


RESEARCH PAPER

Open Access



Pedestrian detection with motion features via two-stream ConvNets

Ryota Yoshihashi^{1*} , Tu Tuan Trinh¹, Rei Kawakami¹, Shaodi You^{2,3}, Makoto Iida¹ and Takeshi Naemura¹

Abstract

Motion information can be important for detecting objects, but it has been used less for pedestrian detection, particularly with deep-learning-based methods. We propose a method that uses deep motion features as well as deep still-image features, following the success of two-stream convolutional networks, each of which are trained separately for spatial and temporal streams. To extract motion clues for detection differentiated from other background motions, the temporal stream takes as input the difference in frames that are weakly stabilized by optical flow. To make the networks applicable to bounding-box-level detection, the mid-level features are concatenated and combined with a sliding-window detector. We also introduce transfer learning from multiple sources in the two-stream networks, which can transfer still image and motion features from ImageNet and an action recognition dataset respectively, to overcome the insufficiency of training data for convolutional neural networks in pedestrian datasets. We conducted an evaluation on two popular large-scale pedestrian benchmarks, namely the Caltech Pedestrian Detection Benchmark and Daimler Mono Pedestrian Detection Benchmark. We observed 10% improvement compared to the same method but without motion features.

Keywords: Pedestrian detection, Motion, Deep learning, Convolutional neural network

1 Introduction

Pedestrian detection is a long-standing challenge in the image recognition field, and its applications are diverse, e.g., in surveillance, traffic security, automatic driving, robotics, and human-computer interaction. However, even state-of-the-art detectors still miss many pedestrians who appear small, vague, or non-salient because appearance features cannot be helpful to resolve such hard instances. Contrarily, one of the important factors in this detection is to exploit motion information. Human brains process motion in the early stage in the pipeline of the visual cortices to enable humans to notice and react to moving objects quickly [1]. Ambiguous objects in still images that are not recognizable to humans can be sometimes easily distinguished when motion is available. Following this idea, several studies have shown that detection performance can be improved by combining motion features with appearance features [2–7]. However, performance gain by motion features has been slight, and how

to incorporate motion clues into detection to achieve the best performance is still under discussion.

Despite the importance of motion information, they are less exploited with current state-of-the-art pedestrian detection methods. According to the benchmark Caltech Pedestrian [8], incorporated motion features still rely only on hand-crafted features such as histograms of oriented gradients of optical flow (HOF [3, 6]), or temporal differences of weakly stabilized frames (SDt [7]), which is relatively more popular as it ignores non-informative motions. The key insights of these hand-crafted features are that (1) contours of moving objects in fine scale is important for detection, and (2) informative motions have to be extracted by factoring out unnecessary motions (such as camera-centric motions).

Deep motion features, on the other hand, have been actively explored in activity recognition [9–14]. Though deep convolutional neural networks (ConvNets) over spatio-temporal space had not been competitive against hand-crafted features with sophisticated encodings and classifiers, recently proposed two-stream ConvNets [15] have achieved remarkable progress. Two-stream ConvNets are inspired by two (ventral and dorsal)

*Correspondence: yoshi@hc.ic.i.u-tokyo.ac.jp

¹The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan
Full list of author information is available at the end of the article

streams in the human brain, and capture spatial and optical flow features in separate ConvNets. Spatial features from ConvNets have been extensively used in pedestrian detection, and deep methods [16–18] have produced state-of-the-art results. Thus, a two-stream framework to combine spatial and temporal features should enable significant and natural improvements over these methods.

We present a deep learning method for pedestrian detection that can exploit both spatial and temporal information with two-stream ConvNets. Based on the findings in hand-crafted motion features, we demonstrate that deep learning over SDt [7] efficiently models the contours of moving objects in fine scale without unwanted motion edges. Our deep motion features are more discriminative than hand-crafted or raw features, as it has been the case for still-image features. The presented ConvNets are novel in the following two aspects. First, SDt, an effective motion feature for pedestrian detection, is used as inputs for temporal ConvNets instead of raw optical flow, as it factors out camera- and object-centric motions that are prominent in videos from car-mounted cameras. Second, we adopt concatenation of mid-level feature maps instead of late fusion of class scores [15], which is similarly done in channel-feature-based methods [16, 19, 20]; thus, our networks can be applied to arbitrary input image sizes and enable sliding-window detection rather than video-level classification. Our detector is implemented on the basis of the convolutional channel feature detector [16], which provides a simple yet powerful baseline for deep pedestrian detection methods. The resulting architecture can effectively learn features from multiple datasets.

Our contributions are four-fold. First, we propose the first method that utilizes deep motion feature for pedestrian detection. Second, we design a two-stream network structure that enables sliding-window detection. Third, we show that an activity recognition dataset [21] can improve pedestrian detection through transfer learning. Finally, our experimental results reveal how deep motion features can be effective by benchmarking in open datasets and via intensive analysis. The proposed method with deep motion features achieved 10% reduction of miss rate from a deep detector only with still-image feature [16] in popular *Caltech Pedestrian* and *DaimlerMono*, and this is a larger improvement than directly incorporating SDt [7].

2 Related work

Pedestrian detection has been one of the hottest topics in computer vision over the last decade, and over 1000 papers have been published on this topic [20]. Pedestrian detection has been mainly addressed with the so-called sliding-window detectors, which classify each window into pedestrian or others. The progress in these methods can be roughly categorized into improvements of

features and classifiers. Features have been improved both by hand-crafted design and feature learning.

Hand-crafted features Exploring more effective features has been one of the central interests in object recognition. The early successful features are Haar-like [22] and Histograms of Oriented Gradients (HOG) [23] and followed by their many variants. The ideas of feature design are the use of gradients [23, 24], edges [25], robust color descriptors [6], multi-resolution [19], covariance and co-occurrence features [26, 27], and speed increase with binary features [28]. Modern detection methods often use multiple features in combination. Some papers found effective sets of features that work better together [29], and others focused on the methods for aggregating different types of features and channels. These excellent hand-crafted features maintain the competitive performance in pedestrian benchmarks [30] even after the appearance of deep learning [20].

Deep learning Deep learnings are data-driven methods for learning feature hierarchies. They are now important in pedestrian detection as well as generic object classification [31–33], detection [34], and semantic segmentation [35]. Historically, researchers have applied simple ConvNets [36], unsupervised ConvNets [37], or specially designed networks [38–41] to pedestrian detection. However, they did not significantly outperform the state-of-the-art detectors with hand-crafted features. Hosang et al. [42] enlightened the strength of ConvNets by introducing transfer learning; namely, the networks become more powerful when they are pre-trained on the large ImageNet rather than trained only on pedestrian datasets. Several recent studies made remarkable progress by using ConvNets in combination with decision forests [16], transfer and multi-task learning in other tasks [17], and part mining [18].

Classifiers Classifiers are the other main component of pedestrian detectors. Beyond basic linear support vector machines (linear SVMs) [43] and AdaBoost [44], many types of classifiers have been developed and introduced into detectors. Boosted forests, which can be interpreted as both a forest version of AdaBoost decision stumps and a boosting version of random forests [45], have recently been preferred in pedestrian detection [16, 19]. Structured classifiers such as partial area-under-curve optimization [46–48] and deformable part models (DPMs) with latent SVMs [49, 50] are also useful. Since DPMs can be interpreted as a variant of ConvNets [51], their advantage is inherent in ConvNets. Faster computation by cascading [52, 53] is also possible over boosted forest classifiers.

Motion features Use of motion features is another important aspect in pedestrian detection, and should be able to boost the performance of detectors. Considering applications to video-based tasks such as surveillance or automatic vehicles, use of motion is natural for pedestrian detection. Several studies involved motion in detection with optical flow [3, 6, 54], multi-frame features [5], temporal differencing [2, 55], and detection by tracking [4]. The most popular among the scoreboard leaders in the benchmark [8] is SDt [7], which can remove camera-centric and object-centric motions by warping the frames with coarse optical flow and detect informative deformation of objects in fine scale by time differencing.

There are more hand-crafted and deep motion features in other video-based tasks, such as video segmentation, activity detection, motion analysis, and event detection. Recent perspectives of the evolution of deep motion features with respect to hand-crafted features can be found, e.g., [56, 57]. In short, rather than using ConvNets over spatio-temporal space, as in multi-frame ConvNets [12, 14], it is more successful to use separate ConvNets for spatial and temporal (optical flow) features [15]. Similar two-stream architectures have been introduced in more recent studies [56–59].

