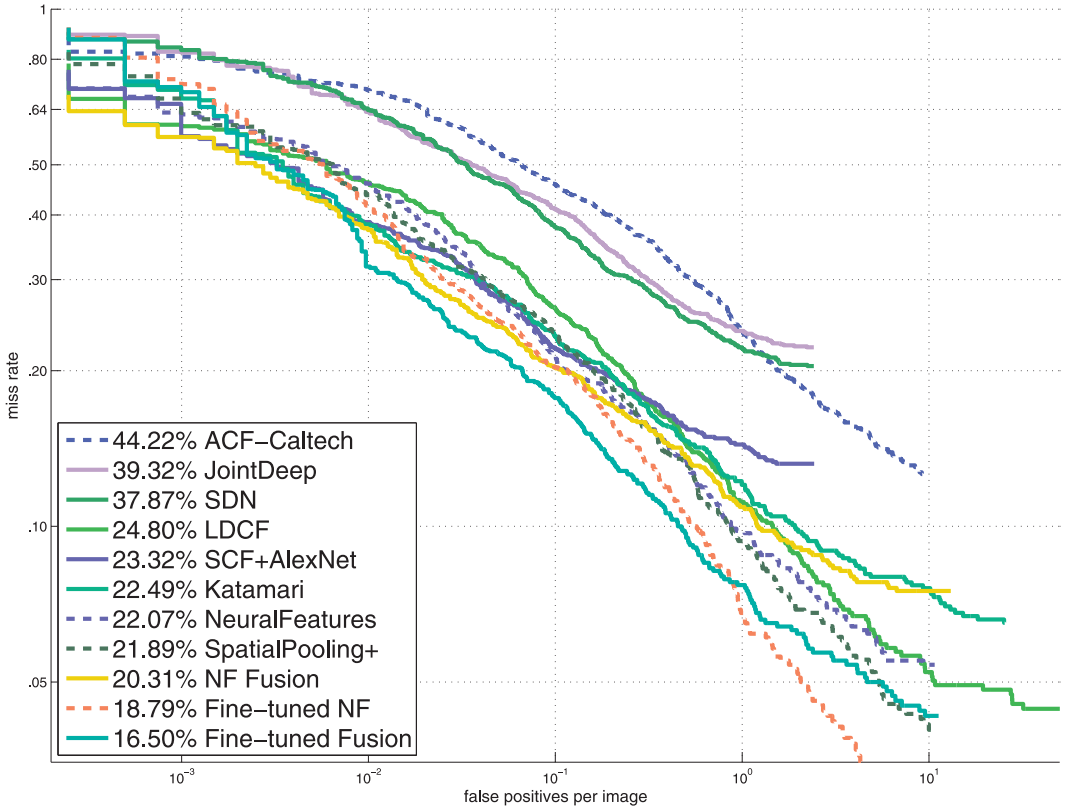


Fig. 7. Comparisons between our method and other popular pedestrian detection algorithms on the Caltech test set. "NeuralFeatures" is our basic method which uses neural features from FCN models. "NF Fusion" is the fusion of detectors which are trained with FCN neural features. While the "Fine-tuned NF" and "Fine-tuned Fusion" are similar to the previous two except that they utilize neural features from FCN-Caltech models.

Table 10
Performance on INRIA test set as dimension (N) varied. The original 256-dimension neural feature of FCN-32s has a 14.14% MR.

Dimension	10	20	30	32	40	50	60	120
MR(%)	24.35	17.45	20.12	**15.98**	17.66	19.24	16.24	16.47
Eigen (%)	35.12	50.87	61.08	62.72	68.13	73.12	77.11	90.16

Table 11
Log-average miss rate (MR) of fusing $relu3-3$ detector with other layers detectors on the INRIA test set.

	$relu2-2$ detector	$conv3-1$ detector	$conv4-3$ detector
Original detection	18.46	16.24	24.21
Fused with $relu3-3$	12.37	12.77	**11.97**

Table 12
Log-average miss rate (MR) of detectors on INRIA test set.

Detector	V1	V2	Fused
MR (%)	11.17	12.87	**9.91**

Table 13
Log-average miss rate (MR) of FCN-Caltech models on INRIA test set.

