Pattern Analysis and Applications
https://doi.org/10.1007/s10044-019-00845-9

SHORT PAPER

Learning multi-scale features for foreground segmentation

Long Ang Lim[1] · Hacer Yalim Keles[1]

Received: 30 August 2018 / Accepted: 22 August 2019

Abstract
Foreground segmentation algorithms aim at segmenting moving objects from the background in a robust way under various challenging scenarios. Encoder–decoder-type deep neural networks that are used in this domain recently perform impressive segmentation results. In this work, we propose a variation of our formerly proposed method (Anonymous 2018) that can be trained end-to-end using only a few training examples. The proposed method extends the feature pooling module of FgSegNet by introducing fusion of features inside this module, which is capable of extracting multi-scale features within images, resulting in a robust feature pooling against camera motion, which can alleviate the need of multi-scale inputs to the network. Sample visualizations highlight the regions in the images on which the model is specially focused. It can be seen that these regions are also the most semantically relevant. Our method outperforms all existing state-of-the-art methods in CDnet2014 datasets by an average overall F-measure of 0.9847. We also evaluate the effectiveness of our method on SBI2015 and UCSD Background Subtraction datasets. The source code of the proposed method is made available at https://github.com/lim-anggun/FgSegNet_v2.

Keywords Foreground segmentation · Convolutional neural networks · Feature pooling module · Background subtraction · Video surveillance

1 Introduction

Extracting foreground objects from video sequences is one of the challenging and major tasks in computer vision domain. The resultant foreground objects, also known as moving objects, can be used in various computer vision tasks [5, 8, 12, 25, 26, 41]. Extracting objects of interests from stationary camera-videos is challenging, especially when the video sequences contain difficult scenarios such as sudden or gradual illumination changes, shadows, dynamic background motion, camera motion, camouflage or subtle regions. Several foreground segmentation approaches have been proposed to address these problems [4, 6, 15, 17, 33, 36, 42] where most of the proposed methods rely on building the stationary background model; this approach is not very effective in adapting to the challenging scenarios (Fig. 1).

Convolutional neural networks (CNNs) [20] based on gradient learning have been shown to be very powerful tools for the extraction of useful feature representations from data [40] and have been successfully used in many practical applications [1, 3, 18–20, 22, 27, 30]. In particular, fully convolutional networks (FCNs) that are based on transfer learning [1, 22] have shown significant improvement over conventional approaches by large margins. In this case, the knowledge that is gained from image classification problem is adapted to dense spatial classification problems (those in which each pixel in an image has to be marked with a class label); such a prediction requires an understanding of both higher-level and lower-level contextual information in a scene. However, due to feature resolution reduction caused by consecutive pooling and stridden convolution operations in the pre-trained models, this is usually difficult since the contextual details are lost. One may remove downsampling operations to keep high-resolution feature maps; however, it is computationally more expensive, and it is harder to expand the effective receptive fields. To take advantages of low-level features in large resolution, authors of [1] recently removed the last block of VGG-16 [30] and fine-tuned the last remaining block, and aggregated contextual information in multiple scales using a feature pooling module (FPM).

✉ Hacer Yalim Keles
hkeles@ankara.edu.tr

[1] Department of Computer Engineering, Ankara University, Ankara, Turkey

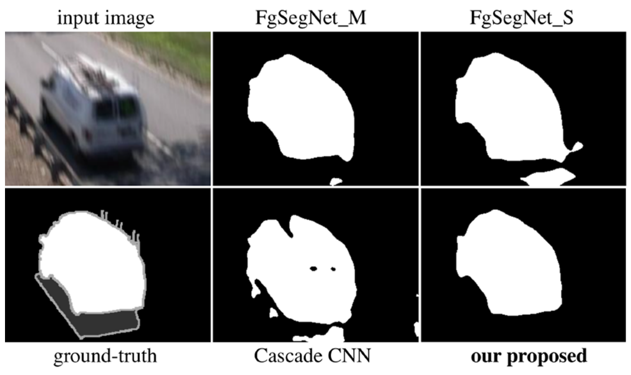

Fig. 1 A comparison between our method and some current state-of-the-art methods on *cameraJitter* category

Motivated by the recent success of deep neural networks for foreground segmentation, we adapt the same encoder that we previously used in work [1]; we found that this improves the performance compared to other pre-trained networks, and we propose some modifications on the original FPM module to capture wide-range multi-scale information, resulting in a more robust module against camera movements. In contrast to work [3] that transfers max-pooling indices from the encoder to the decoder and also differently to work [27] that copies feature maps directly from the encoder to the decoder to refine segmentation results, we use the global average pooling (GAP) from the encoder to guide the high-level features in the decoder part.

In summary, our key contributions are:

1. We propose a robust foreground segmentation network that can be trained using only a few training examples without incorporating temporal data; yet, it provides highly accurate segmentation results.
2. We improve the FPM module by fusing multi-scale features inside that module, resulting in a robust feature pooling against camera motion, which can alleviate the need of multi-scale inputs to the network.
3. We propose a novel decoder network, where high-level features in the decoder part are guided by low-level feature coefficients in the encoder part.
4. Our method can learn/segment faster than our former models and exceeds the state-of-the-art performances in Change Detection 2014 Challenge, SBI2015 and UCSD Background Subtraction datasets.
5. We provide visualizations of some layers in the reference models using the existing techniques [29, 31] and make our network more transparent for better interpretation; finally, we make the source code publicly available to facilitate future comparisons.

2 Related works

Foreground segmentation, also known as background subtraction, is one of the major tasks in computer vision. Various methods have been proposed in this domain. Most conventional approaches rely on building a background model for a specific video sequence. To model the background model, statistical (or parametric) methods using Gaussians are proposed [17, 33, 42] to model each pixel as a background or foreground pixel. However, parametric methods are computationally inefficient; to alleviate this problem, various nonparametric methods [4, 15, 36] are proposed. A different approach proposed in Ref. [6] consists of using genetic programing for selecting different change detection methods and for combining their results.

Recently, deep learning-based methods [1, 28, 38] show impressive results and outperform all classical approaches by large margins. There are different training strategies in this domain; for example, [2, 7] use patch-wise training strategy where the background patches and the image patches are combined, and then fed to CNNs to predict foreground probabilities of the center pixels of the patches. However, this approach is computationally inefficient and may cause overfitting due to redundant pixels and loss of higher context information within patches, and requires large number of patches in training. Different approaches to the problem like those in Refs. [1, 14, 21, 28, 38] approach the problem by using whole resolution images to the network to predict foreground masks. Some methods take advantages of temporal data (see References [21, 28]), whereas others train the networks by combining image frames with the generated background models, like those in Refs. [2, 7, 14, 21]. The number of training data that is utilized to produce the model is also different in different approaches; [1, 2, 7, 38] and [14] use 50%, 5%, 200 frames, 200 frames and 70% from each video sequence, respectively, where [28] splits video sequences into chunks of 10 frames and use 70% for training. In this research, we are generating a model for each scene and our purpose is to use only a few number of frames in training so that ground-truth requirement for different scenes will be very low. In agreement with the authors of [1, 38], we also consider this strategy is essential for a system that works in different domains in practice, since pixel-level ground-truth generation for large number of frames is a time-consuming and difficult process. Hence, we adapt the same training frames selection strategy as in [1, 38], where only 200 frames or less are used for training. This strategy mitigates user interventions on labeling ground-truths considerably (Fig. 2).