Generic object detection in videos also draws attention recently, as the largest competition in image recognition (ILSVRC2015 [60]) has started a competition of this task. Currently, most methods in the competition adopt single-frame detection and tracking of detected objects, rather than utilizing motion features. Nevertheless, novel techniques are introduced in this dataset, including flow-based feature propagation [61, 62] and joint tracking [63]; however, they are for minimizing motion to maintain temporal consistency of object classes, but not for exploiting motion as a clue by itself. In other challenging domains such as bird surveillance, recurrent-net-based detectors have been examined [64, 65], but recurrent nets can be difficult to train [66] especially in complex environments such as road scenes. Overall, these methods are not clearly effective in pedestrian detection.

Considering these discussion on pedestrian detection and deep motion features, there is room to improve pedestrian detection by using deeply learned motion feature. Rather than inputting optical flow stacks into the temporal stream, we use SDt [7], as it can factor out non-informative motion. This is crucial, as many videos captured using car-mounted cameras contain ego-motions from the car, and removing such motions to extract important flow is necessary. In this paper, we discuss the strength of detection by fusing spatial channel features and stabilized motion features both learned by deep neural nets. To the best of our knowledge, this paper proposes the first detection method that uses motion with ConvNets.

3 Motivation from biology

Our motivation of exploiting motion features in pedestrian detection and designing two-stream network architecture is from insights from neuroscience and visual psychology on how human vision processes motion information. From the structural viewpoint, visual cortices have two main streams for recognition and motion, namely ventral stream and dorsal stream [67]. This inspired two-stream structure in artificial neural networks for video tasks [15, 68], including ours. While the processing pipeline of motion is hierarchical, it is worth noting emphasis of temporal changes are performed at early stages by cells sensitive to temporal changes [1].

From the functional viewpoint, perception of motion is related to recognition of objects even when appearance features of objects are unavailable. An example is biological motion [69], which is a phenomenon that motion patterns showed via individually meaningless signals (i.g., point light) can be perceived as human locomotion or gestures. Based on these insights, we design two-stream pedestrian detection method aided by motion information.

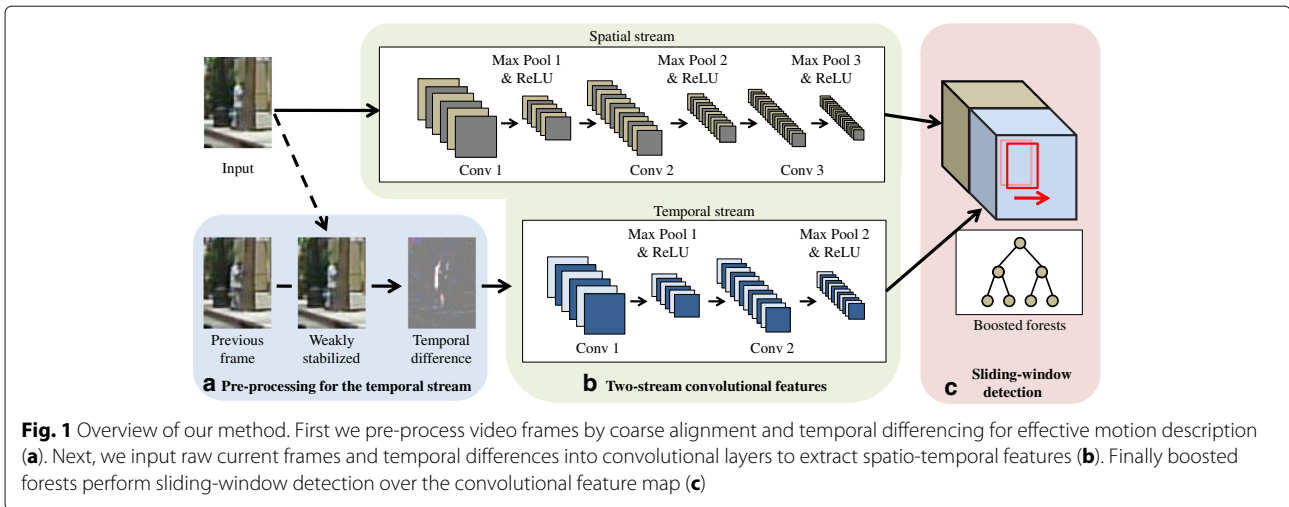
4 Two-stream ConvNets for pedestrian detection

The overview of our method is shown in Fig. 1. It consists of a two-stream ConvNet that takes the current frame for spatial ConvNets, and the temporal difference of weakly stabilized frames via coarse optical flow for temporal ConvNets. The outputs from the two streams are concatenated and compose feature maps, and the boosted forests perform sliding-window detection over them.

4.1 Two-stream convolutional features

Our pipeline incorporates motion information via two-stream convolutional features (Fig. 1b, which is a two-stream network architecture [15] adopted in sliding-window detection. The first stream for spatial features follows the framework of convolutional channel features (CCF) [16] as a baseline. It consists of the lower part of VGGNet-16 [32], which includes three convolutional layers and three max-pooling layers, and is pre-trained on the ILSVRC2012 dataset [60]. According to Yang et al. [16], VGGNet-16 produces better features for pedestrian detection than AlexNet [31] or GoogLeNet [33], supposedly because deeper networks are difficult to transfer to other tasks.

For the second stream for temporal features, we build convolutional features specialized in motion feature extraction. While ImageNet-pretrained features generalized well to detection tasks [16, 34], it does not work for motion handling because they are derived from the still-image classification task. Therefore, we train the ConvNets over difference images collected from video



datasets. The training is necessary, if we want to work on temporal differences, because statistical distribution of difference images has different nature from that of still images. We collect difference images of UCF-101 [21], a large activity recognition dataset with 13,320 video clips categorized into 101 classes, and we train AlexNet on them. AlexNet is chosen because of its trainability. It has less layers than VGGNet-16 or GoogleNet and therefore is easy to train on datasets smaller than ImageNet such as UCF-101. For feature extraction, we used the second convolutional layer in AlexNet, which makes 256-dimensional feature maps, the same dimension as that of the spatial stream. We refer to this trained AlexNet as UCFAlex.

There are several options for the second ConvNets. For instance, one may think that we can reuse VGGNet-16 of the first stream, if we want to input the previous frame or weakly aligned previous frame to the temporal stream. However, such an approach can be weak because statistical distribution of difference images has different nature from that of still images. We see the performance gain by our motion ConvNet and ConvNets in the Section 5.

For training of the temporal stream, we use split01 of UCF-101. Specifically, we use the first group (divided according to the source videos of each clip) for validation and the remaining 24 groups for training. We choose stochastic gradient descent [70], the most common ConvNet solver for training in the same setting as those by Simonyan and Zisserman [15], except that we initialize network parameters with those of AlexNet trained on ILSVRC2012 for smooth training. The accuracy on the validation set is 56.5% after training, which outperforms AlexNet on the raw RGB values of each frame (43.3%) and underperforms the temporal stream on the optical flow reported by Simonyan and Zisserman [15]. However, this is not a problem because our purpose was not to achieve the best performance on the activity dataset

but to acquire effective features for temporal difference images. In training, we used difference images of four-frame intervals. We do not apply weak stabilization to UCF-101 videos because camera-centric motion is much less in this dataset.

4.2 Sliding-window detection

The final output of our network is determined using boosted forests. Our detector is a sliding-window detector that classifies densely sampled windows over the feature maps at test time (Fig. 1c).

A main challenge in training the two-stream detector is the larger dimensionality of the feature maps. The output from the two streams has 512 channels after concatenation, and this becomes twice as large as the single spatial stream which makes forests likely to cause overfitting. Thus, we apply data augmentation in the training of the forests. We include both images with and without motion features in the training data. Let us denote the pair of the inputs to our two-stream ConvNets as $(I, D(I))$, where I is the input image and $D(I)$ is the temporal difference of I . We prepare both $(I, D(I))$ and $(I, 0)$ as training data for each labeled bounding box. The term $D(I) = 0$ means that the object in I is stationary. This data augmentation mitigates overfitting and prevents detectors from missing stationary people in the scenes.