Model	FCN-Caltech-32s	FCN-Caltech-16s	FCN-Caltech-8s
MR (%)	**12.98**	13.54	13.68

the detector fused on $conv4-3$ and $relu3-3$. And when it comes to fusion in the same layer, we use detector $V1$ and another detector named "$V2$". $V2$ is trained with the eighth and ninth variants in Fig. 6, whose translation direction is opposite to $V1$. The results are shown in Table 12. Optimal performance 9.91 % can be achieved through fusion.

Since we have fine-tuned FCN models on Caltech pedestrian dataset, we want to see how they perform in INRIA dataset [4]. The results are shown in Table 13. FCN-Caltech models achieve better performance than the original models. It proves the generalization ability of our fine-tuned models.

Fig. 8 shows the performances of our approach on INRIA dataset [4], compared with some relevant methods. We show two of our detectors in the figure, the best single detector V1 and the detector fused V1 and V2. Their MR are 11.17 and 9.91 %, outperforming most approaches including ACF [25], LDCF [13] and SketchTokens

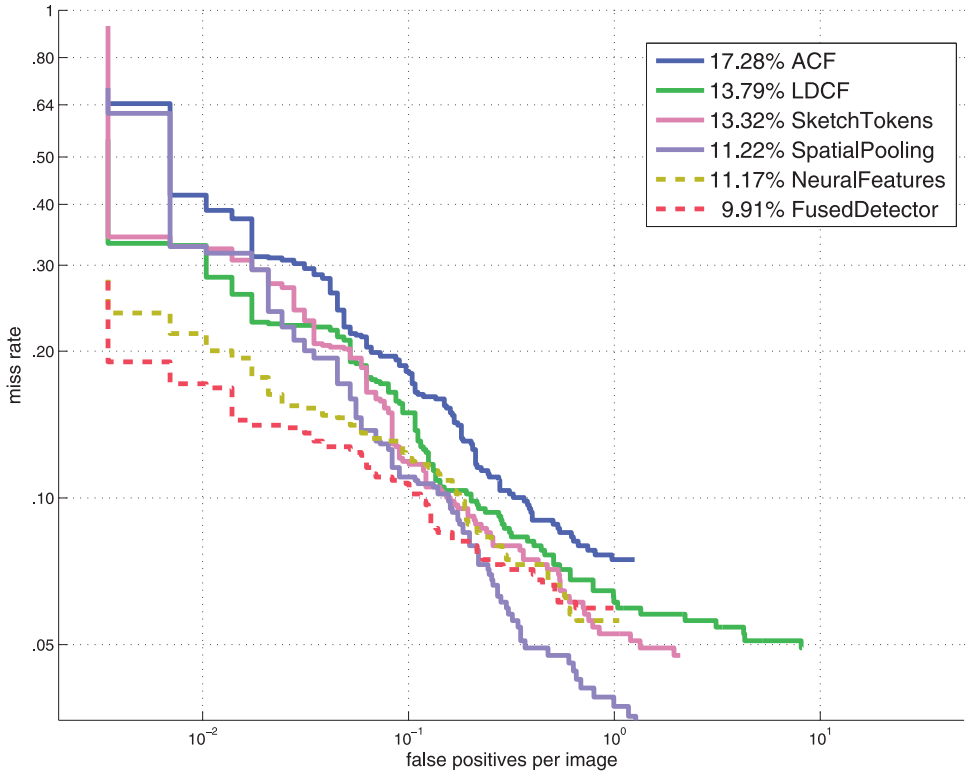


Fig. 8. Comparisons between our method and other popular pedestrian detection algorithms on INRIA. "NeuralFeatures" is the detector V1 and "FusedDetector" is the fusion of V1 and V2.

Table 14
The comparison of detection speed of NeuralFeatures with other detectors.

Method	SCF+AlexNet [40]	SDN [38]	SpatialPooling [17]	DeepPed [41]	NeuralFeatures
Time(s)	2.3	1–1.5	7.7	0.53	0.21

[50]. The exception is SpatialPooling [17], which is on par with our single detector but outperformed by our fused detector.

4.10. Time analysis

In this section, we analyse the time complexity quantitatively.