Dilated Convolution Dilated convolution, also known as atrous convolution, is a technique where convolution is

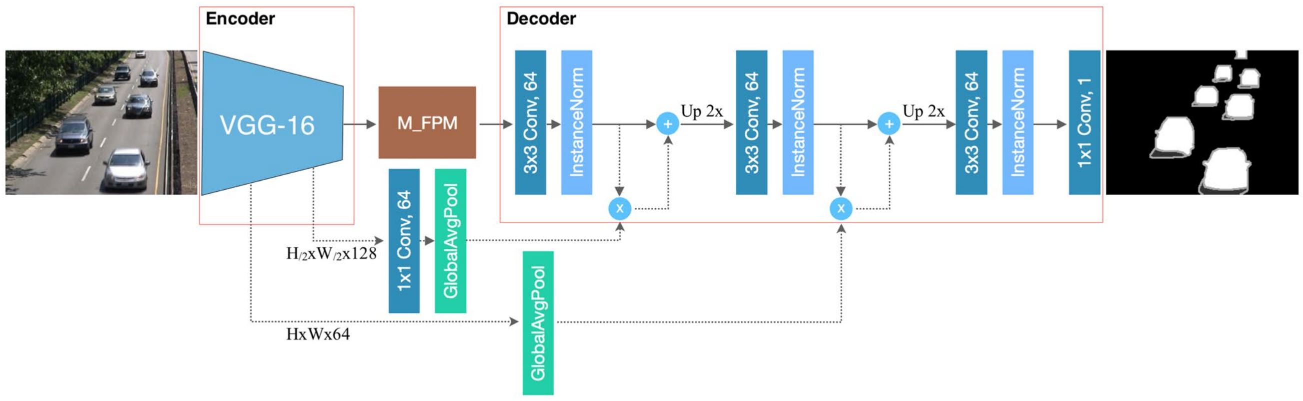

Fig. 2 The flow of FgSegNet_v2 architecture

computed by multiplying the filter coefficients in a spatially sparse way; this is done by the enlargement of the region that the filter is applied without introducing new coefficients to the filter. This convolution has been recently applied in semantic segmentation domain [9–11, 39]; the idea is to enlarge the field of views in the network without increasing the number of learnt parameters. Motivated by the recent success of the previous works, we proposed in [1] an FPM module with parallel dilated convolution layers that is plugged on top of a single-input encoder; it provides comparable foreground segmentation results compared to multi-input encoder.

2.1 The method

In this section, we revisit our previous work, FgSegNet [1], in both encoder and FPM module. For more details, one may refer to the original paper.

2.2 The encoder network

As authors of reference [30] do, we also use in FgSegNet the low-level features of the pre-trained VGG-16 net. We utilize the first four blocks of VGG-16 net by removing the last, i.e., fifth, block and third max-pooling layers, obtaining therefore higher-resolution feature maps. Dropout [32] layers are inserted after every convolutional layer of the modified net, and then, this block is fine-tuned. In this work, we also use the same encoder architecture as in the FgSegNet implementation. We observe that this modified net improves the performance compared to other pre-trained nets.

2.3 The modified FPM

Given feature maps F, which are obtained from the output of our encoder, the original FPM module [1] pools features at multiple scales by operating several convolutional layers with different dilation rates and a max-pooling layer followed by a 1×1 convolution on the same feature F, and then, the pooled features are concatenated along the depth dimension. Finally, the concatenated feature is passed through BatchNormalization [16] and SpatialDropout [34] layers. In this work, we improve the original FPM module by proposing some modifications to it in two parts (Fig. 3): (1) the resultant features f_a from a normal 3×3-conv are concatenated with the feature F and progressively pooled by a 3×3-conv with a dilation rate of 4, resulting in features f_b. Then, F and f_b are concatenated and fed to a 3×3-conv with dilation rate of 8, resulting in features f_c. Again, F and f_c are concatenated and fed to a 3×3-conv with dilation rate of 16. Finally, all features are concatenated to form 5x64 depth features, that we call it as F'; F' contains multi-scale features with wider receptive fields than those used in [1]. (2) We replace BatchNormalization with InstanceNormalization [35] since we empirically observe that InstanceNormalization gives slightly better performance with a small batch size.

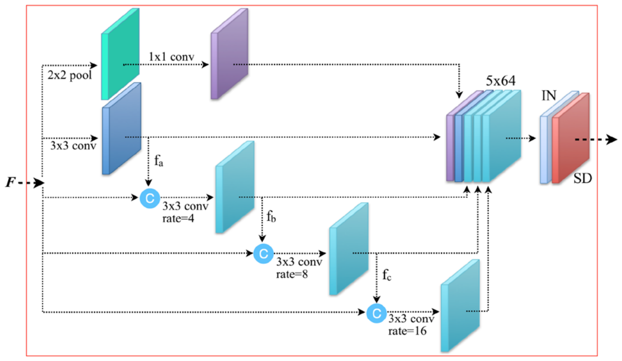

Fig. 3 The modified FPM module, M-FPM. IN (InstanceNormalization), SD (SpatialDropout). All convolution layers have 64 features

Since multiple pooling layers are operated on the same features F, the concatenated features F' are likely to be correlated. To promote independent feature maps, instead of the normal Dropout where individual neurons in a feature plane are dropped, SpatialDropout is used to drop the entire 2D feature maps by some rates, i.e., 0.25 in our implementation. When adjacent pixels within the feature maps are strongly correlated, spatial dropout improves the learning performance better than Dropout with small training sets [34]. We observe that using SpatialDropout helps to improve the performance and prevents overfitting in our network. Note that we apply rectified linear unit (ReLU) nonlinearity once right after InstanceNormalization in the M-FPM module. From now on, we will refer to this modified FPM shortly as M-FPM.

2.4 The decoder with GAP module

Our decoder network with two GAPs is illustrated in Fig. 2. The decoder part contains the stack of three 3×3-conv layers and a 1×1-conv layer, where the 3×3-conv layers followed by InstanceNormalization and the 1×1-conv layer are the projection from feature space to image space. All 3×3-conv layers have 64 feature maps, except the 1×1-conv layer that contains 1 feature slice. Note that ReLU nonlinearities are applied after InstanceNorm and sigmoid activation function is applied after 1×1-conv in the decoder part.

Global Average Pooling (GAP) There are two coefficient vectors that combine information from the low-level features of the encoder and high-level features of the decoder: The first one is pooled from the second convolution layer right before max-pooling layer, and the second one is pooled from the fourth layer. Since 3×3-conv layers have 64 features in the decoder part, the fourth convolution layer of the encoder is first projected from 128 to 64 features. Both coefficient vectors (say α_i) are multiplied with the output features of the first and second convolutional layers (say f_j^i) in the decoder part (see Fig. 2 for details). The scaled features are added with the original features to form features $f_j'^i$, where $f_j'^i : \alpha_i * f_j^i + f_j^i, i \in [0, 63]$ is the index of each feature depth and j is the index of an element in each feature slice. Finally, the concatenated $f_j'^i$ are upscaled by 2x using bilinear interpolation and fed to the next layers. We observed that despite adding very slight computational cost, the network with GAP module improved the overall performance.