We set the hyper parameters of boosted forests to the same as those discussed by Yang et al. [16]. Namely, the number of trees was 4096 and maximum depth of each tree was 5. The ensemble was constructed with the Real AdaBoost algorithm, and the trees were trained with brute-force-like search and pruning [71]. ConvNet features are known to be powerful enough even without fine-tuning on the target tasks [72]. Thus, we did not fine-tune either of the two streams and only the forests were trained on the Caltech Pedestrian Detection Benchmark.

4.3 Pre-processing for the temporal stream

We use temporal differences of weakly stabilized frames [7] as motion descriptors to input to the temporal stream (Fig. 1a). We describe it here for completeness and convenience in notation. First, we calculate the optical flow [73] between the current frame that contains the target and its previous frames. Next, we warp the previous frames using the optical flow, as visualized in Fig. 2. We denote these frames as $I_p(t_0, t_{\text{prev}})$ where t_0 is the current time, t_{prev} is the previous time, and p is the smallest size of the patches used in calculating the optical flow. The optical flow is calculated in a coarse-to-fine manner, and p controls the fineness of the flow. We can use several previous frames and obtain longer motion features as follows. Weakly stabilized frames are

$$S(t_0) = \begin{pmatrix} I_p(t_0, t_1) \\ I_p(t_0, t_2) \\ \dots \\ I_p(t_0, t_n) \end{pmatrix} \quad (1)$$

and temporal differences of these frames are

$$\begin{aligned} D(t_0) &= \begin{pmatrix} I(t_0) \\ I(t_0) \\ \dots \\ I(t_0) \end{pmatrix} - S(t_0) \\ &= \begin{pmatrix} I(t_0) - I_p(t_0, t_1) \\ I(t_0) - I_p(t_0, t_2) \\ \dots \\ I(t_0) - I_p(t_0, t_n) \end{pmatrix} \end{aligned} \quad (2)$$

where $I(t_0)$ is the current frame and n is the number of frames, which gives the time range of motion features.

The choice of p is important for removing relatively uninformative camera-centric or object-centric motions while preserving object deformation and pose change. The coarse flow is useful for this purpose, as previously discussed [7]. We set p to 32 pixels square. For calculating motion, we used the second and fourth previous frames, following [7] for Caltech Pedestrian. There are a variety

of optical flow methods for stabilization. Our choice is Lucas-Kanade flow [73] because it is simple and robust enough to work on Caltech Pedestrian images (30 fps, 640×480 pixels). Other options [74, 75] are also possible and may perform better on more complex scenes or low frame-rate videos.

4.4 Implementation details

Feature map generation We use convolutional layers as feature transformation per frame to enable sliding-window detection over the feature maps, differently from per-window classification approaches [18, 34]. In classical neural networks, the input size (of the image) has to be fixed; however, ConvNets have been recently used as flexible filters, which are applicable to images of arbitrary size (over the size of the convolutional kernels) and fixed channel numbers [16, 76, 77]. This becomes possible when ConvNets are without fully connected layers. We use ConvNets in this manner; namely, they are applied to clipped windows in training but to full-size input images during the test. During the test, the forest classifier looks up pixels in the feature maps, the pre-computed convolutional features of the whole image, to execute sliding-window detection. We also adopt pyramid patchworking [78] during the test. That is, we stitch the spatial pyramid of an input image into a larger single image, which shortens test runtime. This makes the input image size 932×932 pixels while the original size of frames is 640×480 pixels.

Computation Our implementation is based on the CCF [16] implementation, which is based on Piotr's MATLAB Toolbox including the aggregate channel feature (ACF) [19] detector. It also uses Caffe [79] for convolutional feature extraction on GPUs. Weak motion stabilization is also included in Piotr's Toolbox. Our test and training environment was a workstation with two Intel Core i7-5930K CPUs (3.50 GHz, 12 cores), 512 GB memory, and two NVIDIA GeForce GTX TITAN X video cards. Note that 512 GB memory is necessary for training our boosted forests on memory with data augmentation.

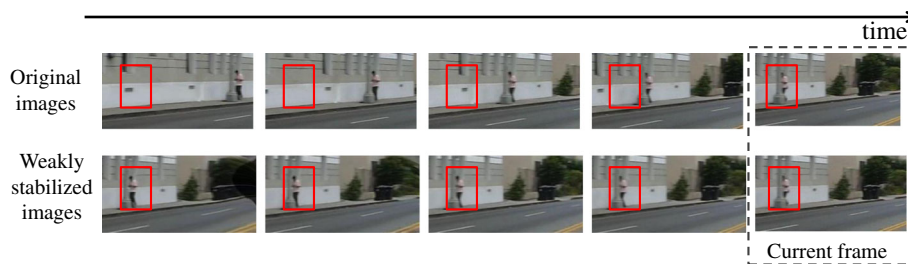


Fig. 2 Effects of weak motion stabilization [7]. Upper images are original input images and lower images are stabilized ones. Background motion is stabilized while deformation of the person remains

5 Experiments

To provide comprehensive understanding of the deep motion feature of our proposed method, we present experiments and analyses consisting of the following three parts: validation, evaluation, and visualization. First, we validated our configuration of networks by comparing it with baselines such as hand-crafted motion features and ImageNet-pretrained ConvNets used as the temporal stream (*validation*). Next, we evaluated our detector with the motion features in the test sets of Caltech Pedestrian Detection Benchmark [80] and Daimler Mono Pedestrian Detection Benchmark Dataset [81] and compared our detector with the state-of-the-art methods from the leader boards (*evaluation*). Finally, we visualize the detection results and learned models to understand how and when the motion features improve pedestrian detection (*visualization*).

5.1 Network selection and validation

First, we compared three candidate combinations of the network architecture, pre-training data, and inputs for the temporal stream of our network: (1) VGGNet-16 pre-trained on ILSVRC2012 by inputting previous weakly stabilized frames (**VGG-SF**), (2) VGGNet-16 pre-trained on ILSVRC2012 by inputting the temporal differences of stabilized frames (**VGG-SDt**), and (3) AlexNet fully trained on UCF-101 by inputting the temporal differences of stabilized frames (**UCFAlex-SDt**). We also validated *ACF*, *CCF*, and our new implementation of *CCF+SDt* for comparison. We used previous second and fourth frames in *CCF+SDt*. Only the previous fourth frames is used in temporal ConvNets for simpler implementation. The compared combinations are listed in Table 1.

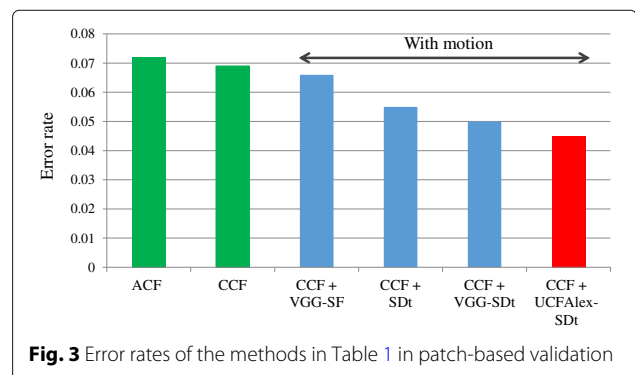
The performances of the candidates were evaluated with a bounding-box-level classification task using the training sets. First, we collected clipped bounding boxes from the training sets of the benchmark (set00–set05). We used every fourth frame from the training set to acquire

more training boxes. This setting is often referred to as *Caltech10x*. We used the ground truth annotations. For negative windows, we used those generated in the hard negative mining process for training of a baseline algorithm, i.e., ACF detector [19]. We conducted 3-stage hard negative mining with ACF and refer to those collected negatives as *NegStage1–3*. Next, we split each group of bounding boxes into two parts, the first half for training and the latter half for validation. We did not apply data augmentation in this validation for comparison in an equal amount of training data. For hyperparameters of boosted forests, the maximum number of weak classifiers was fixed to 4096, and the maximum depth of trees to 5. Note that only the forest classifier was trained on Caltech Pedestrian and the networks were not fine-tuned after pre-training separately on Caltech Pedestrian or other datasets.