4.10.0.3. Training time. Training process roughly consists of feature extraction and classifier training. Extracting $relu3-3$ feature of a Caltech pedestrian image (640 × 480 pixels) takes about 0.053 s in a single GPU. For training classifier, it takes about 70 min to train depth-4 trees of boosting classifiers, and training depth-3 trees take about 110 min. Our machine has 12 cores and we use parallel processing.

4.10.0.4. Detection time. Similarly, detection includes two steps: extracting test image features and running detector on neural feature maps. The overall runtime of our approach to process a 640 × 480 image is about 0.21 s. The detection speed is summarized in Table 14, in which we list some other state-of-the-art detectors. SCF+AlexNet [40] relies on SquaresChnFtrs [3] to produce proposals, and proposing detections takes 2 s per image, which takes most of the detection time. SDN [38] uses a HOG-based approach to prune most candidate windows, and its runtime is about 1–1.5 s. Some detectors, such as SpatialPooling [17] and Katamari [3], are very slow since they combine many features to achieve a good performance. DeepPed [41] requires 0.53 s to process a single frame. Thus our method is faster in detection speed than these methods.

4.11. Discussion

Fig. 9 shows some examples of detection results. Our method can detect some small-size pedestrians. The false positives are usually from upright objects which have similar appearances with pedestrians. It is interesting that our detector regards the person in a billboard as pedestrian in the bottom of Fig. 9(c).

The advantage of our work is that we leverage the neural network to automatically learn a type of features which achieves excellent performance in pedestrian detection. This work outperforms most of traditional methods which use hand-crafted features. There are two major drawbacks in our work. Firstly, the dimension of our neural feature is relatively high (256) compared with some hand-crafted features, such as ACF. Secondly, our system is not trained in an end-to-end manner. The two components, the neural features extractor (FCN) and the AdaBoost classifier, are trained sequentially and cannot be optimized jointly. The interaction among them is not yet well explored.

5. Conclusions

In this work, we propose a pedestrian detection approach based on neural features derived from the fully convolutional network. Experimental results on two benchmark datasets demonstrate the effectiveness of neural features for pedestrian detection. Using pedestrian data that only has bounding boxes labels, we train the FCN models in a weakly-supervised way. The fine-tuned models are adapted to pedestrian detection task and the neural features produced by them can reduce the log-average miss rate (MR)

Fig. 9. Example detections of our method on the Caltech test data. Green rectangles are true positives while red rectangles show false positives. Detection scores are shown on the top of each box. The results are produced by the detector using FCN-Caltech-32s neural features. For a better view, please see original PDF file. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

significantly. In addition, we apply some fusion strategies whose results demonstrate that fusing two detectors can complement each other and enhance the performance. The experiments in this paper also indicate that a large template can make the neural features more discriminative and thus achieves a better result. The proposed method can be applied to pedestrian detection in automotive driving assistance systems. In future, we will focus on the optimization of performance and extend the method to detection of other objects.

Acknowledgments

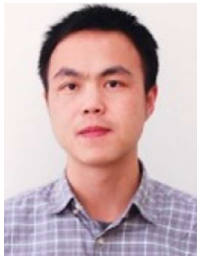
The authors would like to thank the anonymous reviewers for their valuable comments. This work was supported by the National Natural Science Foundation of China (Grant nos. 61503145 and61572207) and the CAST Young Talent Support Program.