2.5 Class visualization

We provide sample high-resolution images of visualizations (see Fig. 5) that depict the areas on which our model focuses more by adapting Grad-CAM [29] and guided backpropagation [31] methods. Note that Guided Grad-CAM is just the multiplication between Grad-CAM and guided backpropagation results. We conduct two experiments: First, we visualize the output features from the last layer (i.e., SpatialDropout layer) of M-FPM and FPM to compare the effectiveness of the two pooling models. Second, we visualize the last convolutional layer of the network, given the class logits (before sigmoid). Note that we pick the foreground class logits that have a threshold greater than or equal to 0.8.

3 Training protocol

We implement our models using Keras framework [13] with Tensorflow backend with a single NVIDIA GTX 970 GPU. We follow the same training procedure as [1]; hence, keeping the pre-trained coefficients of the original VGG-16 net, only the last modified block is fine-tuned. We train using *RMSProp* optimizer (setting *rho* to 0.9 and *epsilon* to 1e-08) with a *batch size* of 1. We use an initial learning rate of 1e-4, which is reduced by a factor of 10 when validation loss does not improve during in 5 consecutive epochs. The maximum number of epochs is set to 100, but training is stopped earlier if validation loss does not improve during 10 consecutive epochs. The training frames (e.g., 200 frames) are hardly shuffled before *training* + *validation* split and further split 80% for training and 20% for validation. The binary cross-entropy loss is used to make an agreement between true labels and predicted labels. Due to highly imbalanced pixels between background/foreground pixels in scenes, we alleviate the imbalanced data classification problem by giving more weight to samples in the less frequent class (foreground) and less weight to those in the more abundant class (background) during training. We compute the class-weights by using the foreground to background pixel ratios for each training frame, independently. Furthermore, since the output from the sigmoid activation is in range [0,1], we use them as the probability values; we apply thresholding (see Sect. 4 for the details) to the activations to obtain discrete binary class labels as foreground and background.

We mainly evaluate the model performance using F-measure and percentage of wrong classifications (PWC), where we want to maximize the F-measure, while minimizing the PWC. Given true positive (TP), false positive (FP), false negative (FN), true negative (TN), F-measure is defined by:

$$\text{F-measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{1}$$

where precision $= \frac{\text{TP}}{\text{TP+FP}}$, recall $= \frac{\text{TP}}{\text{TP+FN}}$. And PWC is defined by:

$$\text{PWC} = \frac{100 \times (\text{FP} + \text{FN})}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{2}$$

4 Results and discussion

In this section, we evaluate the effectiveness of our method using three different datasets, namely CDnet2014, SBI2015 and UCSD Background Subtraction. Each of these datasets contains challenging scenarios and used widely in fg/bg segmentation researches.

4.1 Experiments on CDnet2014 dataset

CDnet2014 dataset [37] contains 11 categories, where each category contains from 4 to 6 video sequences. Totally, there are 53 different video sequences in this dataset. The video sequence contains from 600 to 7999 frames with spatial resolutions varying from 320×240 to 720×576. Moreover, this dataset contains various challenging scenarios such as illumination change, hard shadow, highly dynamic background motion and camera motion etc.

Unlike most previous works that use a large number of training frames, to alleviate ground-truth labeling burden, authors in [1, 38] use only a few frames, i.e., 50 and 200 frames, respectively, for training plus validation. Similar to these works, we use the same training set (i.e., 200 frames) provided by [1] and we attempt to use a small number of training frames by randomly selecting 25 frames from the set of 200 frames to perform another experiment.

The Global Average Pooling Experiments In order to evaluate the effectiveness of GAP layers in the proposed network, we performed two sets of experiments by selecting the most challenging 6 categories from the CDnet2014 dataset (*cameraJitter, badWeather, dynamicBackground, intermittentObjectMotion, shadow, turbulence*); this subset of data totally contains 30 video sequences. The selected video sequences contain a number of frames that varies from 1150 to 7999. We use only 25 frames for *training* + *validation* (as mentioned above) and the remaining frames for *testing*.

In the first setting, we remove the GAP layers entirely from the network and will refer to this modified configuration as *no_GAP* below, and in the second setting we keep the GAP layers. As it can be seen from Table 1, the network with GAP improves over *no_GAP* in most categories by some margins. In particular, GAP improves over *no_GAP* by 2.34% points in *cameraJitter* category.

The Modified FPM (M-FPM) Experiments In this study, we demonstrate the effectiveness of the M-FPM module compared to the original FPM proposed by [1]. We again perform two set of experiments; in the first setting, we utilize the proposed decoder with the M-FPM module, while in the second, we use the proposed decoder with original FPM module. The experimental results are illustrated using a challenging scene, in Fig. 4, where previous networks produce many false positives. As it can be seen, (1) the proposed M-FPM module (Fig. 4c) produces less false positive compared to the original FPM module (Fig. 4d), (2) the proposed decoder (Fig. 4d) is effective compared to the FgSegNet family [1] decoder (Fig. 4f), (3) since *FgSegNet_M* [1]

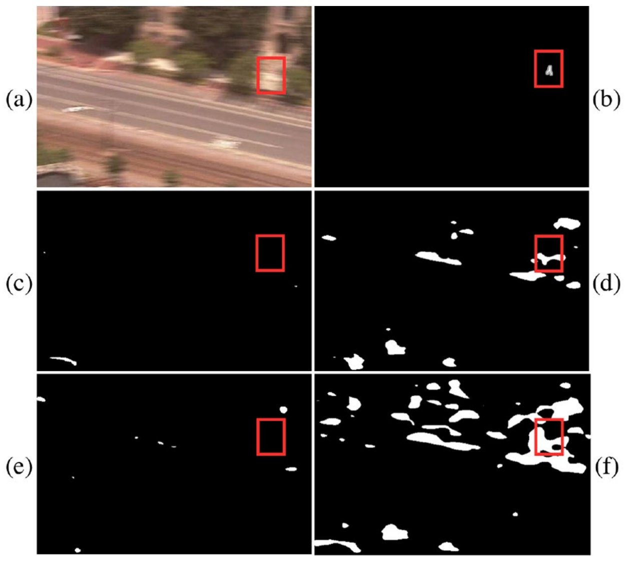

Fig. 4 The improved M_FPM module compared to the original FPM. **a** input image, **b** ground-truth, **c** M_FPM+*proposed decoder* result, **d** FPM+*proposed decoder* result, **e** FgSegNet_M result and **f** FgSegNet_S result

Table 1 The results with GAP and no_GAP

Category	no_GAP				GAP			
	F-measure		PWC		F-measure		PWC	
	25f	200f	25f	200f	25f	200f	25f	200f
cameraJit	0.9506	0.9936	0.3026	0.0436	0.9740	0.9936	0.2271	0.0438
badWeather	0.9781	0.9858	0.0657	0.0271	0.9783	0.9848	0.0639	0.0295
dynamicBg	0.9636	0.9878	0.0311	0.0052	0.9665	0.9881	0.0325	0.0054
intermitt	0.9597	0.9929	0.2997	0.0794	0.9735	0.9935	0.3268	0.0707
shadow	0.9840	0.9960	0.1265	0.0279	0.9853	0.9959	0.1159	0.0290
turbulence	0.9600	0.9779	0.0439	0.0230	0.9587	0.9762	0.0438	0.0232

fuses and jointly learns multi-input network features, it is robust to camera movement (Fig. 4e). As a comparison, we empirically observe that the M-FPM module can mitigate the need of multi-input network, which is computationally more expensive, by introducing the multi-scale feature fusion later instead, resulting in a robust feature pooling module. As it can be seen from Fig. 4, the proposed method (Fig. 4c) produces very less false positives compared to FgSegNet family (Fig. 4e, f) and improves over *FgSegNet_M*, *FgSegNet_S* and *Cascade CNN* [38] by 0.43%, 0.56% and 5.92% points, respectively, in *PTZ* category (see Table 3). We also show the effectiveness of the proposed M-FPM in Fig. 1.