Figure 3 shows the validation results. All methods with multi-frame motion information improved in accuracy compared to single CCF. Motion information from stabilized differences had more positive effects on classification than that from stabilized frames without time differencing, even VGGNet-16 trained on still images was used (*CCF+VGG-SF* vs. *CCF+VGG-SDt*). This should be because of the correlation between frames, which is known to be harmful for decision trees [82]. Time differencing can mitigate this correlation and improve training of decision trees. ConvNets over SDt worked better than simple SDt (*CCF+SDt* vs. *CCF+VGG-SDt*), and our motion-specialized AlexNet was more suited to process SDt (*CCF+VGG-SDt* vs. *CCF+UCFAlex-SDt*). It would be surprising that transferred features from the far task of activity recognition work in pedestrian detection. This suggests that the distance of input domain, i.e., still images vs. difference images, mattered more than the distance of the tasks in reusability of lower-level convolutional features. Several examples for difference images for pre-training are shown in Fig. 4. Overall, the combination of AlexNet re-trained on activity recognition and stabilized differences (*CCF+UCFAlex-SDt*) performed the best

Table 1 Methods we validated in the splits of training data in Caltech Pedestrian Detection Benchmark [8], sorted by validation error rates in descending order (corresponding to Fig. 3)

| Method | Spatial feature | Temporal feature | Temporal pre-training | Temporal input |
|-------------------|-----------------|------------------|-----------------------|----------------|
| ACF [19] | Hand-crafted | - | - | - |
| CCF [16] | VGGNet-16 | - | - | - |
| CCF + VGG-SF | VGGNet-16 | VGGNet-16 | ILSVRC2012 | SF |
| CCF + SDt | VGGNet-16 | SDt itself | - | SDt |
| CCF + VGG-SDt | VGGNet-16 | VGGNet-16 | ILSVRC2012 | SDt |
| CCF + UCFAlex-SDt | VGGNet-16 | UCFAlex | UCF-101 | SDt |



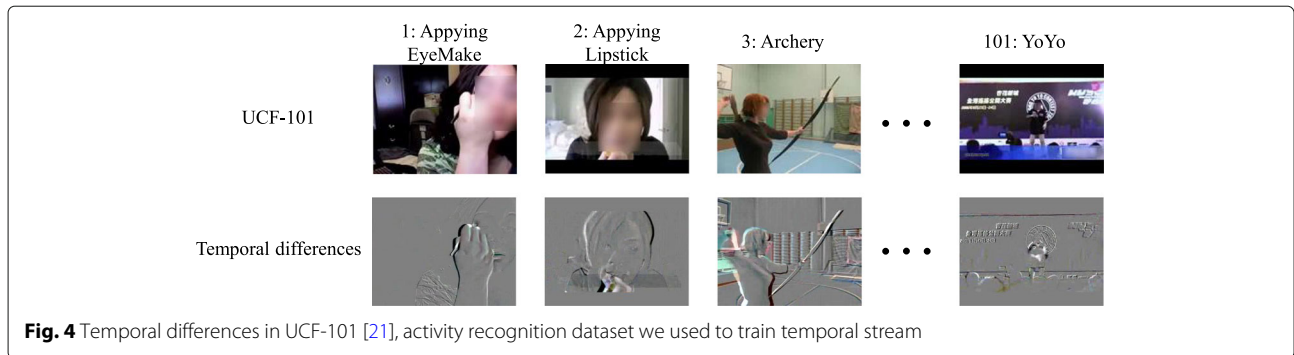


Fig. 4 Temporal differences in UCF-101 [21], activity recognition dataset we used to train temporal stream

among the candidates. Below, we refer to CCF+UCFAlexSDt as *TwoStream*.

5.2 Detection evaluation

Next, we conducted detection experiments on the test sets of the Caltech Pedestrian and DaimlerMono to evaluate the performance of two streams. We also compared the results of two streams to current pedestrian detection methods including ones using only single frames and using motion information. We also evaluated our implementation of CCF+SDt to compare to our temporal ConvNets.

In the evaluation, we strictly followed the evaluation protocol of the benchmark. For the training/test split, we used the suggested split on the benchmark, namely, set00–05 for training and set06–10 for test. Detection was run on every thirtieth frame of each test video. The previous frames of the test frames were also used in the test because two streams requires them. We used second and fourth previous frames in CCF+SDt and fourth frames in two streams. We conducted all the evaluation process using official APIs for the dataset. We also applied our detector to the test set of DaimlerMono in the same manner to see whether our detector generalizes to another dataset. Because DaimlerMono only provide gray-scale frames, we convert them to RGB by simply copying the channels. As we aim to evaluate transferability of our Caltech-trained detector, note that the training set of DaimlerMono is not used.

To train our detector for evaluation, we sampled bounding boxes from every fourth frames from the training sets (set00–05), using the baseline detector, i.e., ACF [19]. For better performance, we used flipped windows of positive samples in addition to the original training data. We also applied data augmentation to the positives and NegStage1 and 3, but not to NegStage2 due to memory reasons. The numbers of positive and negative training windows were 50,000 and 100,000, respectively, after data augmentation.

The results were evaluated using receiver operation characteristic (ROC) curves, which plot detection miss rates to the numbers of false positives per images. Figure 5a shows the ROC curves of the detection results

in Caltech Pedestrian. It also shows published results in the evaluation paper and scoreboard of the benchmark for comparison. We included the results of boosted-forest-based methods, namely, ACF and ACF-caltech+ [19], LDCF [82], Checkerboards+ [20], TA-CNN [17], CCF [16], MultiFtr+Motion [6], ACF+SDt [7], and our baseline implementation for a deep method with motion

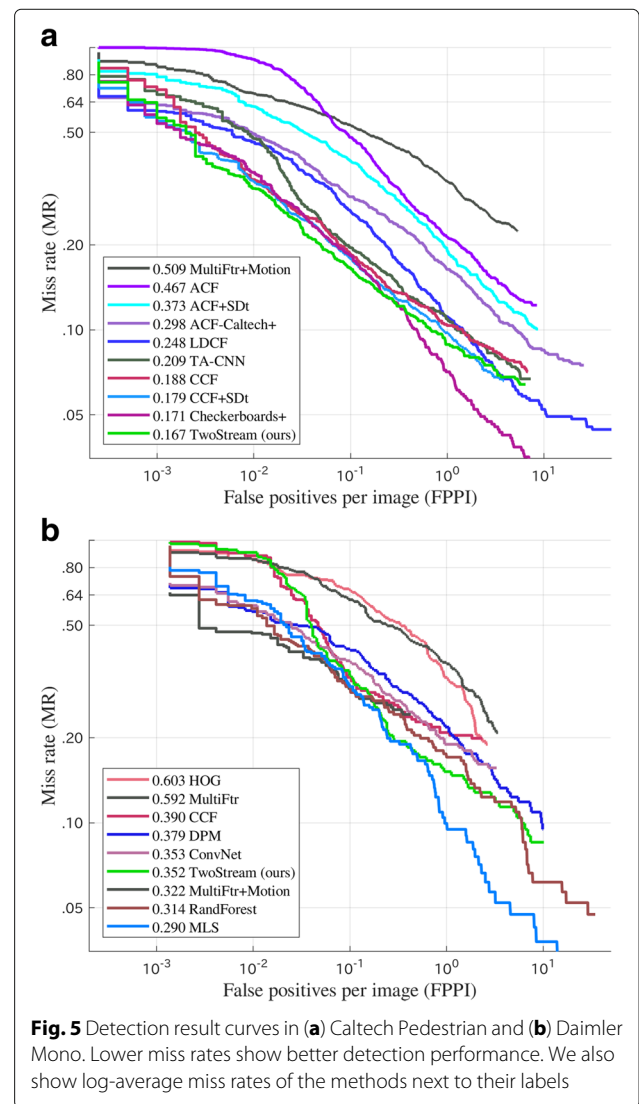


Fig. 5 Detection result curves in (a) Caltech Pedestrian and (b) Daimler Mono. Lower miss rates show better detection performance. We also show log-average miss rates of the methods next to their labels

(CCF+SDt). The proposed method (two streams, log-average miss rate 0.167) performed 10% better than motion-free baseline (CCF, 0.188), and this is a larger improvement than by simply adding SDt to CCF. Two streams performed better than all the hand-crafted-feature-based detectors with or without motion and two deep detectors CCF and TA-CNN.