References

[1] H. Drira, B.B. Amor, A. Srivastava, M. Daoudi, R. Slama, 3D face recognition under expressions, occlusions, and pose variations, IEEE Trans. Pattern Anal. Mach. Intell. 35 (9) (2013) 2270–2283.
[2] P.-W. Tsai, L.-H. Yang, J. Zhang, X. Xue, J.-F. Chen, J.-F. Chang, J.-S. Pan, An accuracy comparison between the time-series and the computational intelligence models on the taiwan-central america free trade agreements, ICIC Expr. Lett. 7 (9) (2016) 1925–1932.
[3] R. Benenson, M. Omran, J. Hosang, B. Schiele, Ten years of pedestrian detection, what have we learned? in: Proceedings of the Workshops on Computer Vision-ECCV, Springer, 2014, pp. 613–627.
[4] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition,, vol. 1, IEEE, 2005, pp. 886–893.
[5] X. Wang, T.X. Han, S. Yan, An HOG-LBP human detector with partial occlusion handling, in: Proceedings of the IEEE 12th International Conference on Computer Vision, IEEE, 2009, pp. 32–39.
[6] S. Zhang, C. Bauckhage, A. Cremers, Informed Haar-like features improve pedestrian detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 947–954.
[7] D.M. Gavrila, S. Munder, Multi-cue pedestrian detection and tracking from a moving vehicle, Int. J. Comput. Vis. 73 (1) (2007) 41–59.
[8] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, T. Poggio, Pedestrian detection using wavelet templates, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE, 1997, pp. 193–199.
[9] G.-S. Hong, B.-G. Kim, Y.-S. Hwang, K.-K. Kwon, Fast multi-feature pedestrian detection algorithm based on histogram of oriented gradient using discrete wavelet transform, Multimedia Tools Appl. (2015) 1–17.
[10] J. Li, W. Gong, W. Li, X. Liu, Robust pedestrian detection in thermal infrared imagery using the wavelet transform, Infrared Phys. Technol. 53 (4) (2010) 267–273.
[11] Q. Liu, J. Zhuang, J. Ma, Robust and fast pedestrian detection method for far-infrared automotive driving assistance systems, Infrared Phys. Technol. 60 (2013) 288–299.
[12] P. Dollár, Z. Tu, P. Perona, S. Belongie, Integral channel features., in: Proceedings of BMVC, vol.2, 2009, p. 5.
[13] W. Nam, P. Dollár, J.H. Han, Local decorrelation for improved pedestrian detection, in: Proceedings of the Advances in Neural Information Processing Systems, 2014, pp. 424–432.
[14] R. Benenson, M. Mathias, T. Tuytelaars, L. Gool, Seeking the strongest rigid detector, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 3666–3673.
[15] S. Zhang, R. Benenson, B. Schiele, Filtered channel features for pedestrian detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2015, pp. 1751–1760.
[16] D. Park, C. Zitnick, D. Ramanan, P. Dollár, Exploring weak stabilization for motion feature extraction, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 2882–2889.
[17] S. Paisitkriangkrai, C. Shen, A. van den Hengel, Strengthening the effectiveness of pedestrian detection with spatially pooled features, in: Proceedings of the Computer Vision–ECCV, Springer, 2014, pp. 546–561.
[18] R. Benenson, M. Mathias, R. Timofte, L. Van Gool, Pedestrian detection at 100 frames per second, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2012, pp. 2903–2910.
[19] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.
[20] P. Viola, M.J. Jones, D. Snow, Detecting pedestrians using patterns of motion and appearance, Int. J. Comput. Vis. 63 (2) (2005) 153–161.
[21] T. Kobayashi, A. Hidaka, T. Kurita, Selection of histograms of oriented gradients features for pedestrian detection, in: Proceedings of the International Conference on Neural Information Processing, Springer, 2007, pp. 598–607.
[22] S. Walk, K. Schindler, B. Schiele, Disparity statistics for pedestrian detection: combining appearance, motion and stereo, in: Proceedings of the European Conference on Computer Vision, Springer, 2010, pp. 182–195.
[23] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, D. Ramanan, Object detection with discriminatively trained part-based models, IEEE Trans. Pattern Anal. Mach. Intell. 32 (9) (2010) 1627–1645.
[24] V.-D. Hoang, M.-H. Le, K.-H. Jo, Hybrid cascade boosting machine using variant scale blocks based hog features for pedestrian detection, Neurocomputing 135 (2014) 357–366.