Class Visualization First, we generate visualizations that highlight the focused regions of the encoder networks with the M-FPM and FPM in Fig. 5. As it can be seen, the model with FPM module is focusing on redundant parts, in a more random fashion, in the scene, leading to producing more false positives (Fig. 5, first row) compared to the model with M-FPM module (Fig. 5, second row). The focused regions in M-FPM indicate that M-FPM is more confident on the

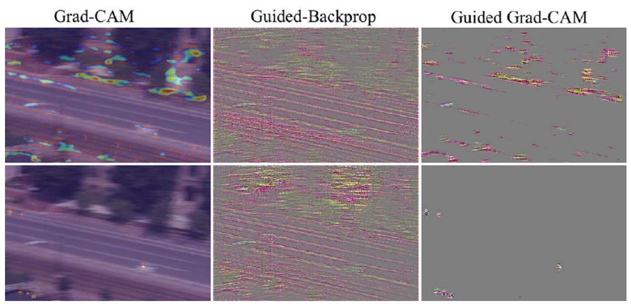

Fig. 5 The M-FPM versus FPM visualization. First row shows *Encoder + FPM* results, and second row shows *Encoder + M-FPM* results

regions that it looks at. We believe that M-FPM can learn semantically relevant features better by combining a cascade of increasing field of view with increased dilation rates in its convolutional layers.

Second, we generate sample visualizations of the entire network to analyze where the model is looking at (Fig. 6). We pick two sample scenes (from highway and traffic) for this visualization. As it can be seen (Fig. 6, first row), FgSegNet_v2 model focuses on the objects, i.e., cars, with higher scores with better coverages compared to FgSegNet_S and FgSegNet_M models; more red colors in the outputs correspond to higher class scores in those regions. The highlighted regions show that with the M-FPM and the direct feedback connections between the encoder and the decoder networks in the proposed method enabled learning robust contextual clues in the objects.

After evaluating the effectiveness of the extensions that we propose in this work with the challenging subset of CDnet2014 dataset, we perform further experiments by using the proposed architecture configuration (Fig. 2). As mentioned above, labeling the dense ground-truths requires more human efforts and to reduce labeling burden, as in Refs. [1, 38], we use only a few training examples. We adapt the same idea for 200-frame experiment; however, in this work, we further reduce the training examples by 8x to 25 frames. Specifically, we take two sets of experiments by using 25 frames and 200 frames and illustrate the *test results* in Table 2. As it can be seen, for 25-frame experiments, we obtain an overall F-measure of 0.9473 across 11 categories. By increasing the number of frames to 200, F-measure increases by 3.16% compared to the 25-frame experiment results. Similarly, the PWC decreases by 0.115% when we increase the number of frames from 25 to 200. Note that the training frames are not included in these evaluations and

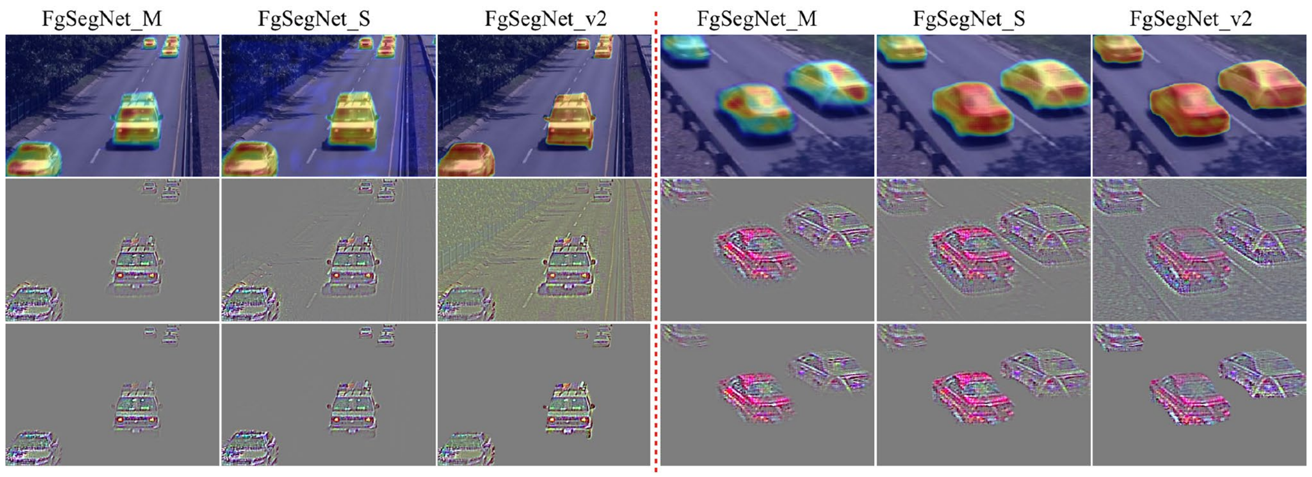

Fig. 6 Some comparisons among three methods. First row shows Grad-CAM results, second row shows Guided-Backprop results, and third row shows guided Grad-CAM results. Coverage with more red colors on the objects is the better

Table 2 The *test results* are obtained by manually and randomly selecting, 25 and 200 frames from CDnet2014 dataset across 11 categories

Category	FPR		FNR		Recall		Precision		PWC		F-measure	
	25f	200f	25f	200f	25f	200f	25f	200f	25f	200f	25f	200f
baseline	0.0002	0.00004	0.0100	0.0038	0.9900	0.9962	0.9942	0.9985	0.0480	0.0117	**0.9921**	**0.9974**
cameraJit	0.0004	0.00012	0.0419	0.0093	0.9581	0.9907	0.9907	0.9965	0.2271	0.0438	**0.9740**	**0.9936**
badWeather	0.0003	0.00009	0.0257	0.0215	0.9743	0.9785	0.9825	0.9911	0.0639	0.0295	**0.9783**	**0.9848**
dynamicBg	0.0001	0.00002	0.0315	0.0075	0.9685	0.9925	0.9655	0.9840	0.0325	0.0054	**0.9665**	**0.9881**
intermitt	0.0017	0.00015	0.0243	0.0104	0.9757	0.9896	0.9720	0.9976	0.3268	0.0707	**0.9735**	**0.9935**
lowFrameR.	0.0003	0.00008	0.2496	0.0956	0.7504	0.9044	0.7860	0.8782	0.1581	0.0299	**0.7670**	**0.8897**
nightVid.	0.0008	0.00022	0.1197	0.0363	0.8803	0.9637	0.9540	0.9861	0.3048	0.0802	**0.9148**	**0.9747**
PTZ	0.0002	0.00004	0.0870	0.0215	0.9130	0.9785	0.9776	0.9834	0.0892	0.0128	**0.9423**	**0.9809**
shadow	0.0003	0.0001	0.0203	0.0056	0.9797	0.9944	0.9911	0.9974	0.1159	0.0290	**0.9853**	**0.9959**
thermal	0.0009	0.00024	0.0456	0.0089	0.9544	0.9911	0.9815	0.9947	0.2471	0.0575	**0.9677**	**0.9929**
turbulence	0.0002	0.00011	0.0369	0.0221	0.9631	0.9779	0.9546	0.9747	0.0438	0.0232	**0.9587**	**0.9762**
Overall	**0.0005**	**0.0001**	**0.0630**	**0.0220**	**0.9370**	**0.9780**	**0.9591**	**0.9802**	**0.1507**	**0.0358**	**0.9473**	**0.9789**