Figure 6 shows several examples of detection in Caltech Pedestrian. Two streams detected more pedestrians than single-frame-based CCF due to temporal information, as shown in Fig. 6a. Low-resolution pedestrians were particularly difficult even for humans to detect in single frames; however, Two stream succeeded in detecting them from motion. In contrast, Fig. 6b shows suppressed false detections by TwoStream, which were misdetected by only single-frame information. This shows motion is also

useful to reject non-pedestrian region more easily than only by appearance.

Figure 5b shows the ROC curves of the detection results in DaimlerMono. In this transfer setting from Caltech to DaimlerMono, the proposed TwoStream performed better than the baseline CCF by 3.8 percentage points or 10.7% in relative. This means TwoStream is transferable to another dataset from Caltech Pedestrian. TwoStream also outperformed existing *ConvNet* [37], while the hand-crafted-feature-based methods (MultiFtr+Motion [6], RandomForest [83], and MLS [84]), which are considered to be more robust in gray-scale images, marked better scores. We also note that these three detectors are trained in INRIA dataset, that seems to be closer to DaimlerMono, while we used Caltech Pedestrian.

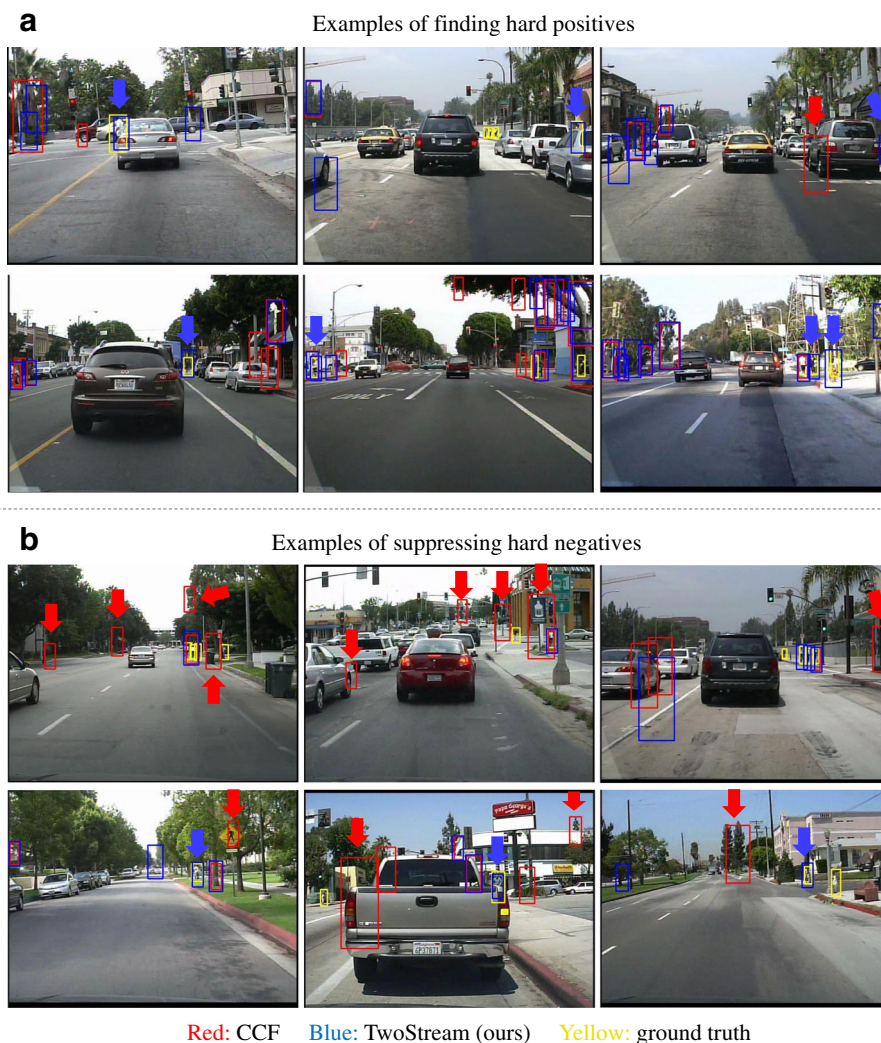


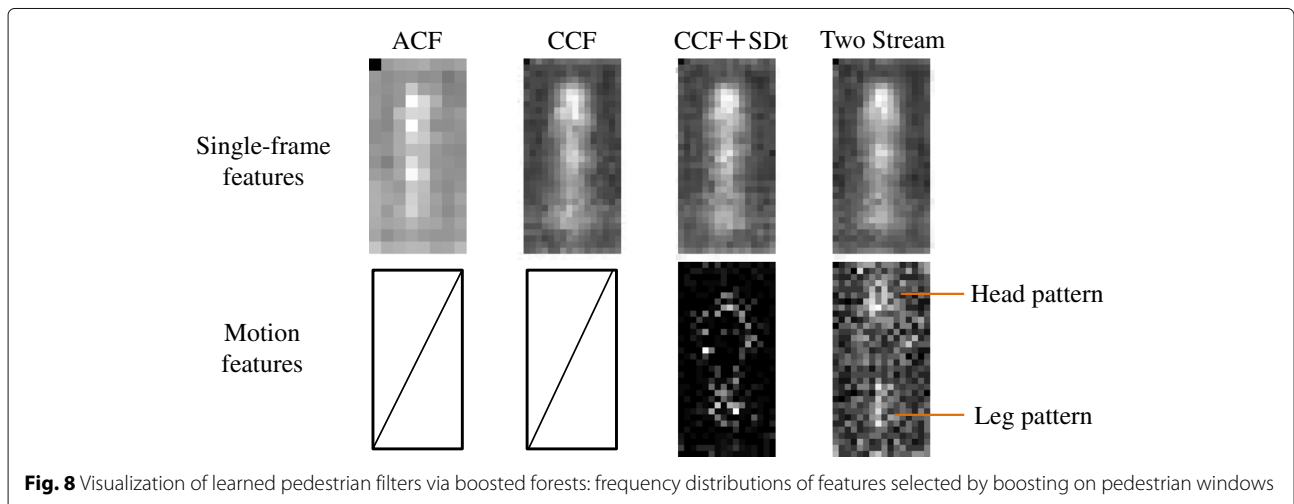
Fig. 6 Detection examples and comparison with CCF (baseline) and TwoStream (ours). TwoStream detects more pedestrians, including hard-to-detect ones, with help of temporal information (a). TwoStream is also more robust against hard negative samples than CCF (b)



5.3 Analyses and visualization

How did detection confidences change? To understand how the motion stream helps recognition, we present examples of patches differently scored with TwoStream and single-frame baseline CCF in Fig. 7. We also show each patch’s temporal differences with weak stabilization, which were the input for the motion stream in TwoStream. Figure 7a shows negative samples scored highly with CCF but were mitigated with TwoStream. Non-pedestrian objects such as poles (a and b) or parts of vehicles (c and d) also scored highly. They have vertical edges and may be misclassified as pedestrians only

by appearance, but they can be correctly rejected by motion information because they are rigid and their temporal differences are negligible after stabilization. However, non-pedestrian samples may produce large temporal differences by fast motion information such as crossing vehicles (e). In this case, the motion stream CNN seems to discriminate difference patterns and correctly ignore non-pedestrian motions. Panel d is a hard negative of a silhouette of a person in a traffic sign, that was successfully rejected by TwoStream. Figure 7b includes pedestrians that scored higher with TwoStream. TwoStream scored highly for pedestrians with typical and salient motion



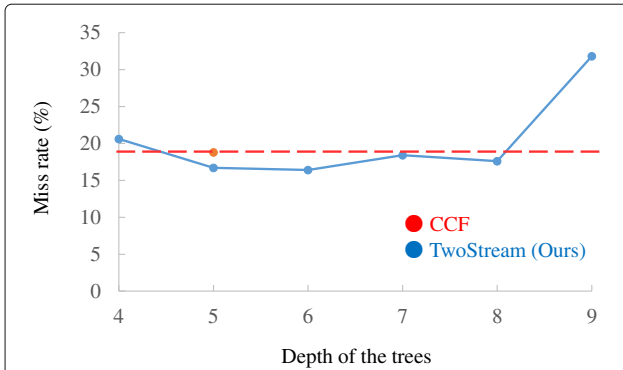


Fig. 9 Relationship between tree depth in boosted forests and log-average miss rate in Caltech *Reasonable* subset. TwoStream consistently outperformed the baseline with depth of 5 to 8, thus was robust against this parameter, while overly shallow (less than 4) or deep (over 9) trees degraded detection performance

Visualization We visualized trained forests on different features, as shown in Fig. 8. The figure shows frequency distributions of where in feature maps was seen by decision nodes in the forests. Comparing ACF and CCF, a clearer pedestrian shape on the deep feature maps via CCF was observed, while ACF used backgrounds as well. Comparing the temporal features of CCF+SDt and TwoStream, SDt grasps pedestrian contours while TwoStream used more motion patterns around the legs and heads.