[25] P. Dollár, R. Appel, S. Belongie, P. Perona, Fast feature pyramids for object detection, IEEE Trans. Pattern Anal. Mach. Intell. 36 (8) (2014) 1532–1545.
[26] S. Yao, S. Pan, T. Wang, C. Zheng, W. Shen, Y. Chong, A new pedestrian detection method based on combined HOG and LSS features, Neurocomputing 151 (2015) 1006–1014.
[27] P. Dollár, S. Belongie, P. Perona, The fastest pedestrian detector in the west., in: Proceedings of BMVC, vol.2, Citeseer, 2010, p. 7.
[28] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Proceedings of the Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
[29] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556 (2014).
[30] P. Barros, D. Jirak, C. Weber, S. Wermter, Multimodal emotional state recognition using sequence-dependent deep hierarchical features, Neural Netw. 72 (2015) 140–151.
[31] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2014, pp. 580–587.
[32] R. Girshick, Fast R-CNN, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1440–1448.
[33] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, in: Proceedings of the Advances in Neural Information Processing Systems, 2015, pp. 91–99.
[34] X.-Y. Chen, X.-Y. Peng, J.-B. Li, Y. Peng, Overview of deep kernel learning based techniques and applications, J. Netw. Intell. 1 (3) (2016) 83–98.
[35] A.G. Wilson, Z. Hu, R. Salakhutdinov, E.P. Xing, Deep kernel learning, in: Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, 2016, pp. 370–378.
[36] P. Sermanet, K. Kavukcuoglu, S. Chintala, Y. LeCun, Pedestrian detection with unsupervised multi-stage feature learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2013, pp. 3626–3633.
[37] W. Ouyang, X. Wang, Joint deep learning for pedestrian detection, in: Proceedings of the IEEE International Conference on Computer Vision, IEEE, 2013, pp. 2056–2063.
[38] P. Luo, Y. Tian, X. Wang, X. Tang, Switchable deep network for pedestrian detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2014, pp. 899–906.
[39] A. Angelova, A. Krizhevsky, V. Vanhoucke, A.S. Ogale, D. Ferguson, Real-time pedestrian detection with deep network cascades, in: BMVC, 2015, pp. 32–1.
[40] J. Hosang, M. Omran, R. Benenson, B. Schiele, Taking a deeper look at pedestrians, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 4073–4082.
[41] D. Tomè, F. Monti, L. Baroffio, L. Bondi, M. Tagliasacchi, S. Tubaro, Deep convolutional neural networks for pedestrian detection, Signal Processing: Image Communication 47 (2016) 482–489.
[42] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the Computer Vision and Pattern Recognition, 2014.
[43] S. Xie, Z. Tu, Holistically-nested edge detection, in: Proceedings of IEEE International Conference on Computer Vision, 2015.
[44] P. Dollar, C. Wojek, B. Schiele, P. Perona, Pedestrian detection: an evaluation of the state of the art, IEEE Trans. Pattern Anal. Mach. Intell. 34 (4) (2012) 743–761.
[45] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Caffe: Convolutional architecture for fast feature embedding, in: Proceedings of the 22nd ACM international conference on Multimedia, ACM, 2014, pp. 675–678.
[46] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.
[47] I. Jolliffe, Principal component analysis, Spring-Verlag, New York, 1986.
[48] Y. Ke, R. Sukthankar, PCA-SIFT: a more distinctive representation for local image descriptors, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol.2, IEEE, 2004, pp. II–506.
[49] P. Dollár, Piotr's Computer Vision Matlab Toolbox (PMT), (http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html).
[50] J.J. Lim, C.L. Zitnick, P. Dollár, Sketch tokens: a learned mid-level representation for contour and object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2013, pp. 3158–3165.

Chao Li received the B.S. degree in Electronics and Information Engineering from Huazhong University of Science and Technology (HUST), Wuhan, China in 2013. He is currently a Ph.D. student in the School of Electronic Information and Communications, HUST. His research areas mainly include object detection and medical image analysis.



Xinggang Wang is an Assistant Professor of School of Electronic Information and Communications of Huazhong University of Science and Technology. He received his Bachelors' degree in communication and information system and Ph.D. degree in Computer Vision both from Huazhong University of Science and Technology. From May 2010 to July 2011, he was with the Department of Computer and Information Science, Temple University, Philadelphia, PA, as a visiting scholar. From February2013 to September 2013, he was with the University of California, Los Angeles, as a Visiting Graduate Researcher. He is a Reviewer of IEEE Transaction on Cybernetics, Pattern Recognition, Computer Vision and Image Understanding, Neurocomputing, CVPR, ICCV, ECCV, etc. His research interests include computer vision and machine learning.

Wenyu Liu is now a professor and associate dean of the School of Electronic Information and Communications, HUST. He received the B.S. degree in Computer Science from Tsinghua University, Beijing, China, in 1986, and the M.S. and Ph.D. degrees, both in Electronics and Information Engineering, from Huazhong University of Science and Technology (HUST), Wuhan, China, in 1991 and 2001, respectively. His current research areas include computer vision, multimedia, and sensor network. He is a senior member of IEEE.