Each row shows the average results of each category

Table 3 A comparison among 8 methods across 11 categories. Each row shows the results for each method

Methods	F-measure											Overall
	baseline	camJit	badWeat	dynaBg	intermit	lowFrame	nightVid	PTZ	shadow	thermal	turbul.	
FgSegNet_v2	**0.9980**	**0.9961**	0.9900	0.9950	0.9939	**0.9579**	0.9816	**0.9936**	0.9966	0.9942	**0.9815**	**0.9890**
FgSegNet_S [1]	**0.9980**	0.9951	**0.9902**	**0.9952**	**0.9942**	0.9511	**0.9837**	0.9880	**0.9967**	**0.9945**	0.9796	0.9878
FgSegNet_M [1]	0.9975	0.9945	0.9838	0.9939	0.9933	0.9558	0.9779	0.9893	0.9954	0.9923	0.9776	0.9865
Cascade CNN [38]	0.9786	0.9758	0.9451	0.9658	0.8505	0.8804	0.8926	0.9344	0.9593	0.8958	0.9215	0.9272
DeepBS [2]	0.9580	0.8990	0.8647	0.8761	0.6097	0.5900	0.6359	0.3306	0.9304	0.7583	0.8993	0.7593
IUTIS-5 [6]	0.9567	0.8332	0.8289	0.8902	0.7296	0.7911	0.5132	0.4703	0.9084	0.8303	0.8507	0.7820

Bold values indicate the best performing method for a category

Each column shows the average results in each category. Note that we consider *all the frames* in the ground-truths of CDnet2014 dataset. FgSegNet_M and FgSegNet_S [1], Cascade CNN [38], DeepBS [2], IUTIS-5 [6]

only the *test frames* are utilized. We performed a systematic analysis of the results for a range of threshold values and selected the threshold values with the best segmentation results for the test data. We selected to use a threshold of 0.7 for all 25-frame experiments and threshold of 0.9 for all 200-frame experiments.

We further compare the results between the proposed method and state-of-the-art methods in Table 3. Note that to make a comparison in terms of the number of frames (follow *changedetection.net*), we tested our model using all the provided ground-truths in CDnet2014 dataset. As it can be seen, our method (*FgSegNet_v2*) improves over current state-of-the-art methods by some margins. In particular, it significantly improves on camera motion categories (as discussed above), i.e., *PTZ* and *cameraJitter* category, where camera is shaking, panning, tilting or zooming around the scenes. This eliminates the trade-off of using multi-input features fusion in *FgSegNet_M*.

We further perform another experiment by using the training frames provided by [38] and evaluate our model on Change Detection 2014 Challenge (changedetection.net). The comparison is provided in Table 4. As it can be seen, our method outperformed the existing state-of-the-art methods by some margins; specifically, for the deep learning-based methods, it improves over *FgSegNet_S*, *FgSegNet_M*, *Cascade CNN* and *DeepBS* by 0.43%, 0.77%, 6.38% and 23.89% points, respectively. It also improves over all traditional

Table 4 A comparison with the state-of-the-art methods. These average results are obtained from Change Detection 2014 Challenge

Methods	Overall			
	Precision	Recall	PWC	F-measure
FgSegNet_v2	**0.9823**	**0.9891**	**0.0402**	**0.9847**
FgSegNet_S [1]	0.9751	0.9896	0.0461	0.9804
FgSegNet_M [1]	0.9758	0.9836	0.0559	0.9770
Cascade CNN [38]	0.8997	0.9506	0.4052	0.9209
DeepBS [2]	0.8332	0.7545	1.9920	0.7458
IUTIS-5 [6]	0.8087	0.7849	1.1986	0.7717

methods over 21.3% points. Our method is ranked as number 1 at the time of submission.

In order to display the generalization capability of all methods with some exemplary frames from the dataset, we provide segmentation results in two ways; first, we choose some segmentation results randomly in the range of the provided ground-truths (i.e., *intermitObjMotion* and *thermal* categories), and second, we randomly chose some example frames from the *test set* where ground-truths are not publicly shared (i.e., *PTZ, nightVideo* and *badWeather* categories). As it can be seen from Fig. 7, our method gives good segmentation results, especially in *intermitObjMotion* category where a car stopped in the scene for a long time and starts moving immediately. This is the case where most methods fail dramatically in this scenario (e.g., method in (g)). As for *test results* where ground-truths are not available in Fig. 7 (*PTZ, nightVideo, badWeather* categories), our method generalizes well to completely unseen data where challenging content of the scenes change over time. Our method produces good segmentation masks compared to other methods.