Runtime The runtime of TwoStream on 640×480 videos were 5.0 s per frame on average. The runtime of CCF was 3.7 s per frame in [16] and 2.9 s per frame in our environment. Both were implemented on MATLAB and Caffe on GPUs. The overheads with our temporal stream and coarse optical flow were less dominant than other factors including video decoding and communication with GPUs in our implementation.

patterns (g, h, and j) by large margins, which supports that it utilizes pedestrian motions for recognition. In addition, blurred or non-salient pedestrians (i, k, and l) tended to be scored lower on the basis of only appearance features, but many parts of such pedestrians were detected, thanks to motion feature information. There were several contrasted cases, i.e., non-pedestrians scored higher and pedestrians scored lower with TwoStream, as shown in Fig. 7c, d, respectively. Negative patches with complex shapes such as trees or parts of buildings were sometimes scored the same or higher with TwoStream (m and n), and a vehicle’s motion that was not correctly stabilized (o) also caused misdetection. Pedestrians were scored lower when they were static (p), occluded by moving vehicles (r), or made rare motions such as getting into cars (q), whose motion patterns the detector may fail to learn from the training set.

Tree depth Tree depth is an important parameter to fully exploit strong features in boosted-forest-based detectors. We further investigated the impact of tree depth in the forest in Fig. 9. We found that the miss rate did not largely differs with depths of around 6. It consistently outperformed the baseline CCF with depth of 5 to 8; and thus, it is robust against setting of this hyperparameter. Here, we put best-performing CCF with tree depth of 5 as the baseline. Nevertheless, overly shallow (less than 4) or deep (over 9) trees degraded detection performance.

Failure cases Failure cases where TwoStream did not improve detection compared to the original CCF are shown in Fig. 10. A major cause of failures is inaccurate optical flow around occlusion (the top sample in Fig. 10)

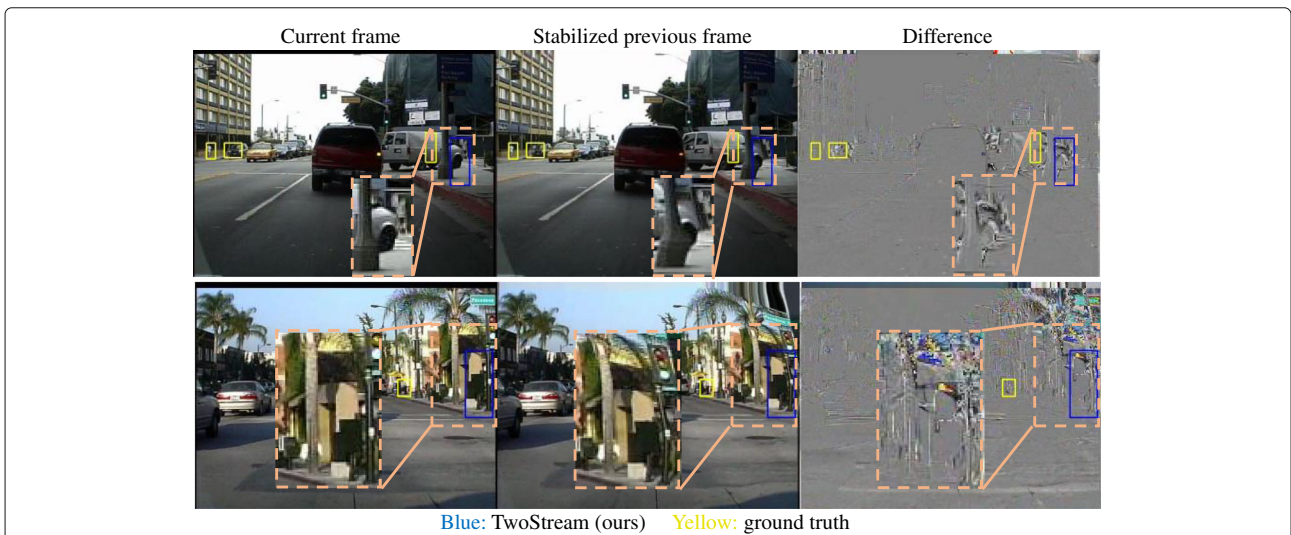


Fig. 10 Failure cases. In these frames, TwoStream caused new misdetections that CCF did not misdetected, due to improperly stabilized motions by inaccurate optical flow

Table 2 Reduction of MR by combining our deep motion feature with various appearance features

| Base method | ACF-ours | LDCF-ours | CCF | Two-stage CNN |
|------------------------------|-------------|-------------|-------------|---------------|
| Appearance feature only | 36.7 | 33.4 | 18.8 | 13.6 |
| With our deep motion feature | 31.4 | 30.6 | 16.4 | 12.4 |

Lower is better

or image boundaries (the lower samples), which fails to remove background motions. However, we also noted that TwoStream improved the total detection accuracy despite such optical flow errors, thanks to complementary usage of appearance and motion features.

5.4 Combination with other methods

To show the effectiveness of our deep motion feature, we combined them with other types of methods and evaluated relative performances. First, we adopted popular hand-crafted channel features, ACF [19] and LDCF [82], in addition to CCF. In implementation, we replaced CCF in TwoStream by the other appearance features without modification. Training details are the same as those for our implementation of CCF. The results are shown in Table 2. Our deep motion features decreased MR by 5.3% with ACF and by 2.8% with LDCF. The results confirmed that our deep motion features consistently improve detection performance of various appearance-based features.

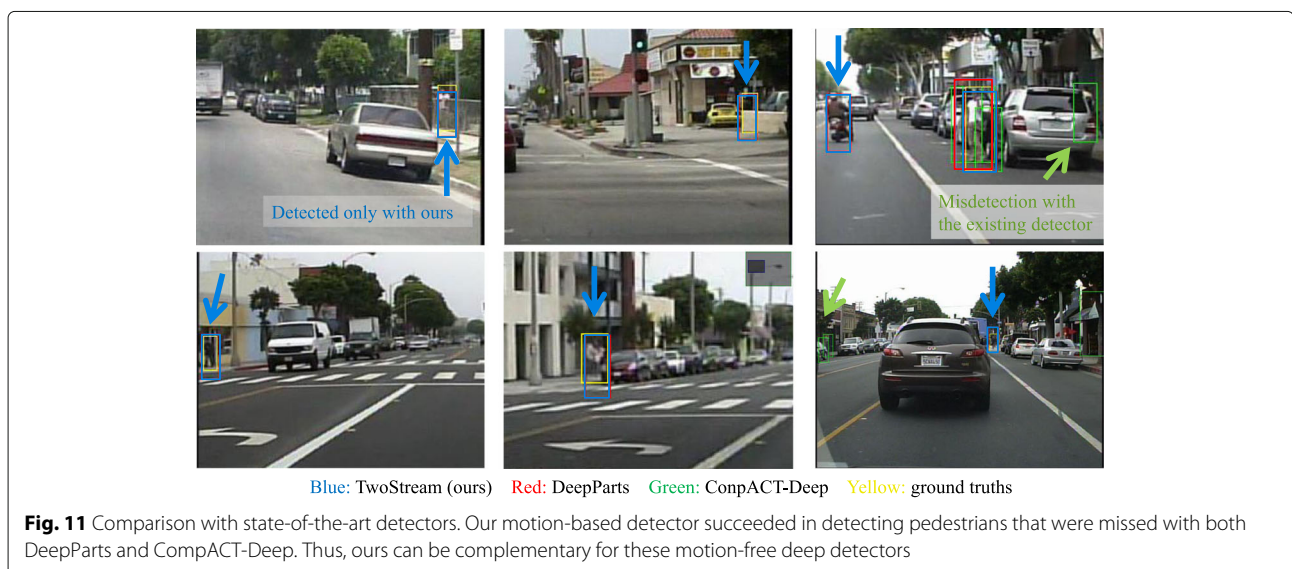
Second, we examined the combination with second-stage CNN (two-stage CNN) that gives two-stage pipelines similar to state-of-the-art systems [18, 53, 85]. In this setting, we rescore the detected regions by CCF or TwoStream by a vanilla VGGNet-16 [32] fine-tuned on Caltech Pedestrian by ourselves, instead of the CNNs re-engineered for pedestrians [18, 53, 85]. TwoStream still provided significant improvement of MR by 1.2%, and this

suggests that our deep motion features can improve the state-of-the-art detectors with two-stage pipelines.