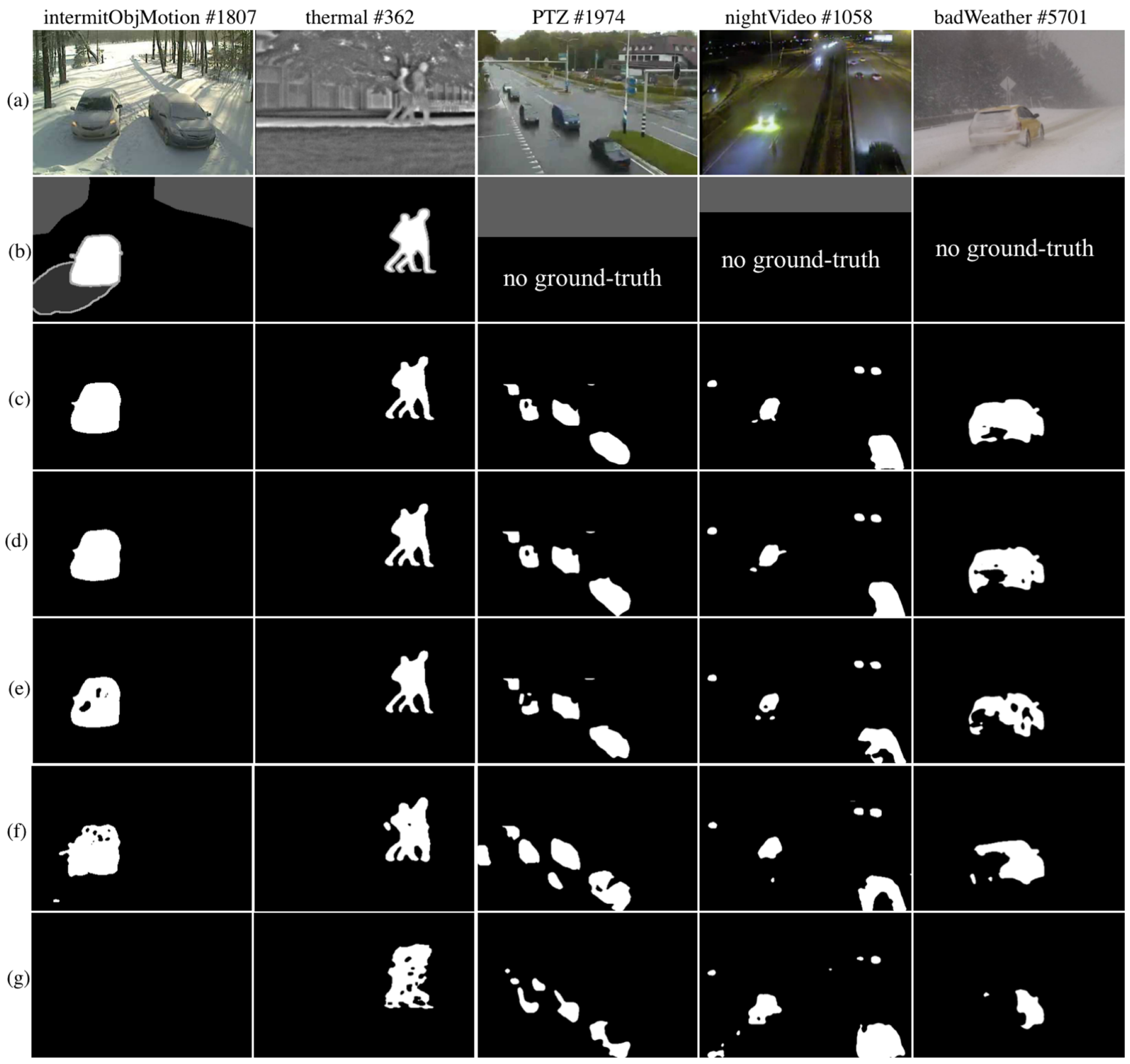

Fig. 7 Some comparisons among 5 methods. **a** Input images, **b** ground-truths or ROI, **c** our segmentation result, **d** FgSegNet_S result, **e** FgSegNet_M result, **f** Cascade CNN result and **g** DeepBS results

Our method performs better in almost all categories, except *LowFrameRate* category where it performs poorly (F-measure = 0.9579) compared to other categories (see Table 3). This low performance is primarily due to a challenging video sequence (in *lowFrameRate* category), where there are extremely small foreground objects in dynamic scenes with gradual illumination changes (Fig. 8). In this case, the network may pay more attention to the major class (bg) but less attention to the rare class (fg), resulting in misclassifying very small foreground objects. However, the proposed method still improves over the best method by some margins in this category. Furthermore, our method fails to detect blended objects into the scene (see Fig. 4); in this case, an object is blended to the background completely and it is even hard for human to distinguish between foreground and background. In addition, FgSegNet_v2 is capable of segmenting about 23fps compared to FgSegNet_S (21fps) and FgSegNet_M (18fps) on a single NVIDIA GTX 970 GPU.

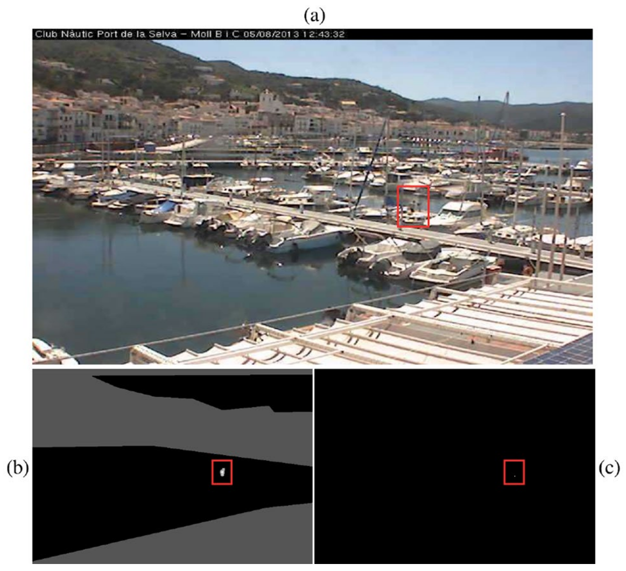
(a)
Club Nàutic Port de la Selva – Moll B i C 05/08/2013 12:43:32
(b)
(c)

Fig. 8 The video sequence in *lowFrameRate* category that our method performs poorly. **a** Shows input image, **b** shows ground-truth, and **c** shows our segmentation result, respectively

4.2 Experiments on SBI2015 dataset

We conduct additional experiments on Scene Background Initialization 2015 (SBI2015) dataset [23], which contains 14 video sequences with ground-truth labels provided by [38]. We follow the same training protocol as in [1, 38], where 20% of the frames are used for *training* + *validation* and the remaining 80% split for testing; if we denote the number of training examples used in training as t_n, in these experiments $t_n \in [2, 148]$.

The *test results* are illustrated in Table 5. As it can be seen, our method outperformed previous state-of-the-art methods by some margins. To be concrete, it improves over FgSegNet family by 0.22% and 0.59% points, while it improves over *Cascade CNN* [38] by 9.21% points. Similarly, the PWC of our method is significantly less compared to other methods. We obtain the lowest performance in *Toscana* sequence with an F-measure of 0.9291; this is primarily due to using very low number of training frames: only 2 frames, for *training* + *validation*. We depict some exemplary results in Fig. 9. As it can be seen, our method

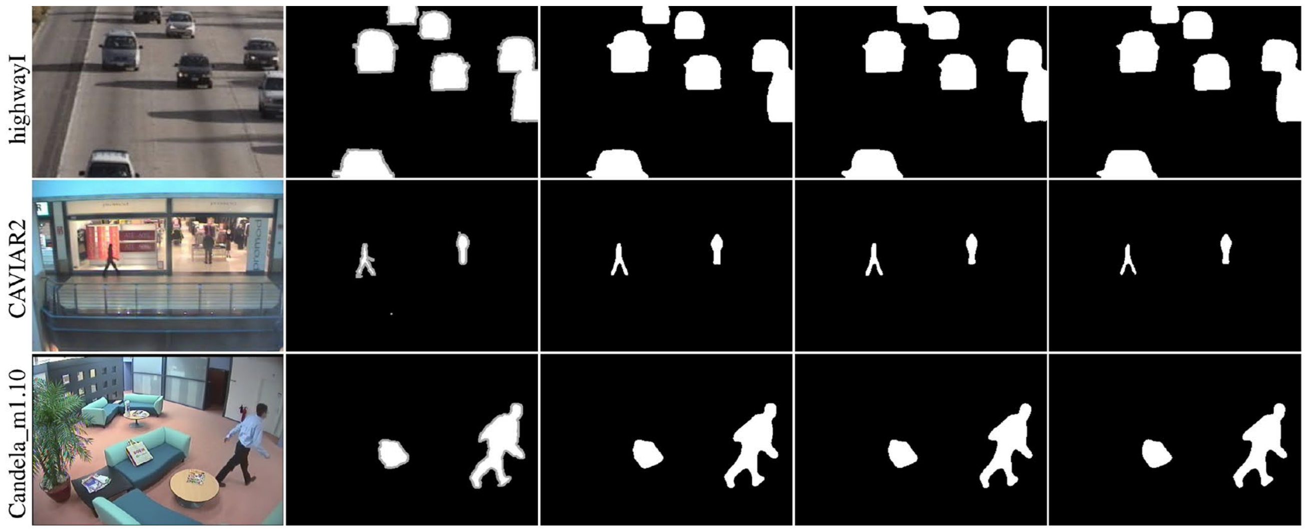

Fig. 9 A comparison on SBI2015 dataset. Each column shows raw images, the ground-truths, our segmentation results, FgSegNet_S and FgSegNet_M results, respectively

Table 5 The *test results* on SBI2015 dataset with threshold of 0.3 and some comparisons with state-of-the-art methods