Some very recent deep-learning-based methods outperformed ours in Caltech Pedestrian by using more complex networks. Three deep methods without motion, namely DeepParts (11.9% MR) [18], CompACT-Deep (11.7% MR) [53], and SA-FastRCNN (9.7% MR) [85] outperformed TwoStream because of the differences in the frameworks and the learning methodology. Specifically, one main difference is that these methods adopted two-stage architecture, i.e., first generate candidate bounding boxes by other detectors and then re-categorize them by second-stage ConvNets.

They also introduced techniques to improve second-stage ConvNets, such as part mining [18], cascading with shallow features [53], or scale adaptation [85]. The other difference is that the ConvNets in them are pre-trained using ILSVRC2014-DET and then fine-tuned by Caltech Pedestrian. TwoStream realizes simpler system, as it does not require other detectors for object proposals; however, it may improve by updating the ImageNet dataset for pre-training and introducing fine-tuning over pedestrian datasets. In addition, TwoStream can alternate the sliding-window detectors in those deep methods, and this would further improve overall performance since TwoStream can detect some pedestrians that the above methods miss due to visual obscurity or hard blur, as shown in Fig. 11. Specifically, [18, 83] used LDCF [82], and [53] used ACF [19] + LDCF [82] + CheckerBoard [20] as their first-stage detectors, and ours outperformed all of them.

End-to-end deep-learning-based approaches [86, 87] also work well in Caltech, but they require full-frame-annotated training data; thus, are not applicable to datasets that only provide pre-cropped training windows such as DaimlerMono.



6 Conclusions

We demonstrated a method for exploiting motion information in pedestrian detection with deep learning. With our method, we fused the ideas from a two-stream network architecture for activity recognition and convolutional channel features for pedestrian detection. In the experiments on the Caltech Pedestrian Detection Benchmark and Daimler Mono Pedestrian Detection Benchmark, we achieved a reasonable decrease of detection miss rate compared to existing convolutional network pedestrian detectors, and the analyses revealed that the motion feature improved detection in recognizing hard examples, which even state-of-the-art detectors fail to discriminate.

We also believe that our framework is helpful in considering other video-based localization tasks, such as generic object detection in video and video segmentation. In future, we explore such applications of deep motion features.

Acknowledgements

The authors would like to thank Prof. Toshihiko Yamasaki of the University of Tokyo for his helpful advice on the manuscript.

Funding

This work is in part entrusted by the Ministry of the Environment, Japan (MOEJ), the project of which is to examine effective measures for preventing birds, especially sea eagles, from colliding with wind turbines. This work is also supported by JSPS KAKENHI grant number JP16K16083 and Grant-in-Aid for JSPS Fellows JP16J04552.

Availability of data and materials

The authors will release the code to reproduced results in this manuscript.

Authors' contributions

RY designed and executed the experiments and wrote the manuscript. TT contributed to the concept and helped with the experiments. RK advised RY on the concept and experiments and wrote the manuscript. SY advised RY on the concept edited the manuscript. MI and TN supervised the work and edited the manuscript. All authors reviewed and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Author details

¹The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan. ²Data61-CSIRO, Canberra, Australia. ³Australian National University, Canberra, Australia.

Received: 12 February 2018 Accepted: 24 August 2018

Published online: 27 September 2018

References

- Krüger N, Janssen P, Kalkan S, Lappe M, Leonardis A, Piater J, Rodríguez-Sánchez AJ, Wiskott L (2013) Deep hierarchies in the primate visual cortex: what can we learn for computer vision? *PAMI* 35(8):1847–1871
- Viola P, Jones MJ, Snow D (2003) Detecting pedestrians using patterns of motion and appearance. In: *CVPR*
- Dalal N, Triggs B, Schmid C (2006) Human detection using oriented histograms of flow and appearance. In: *ECCV*. pp 428–441
- Andriluka M, Roth S, Schiele B (2008) People-tracking-by-detection and people-detection-by-tracking. In: *CVPR*. pp 1–8
- Jones M, Snow D (2008) Pedestrian detection using boosted features over many frames. In: *ICPR*. pp 1–4
- Walk S, Majer N, Schindler K, Schiele B (2010) New features and insights for pedestrian detection. In: *CVPR*. pp 1030–1037
- Park D, Zitnick C, Ramanan D, Dollár P (2013) Exploring weak stabilization for motion feature extraction. In: *CVPR*
- Dollár P, Wojek C, Schiele B, Perona P (2012) Pedestrian detection: an evaluation of the state of the art. *PAMI* 34(4):743–761
- Jhuang H, Serre T, Wolf L, Poggio T (2007) A biologically inspired system for action recognition. In: *ICCV*
- Taylor GW, Fergus R, LeCun Y, Bregler C (2010) Convolutional learning of spatio-temporal features. In: *ECCV*
- Le QV, Zou WY, Yeung SY, Ng AY (2011) Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In: *CVPR*
- Ji S, Xu W, Yang M, Yu K (2013) 3D convolutional neural networks for human action recognition. *PAMI* 35(1):221–231
- Hasan M, Roy-Chowdhury AK (2014) Continuous learning of human activity models using deep nets. In: *ECCV*. pp 705–720
- Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Fei-Fei L (2014) Large-scale video classification with convolutional neural networks. In: *CVPR*
- Simonyan K, Zisserman A (2014) Two-stream convolutional networks for action recognition in videos. In: *NIPS*. pp 568–576
- Yang B, Yan J, Lei Z, Li S (2015) Convolutional channel features for pedestrian, face and edge detection. In: *ICCV*
- Tian Y, Luo P, Wang X, Tang X (2015) Pedestrian detection aided by deep learning semantic tasks. In: *CVPR*
- Tian Y, Luo P, Wang X, Tang X (2015) Deep learning strong parts for pedestrian detection. In: *ICCV*
- Dollár P, Appel R, Belongie S, Perona P (2014) Fast feature pyramids for object detection. *PAMI* 36(8):1532–1545
- Zhang S, Benenson R, Schiele B (2015) Filtered feature channels for pedestrian detection. In: *CVPR*
- Soomro K, Zamir A, Shah M (2012) Ucf101: A dataset of 101 human action classes from videos in the wild. *CRCV-TR-12-01*
- Viola P, Jones M (2001) Rapid object detection using a boosted cascade of simple features. In: *CVPR Vol. 1*. pp 511–518
- Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: *CVPR Vol. 1*. pp 886–893
- Lowe DG (1999) Object recognition from local scale-invariant features. In: *ICCV Vol. 2*. pp 1150–1157
- Wu B, Nevatia R (2005) Detection of multiple, partially occluded humans in a single image by Bayesian combination of edgelet part detectors. In: *ICCV Vol. 1*. pp 90–97
- Tuzel O, Porikli F, Meer P (2008) Pedestrian detection via classification on Riemannian manifolds. *PAMI* 30(10):1713–1727
- Ren H, Li Z-N (15) Object detection using generalization and efficiency balanced co-occurrence features. In: *ICCV*. pp 46–54
- Ojala T, Pietikäinen M, Mäenpää T (2000) Gray scale and rotation invariant texture classification with local binary patterns. In: *ECCV*. pp 404–420
- Wang X, Han T, Yan S (2009) An HOP-LBP human detector with partial occlusion handling. In: *ICCV*. pp 32–39
- Ohn-Bar E, Trivedi MM (2016) To boost or not to boost? On the limits of boosted trees for object detection. In: *ICPR. IEEE*. pp 3350–3355
- Krizhevsky A, Sutskever I, Hinton G (2012) ImageNet classification with deep convolutional neural networks. In: *NIPS*
- Simonyan K, Zisserman A (2015) Very deep convolutional networks for large-scale image recognition. In: *ICLR*
- Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In: *CVPR*
- Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: *CVPR*. pp 580–587
- Long J, Shelhamer E, Darrell T (2015) Fully convolutional networks for semantic segmentation. In: *CVPR*
- Szarvas M, Yoshizawa A, Yamamoto M, Ogata J (2005) Pedestrian detection with convolutional neural networks. In: *Intelligent Vehicles Symposium*. pp 224–229