Video Seq.	FPR	FNR	F-measure	PWC
Board	0.0009	0.0019	0.9979	0.1213
Candela_m1.10	0.0003	0.0037	0.9950	0.0399
CAVIAR1	0.0001	0.0007	0.9988	0.0086
CAVIAR2	0.0001	0.0092	0.9834	0.0131
CaVignal	0.0027	0.0076	0.9859	0.3310
Foliage	0.0771	0.0207	0.9732	3.7675
HallAndMonitor	0.0002	0.0051	0.9926	0.0357
HighwayI	0.0008	0.0068	0.9924	0.1358
HighwayII	0.0001	0.0051	0.9952	0.0289
HumanBody2	0.0009	0.0079	0.9920	0.1636
IBMtest2	0.0007	0.0220	0.9817	0.1680
PeopleAndFoliage	0.0066	0.0102	0.9919	0.8468
Snellen	0.0211	0.0147	0.9644	1.7573
Toscana	0.0046	0.1155	0.9291	2.5901
FgSegNet_v2	**0.0083**	**0.0165**	**0.9853**	**0.7148**
FgSegNet_S [1]	**0.0090**	**0.0146**	**0.9831**	**0.8524**
FgSegNet_M [1]	**0.0059**	**0.0310**	**0.9794**	**0.9431**
Cascade CNN [38]	–	–	**0.8932**	**5.5800**

produces good segmentation masks, especially in *highwayI* video sequence where hard shadows are eliminated completely.

4.3 Experiments on UCSD dataset

Similarly to [1], we further evaluate our method on UCSD Background Subtraction dataset [24], which contains 18 video sequences with ground-truth labels. This dataset contains highly dynamic backgrounds, which are extremely challenging, and the number of frames is relatively small compared to CDnet2014 and SBI2015 datasets. We use the same training/testing splits provided by [1], where, first, the 20% split is used for training, i.e., $t_n \in [3, 23]$, and 80% for testing; second, 50% is used for training, i.e., $t_n \in [7, 56]$, and remaining 50% for testing. *Test results* are depicted in Table 6. Although there are very small numbers of *training + validation* frames, we obtain an average F-measure of 0.8945 in case of 20% split and 0.9203 in case of 50% split. Our method produces comparable results to the previous methods, while PWC decreases remarkably compared to the previous methods. We also depict some segmentation results in Fig. 10.

Table 6 The *test results* on UCSD dataset with threshold of 0.6 and some comparisons with state-of-the-art methods

Video Seq.	20% split				50% split			
	FPR	FNR	F-measure	PWC	FPR	FNR	F-measure	PWC
Birds	0.0025	0.1423	0.8649	0.5205	0.0021	0.1162	0.8884	0.4315
Boats	0.0009	0.0729	0.9213	0.1678	0.0008	0.0382	0.9437	0.1213
Bottle	0.0014	0.0406	0.9550	0.2462	0.0009	0.0476	0.9605	0.2134
Chopper	0.0023	0.0760	0.9140	0.3991	0.0017	0.0810	0.9232	0.3544
Cyclists	0.0030	0.0738	0.9213	0.5382	0.0019	0.0488	0.9492	0.3457
Flock	0.0048	0.0641	0.9383	0.9270	0.0036	0.0375	0.9591	0.6179
Freeway	0.0013	0.3012	0.7787	0.5480	0.0028	0.1349	0.8394	0.4635
Hockey	0.0197	0.0716	0.9165	2.8430	0.0138	0.0527	0.9400	2.0359
Jump	0.0066	0.0746	0.9358	1.4309	0.0041	0.0464	0.9603	0.8892
Landing	0.0007	0.0653	0.9245	0.1278	0.0006	0.0559	0.9388	0.1031
Ocean	0.0012	0.1014	0.8931	0.2253	0.0010	0.0607	0.9243	0.1615
Peds	0.0048	0.0955	0.8776	0.7499	0.0038	0.0907	0.8942	0.6402
Rain	0.0029	0.1180	0.9174	1.0490	0.0025	0.0558	0.9534	0.5881
Skiing	0.0022	0.0846	0.9171	0.4338	0.0018	0.0586	0.9385	0.3251
Surfers	0.0015	0.1147	0.8887	0.2990	0.0012	0.0764	0.9173	0.2232
Surf	0.0008	0.2941	0.7307	0.1778	0.0005	0.2321	0.7968	0.1343
Traffic	0.0017	0.0899	0.9070	0.3186	0.0015	0.0566	0.9301	0.2427
Zodiac	0.0003	0.0735	0.8988	0.0429	0.0002	0.0815	0.9086	0.0383
FgSegNet_v2	**0.0033**	**0.1086**	**0.8945**	**0.6136**	**0.0025**	**0.0762**	**0.9203**	**0.4405**
FgSegNet_S [1]	**0.0058**	**0.0559**	**0.8822**	**0.7052**	**0.0039**	**0.0544**	**0.9139**	**0.5024**
FgSegNet_M [1]	**0.0037**	**0.0904**	**0.8948**	**0.6260**	**0.0027**	**0.0714**	**0.9203**	**0.4637**

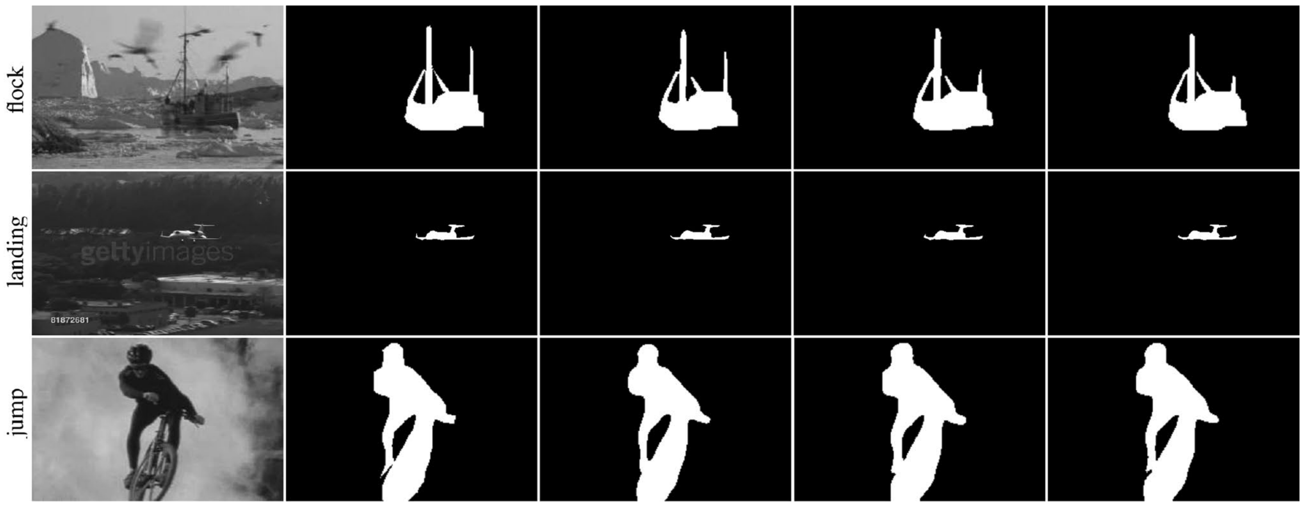

Fig. 10 A comparison on UCSD dataset. Each column shows raw images, the ground-truths, our segmentation results, FgSegNet_S and FgSegNet_M results, respectively

5 Conclusion

In this work, we propose a robust encoder–decoder network, which can be trained end-to-end in a supervised manner. Our network is simple, yet can learn accurate foreground segmentation by using a few training examples, which alleviates the need of ground-truth labeling burden. We improve the original FPM module by fusing multiple scale features inside FPM module, resulting in a robust module against camera motion, which can alleviate the need for training the network with multi-scale inputs. We further propose a simple decoder, which can help in improving the performance. Our method neither requires any post-processing to refine the segmentation results nor temporal data into consideration. The experimental results reveal that our network outperforms the existing state-of-the-art methods in several benchmarks. We also provide visualizations to depict that our network is looking at the foreground object more precisely compared to our former networks. As a future work, we plan to incorporate temporal data and redesign a method, which can learn from very small number of examples.