37. Sermanet P, Kavukcuoglu K, Chintala S, LeCun Y (2013) Pedestrian detection with unsupervised multi-stage feature learning. In: CVPR. pp 3626–3633
38. Ouyang W, Wang X (2013) Joint deep learning for pedestrian detection. In: CVPR. pp 2056–2063
39. Luo P, Tian Y, Wang X, Tang X (2014) Switchable deep network for pedestrian detection. In: CVPR. pp 899–906
40. Fukui H, Yamashita T, Yamauchi Y, Fujiyoshi H, Murase H (2015) Pedestrian detection based on deep convolutional neural network with ensemble inference network. In: IEEE Intelligent Vehicle Symposium
41. Yamashita T, Fukui H, Yamauchi Y, Fujiyoshi H (2016) Pedestrian and part position detection using a regression-based multiple task deep convolutional neural network. In: International Conference on Pattern Recognition
42. Hosang J, Omran M, Benenson R, Schiele B (2015) Taking a deeper look at pedestrians. In: CVPR
43. Corinna C, Vapnik V (1995) Support-vector networks. *Mach Learn* 20(3):273–297
44. Freund Y, Schapire R (1995) A decision-theoretic generalization of on-line learning and an application to boosting. *Comput Learn Theory* 904:23–37
45. Breiman L (2001) Random forests. *Mach Learn* 45(1):5–32
46. Narasimhan H, Agarwal S (2013) SVM pAUC tight: a new support vector method for optimizing partial AUC based on a tight convex upper bound. In: SIGKDD. ACM. pp 167–175
47. Narasimhan H, Agarwal S (2013) A structural {SVM} based approach for optimizing partial AUC. In: ICML. pp 516–524
48. Paisitkriangkrai S, Shen C, van den Hengel A (2014) Pedestrian detection with spatially pooled features and structured ensemble learning. arXiv preprint arXiv:1409.5209
49. Felzenszwalb P, McAllester D, Ramanan D (2008) A discriminatively trained, multiscale, deformable part model. In: CVPR. pp 1–8
50. Everingham M, Eslami SMA, Van Gool L, Williams C, Winn J, Zisserman A (2015) The PASCAL Visual Object Classes challenge: a retrospective. *IJCV* 111(1):98–136
51. Girshick R, Iandola F, Darrell T, Malik J (2015) Deformable part models are convolutional neural networks. In: CVPR
52. Saberian MJ, Vasconcelos N (2012) Learning optimal embedded cascades. *PAMI* 34(10):2005–2012
53. Cai Z, Saberian M, N V (2015) Learning complexity-aware cascades for deep pedestrian detection. In: ICCV
54. Mao J, Xiao T, Jiang Y, Cao Z (2017) What can help pedestrian detection? In: CVPR
55. Massimo P. (2004) Background subtraction techniques: a review. In: Systems, man and cybernetics, 2004 IEEE international conference on. IEEE Vol. 4. pp 3099–3104
56. Wang L, Qiao Y, Tang X (2015) Action recognition with trajectory-pooled deep-convolutional descriptors. In: CVPR
57. Gkioxari G, Malik J (2015) Finding action tubes. In: CVPR
58. Shao J, Kang K, Loy CC, Wang X (2015) Deeply learned attributes for crowded scene understanding. In: CVPR
59. Cheron G, Laptev I, Schmid C (2015) P-CNN: pose-based CNN features for action recognition. In: ICCV
60. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, et al. (2015) ImageNet large scale visual recognition challenge. *Int J Comput Vis* 115(3):211–252
61. Zhu X, Xiong Y, Dai J, Yuan L, Wei Y (2017) Deep feature flow for video recognition. In: CVPR
62. Zhu X, Wang Y, Dai J, Yuan L, Wei Y (2017) Flow-guided feature aggregation for video object detection. In: ICCV
63. Feichtenhofer C, Pinz A, Zisserman A (2017) Detect to track and track to detect. In: ICCV
64. Trinh TT, Yoshihashi R, Kawakami R, Iida M, Naemura T (2016) Bird detection near wind turbines from high-resolution video using LSTM networks. In: World Wind Energy Conference (WVEC)
65. Yoshihashi R, Trinh TT, Kawakami R, You S, Iida M, Naemura T (2017) Differentiating objects by motion: joint detection and tracking of small flying objects. arXiv preprint arXiv:1709.04666
66. Pascanu R, Mikolov T, Bengio Y (2013) On the difficulty of training recurrent neural networks. In: ICML. pp 1310–1318
67. Goodale MA, Milner AD (1992) Separate visual pathways for perception and action. *Trends Neurosci.* 15(1):20–25
68. Glad S, Danelljan M, Khan FS, Felsberg M (2016) Deep motion features for visual tracking. In: ICPR. IEEE. pp 1243–1248
69. Johansson G. (1973) Visual perception of biological motion and a model for its analysis. *Percept. Psychophys.* 14(2):201–211
70. Bottou L (2012) Stochastic gradient descent tricks. In: Neural networks: Tricks of the trade. Springer. pp 421–436
71. Appel R, Fuchs T, Dollár P, Perona P (2013) Quickly boosting decision trees-pruning underachieving features early. In: ICML Vol. 28. pp 594–602
72. Donahue J, Jia Y, Vinyals O, Hoffman J, Zhang N, Tzeng E, Darrell T (2014) DeCAF: a deep convolutional activation feature for generic visual recognition. In: ICML
73. Lucas B, Kanade T (1981) An iterative image registration technique with an application to stereo vision. In: IJCAI Vol. 81. pp 674–679
74. Liu C, Yuen J, Torralba A, Sivic J, Freeman WT (2008) Sift flow: dense correspondence across different scenes. In: ECCV. pp 28–42
75. Weinzaepfel P, Revaud J, Harchaoui Z, Schmid C (2013) Deepflow: large displacement optical flow with deep matching. In: ICCV. pp 1385–1392
76. He K, Zhang X, Ren S, Sun J (2014) Spatial pyramid pooling in deep convolutional networks for visual recognition. In: ECCV. pp 346–361
77. Ren S, He K, Girshick R, Sun J (2015) Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems. pp 91–99
78. Dubout C, Fleuret F (2012) Exact acceleration of linear object detectors. In: ECCV. pp 301–311
79. Jia Y, Shelhamer E, Donahue J, Karayev S, Long J, Girshick R, Guadarrama S, Darrell T (2014) Caffe: Convolutional architecture for fast feature embedding. In: ACMMM. ACM. pp 675–678
80. Dollár P, Wojek C, Schiele B, Perona P (2009) Pedestrian detection: a benchmark. In: CVPR. pp 304–311
81. Enzweiler M, Gavriila DM (2009) Monocular pedestrian detection: survey and experiments. *PAMI* 31(12):2179–2195
82. Nam W, Dollár P, Han JH (2014) Local decorrelation for improved pedestrian detection. In: NIPS. pp 424–432
83. Marin J, Vázquez D, López AM, Amores J, Leibe B (2013) Random forests of local experts for pedestrian detection. In: Proceedings of the IEEE International Conference on Computer Vision. pp 2592–2599
84. Nam W, Han B, Han JH (2011) Improving object localization using macrofeature layout selection. In: ICCVW. IEEE. pp 1801–1808
85. Li J, Liang X, Shen S, Xu T, Yan S (2015) Scale-aware fast R-CNN for pedestrian detection. arXiv preprint arXiv:1510.08160
86. Zhang L, Lin L, Liang X, He K (2016) Is faster R-CNN doing well for pedestrian detection? In: ECCV. Springer. pp 443–457
87. Cai Z, Fan Q, Feris RS, Vasconcelos N (2016) A unified multi-scale deep convolutional neural network for fast object detection. In: ECCV. Springer. pp 354–370

Submit your manuscript to a SpringerOpen[®] journal and benefit from:

- Convenient online submission
- Rigorous peer review
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

Submit your next manuscript at ► springeropen.com