Acknowledgements We would like to thank the anonymous reviewers for their valuable suggestions and comments.

References

1. Lim LA, Keles HY (2018) Foreground segmentation using convolutional neural networks for multiscale feature encoding. Pattern Recogn Lett 112:256–262
2. Babaee M, Dinh DT, Rigoll G (2017) A deep convolutional neural network for background subtraction. arXiv preprint arXiv:1702.01731
3. Badrinarayanan V, Kendall A, Cipolla R (2015) Segnet: A deep convolutional encoder-decoder architecture for image segmentation. arXiv preprint arXiv:1511.00561
4. Barnich O, Van Droogenbroeck M (2011) Vibe: a universal background subtraction algorithm for video sequences. IEEE Trans Image Process 20(6):1709–1724
5. Basharat A, Gritai A, Shah M (2008) Learning object motion patterns for anomaly detection and improved object detection. In: IEEE conference on computer vision and pattern recognition, 2008. CVPR 2008. IEEE, pp 1–8
6. Bianco S, Ciocca G, Schettini R (2017) How far can you get by combining change detection algorithms? In: International conference on image analysis and processing. Springer, Berlin, pp 96–107
7. Braham M, Van Droogenbroeck M (2016) Deep background subtraction with scene-specific convolutional neural networks. In: 2016 International conference on systems, signals and image processing (IWSSIP). IEEE, pp 1–4
8. Brutzer S, Höferlin B, Heidemann G (2011) Evaluation of background subtraction techniques for video surveillance. In: 2011 IEEE conference on computer vision and pattern recognition (CVPR). IEEE, pp 1937–1944
9. Chen LC, Papandreou G, Kokkinos I, Murphy K, Yuille AL (2018) Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE Trans Pattern Anal Mach Intell 40(4):834–848
10. Chen LC, Papandreou G, Schroff F, Adam H (2017) Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587
11. Chen LC, Zhu Y, Papandreou G, Schroff F, Adam H (2018) Encoder-decoder with atrous separable convolution for semantic image segmentation. arXiv preprint arXiv:1802.02611
12. Cheung SCS, Kamath C (2004) Robust techniques for background subtraction in urban traffic video. Proc SPIE 5308:881–892
13. Chollet F, et al (2015) Keras. https://keras.io. Accessed 29 Aug 2019
14. Cinelli LP, Thomaz LA, da Silva AF, da Silva EA, Netto SL (2017) Foreground segmentation for anomaly detection in surveillance videos using deep residual networks. In: Proceedings

XXXV Brazilian communication signal processing symposium, pp 914–918
15. Hofmann M, Tiefenbacher P, Rigoll G (2012) Background segmentation with feedback: the pixel-based adaptive segmenter. In: 2012 IEEE computer society conference on computer vision and pattern recognition workshops (CVPRW). IEEE, pp 38–43
16. Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167
17. KaewTraKulPong P, Bowden R (2002) An improved adaptive background mixture model for real-time tracking with shadow detection. Video-based Surveill Syst 1:135–144
18. Karpathy A, Fei-Fei L (2015) Deep visual-semantic alignments for generating image descriptions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3128–3137
19. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, NIPS 2012, 3–8 Dec 2012. Nevada, USA, pp 1097–1105
20. LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324
21. Lim K, Jang WD, Kim CS (2017) Background subtraction using encoder-decoder structured convolutional neural network. In: 2017 14th IEEE international conference on advanced video and signal based surveillance (AVSS). IEEE, pp 1–6
22. Long J, Shelhamer E, Darrell T (2015) Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3431–3440
23. Maddalena L, Petrosino A (2015) Towards benchmarking scene background initialization. In: International conference on image analysis and processing. Springer, Berlin, pp 469–476
24. Mahadevan V, Vasconcelos N (2010) Spatiotemporal saliency in dynamic scenes. IEEE Trans Pattern Anal Mach Intell 32(1):171–177. https://doi.org/10.1109/TPAMI.2009.112
25. Poppe R (2010) A survey on vision-based human action recognition. Image Vis Comput 28(6):976–990
26. Porikli F, Tuzel O (2003) Human body tracking by adaptive background models and mean-shift analysis. In: IEEE international workshop on performance evaluation of tracking and surveillance, pp 1–9
27. Ronneberger O, Fischer P, Brox T (2015) U-net: Convolutional networks for biomedical image segmentation. In: International conference on medical image computing and computer-assisted intervention. Springer, Berlin, pp 234–241
28. Sakkos D, Liu H, Han J, Shao L (2017) End-to-end video background subtraction with 3d convolutional neural networks. Multimed Tools Appl 77(17):23023–23041
29. Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D (2017) Grad-cam: visual explanations from deep networks via gradient-based localization. In: 2017 IEEE international conference on computer vision (ICCV). IEEE, pp 618–626
30. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556
31. Springenberg JT, Dosovitskiy A, Brox T, Riedmiller M (2014) Striving for simplicity: the all convolutional net. arXiv preprint arXiv:1412.6806
32. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958
33. Stauffer C, Grimson WEL (1999) Adaptive background mixture models for real-time tracking. In: IEEE computer society conference on computer vision and pattern recognition, 1999, vol 2. IEEE, pp 246–252
34. Tompson J, Goroshin R, Jain A, LeCun Y, Bregler C (2015) Efficient object localization using convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 648–656
35. Ulyanov D, Vedaldi A, Lempitsky VS (2016) Instance normalization: the missing ingredient for fast stylization. CoRR **abs/1607.08022**
36. Van Droogenbroeck M, Paquot O (2012) Background subtraction: experiments and improvements for vibe. In: 2012 IEEE computer society conference on computer vision and pattern recognition workshops (CVPRW). IEEE, pp 32–37
37. Wang Y, Jodoin PM, Porikli F, Konrad J, Benezeth Y, Ishwar P (2014) Cdnet 2014: an expanded change detection benchmark dataset. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp 387–394
38. Wang Y, Luo Z, Jodoin PM (2017) Interactive deep learning method for segmenting moving objects. Pattern Recognit Lett 96(Supplement C):66–75. https://doi.org/10.1016/j.patrec.2016.09.014
39. Yu F, Koltun V (2015) Multi-scale context aggregation by dilated convolutions. arXiv preprint arXiv:1511.07122
40. Zeiler MD, Fergus R (2014) Visualizing and understanding convolutional networks. In: European conference on computer vision. Springer, Berlin, pp 818–833
41. Zhu S, Xia L (2015) Human action recognition based on fusion features extraction of adaptive background subtraction and optical flow model. Math Probl Eng 2015:387–464
42. Zivkovic Z (2004) Improved adaptive gaussian mixture model for background subtraction. In: Proceedings of the 17th international conference on pattern recognition, 2004. ICPR 2004, vol 2. IEEE, pp 28–31