# BSUV-Net: A Fully-Convolutional Neural Network for Background Subtraction of Unseen Videos

M. Ozan Tezcan, Prakash Ishwar, Janusz Konrad [*]
Department of Electrical and Computer Engineering
Boston University, Boston, MA
[mtezcan, pi, jkonrad]@bu.edu

## Abstract

*Background subtraction is a basic task in computer vision and video processing often applied as a pre-processing step for object tracking, people recognition, etc. Recently, a number of successful background-subtraction algorithms have been proposed, however nearly all of the top-performing ones are supervised. Crucially, their success relies upon the availability of some annotated frames of the test video during training. Consequently, their performance on completely "unseen" videos is undocumented in the literature. In this work, we propose a new, supervised, background-subtraction algorithm for unseen videos (BSUV-Net) based on a fully-convolutional neural network. The input to our network consists of the current frame and two background frames captured at different time scales along with their semantic segmentation maps. In order to reduce the chance of overfitting, we also introduce a new data-augmentation technique which mitigates the impact of illumination difference between the background frames and the current frame. On the CDNet-2014 dataset, BSUV-Net outperforms state-of-the-art algorithms evaluated on unseen videos in terms of several metrics including F-measure, recall and precision.*

## 1. Introduction

Background subtraction (BGS) is a foundational, low-level task in computer vision and video processing. The aim of BGS is to segment an input video frame into regions corresponding to either foreground or background. It is frequently used as a pre-processing step for higher-level tasks such as object tracking, people and motor-vehicle recognition, human activity recognition, etc. Since BGS is often the first pre-processing step, the accuracy of its output has an overwhelming impact on the overall performance of subsequent steps. Therefore, it is critical that BGS produce as accurate a foreground/background segmentation as possible.

*This work was supported in part by ARPA-E under agreement DE-AR0000944 and by the donation of Titan GPUs from NVIDIA Corp.

Traditional BGS algorithms are unsupervised and rely on a background model to predict foreground regions [24, 34, 10, 18, 3, 22, 23, 13, 14]. PAWCS [22], SWCD [13] and WisenetMD [14] are considered to be state-of-the-art unsupervised BGS algorithms. However, since they rely on the accuracy of the background model, they encounter difficulties when applied to complex scenes. Recently, ensemble methods and a method leveraging semantic segmentation have been proposed which significantly outperform traditional algorithms [4, 31, 7].

The success of deep learning in computer vision did not bypass BGS research [6]. A number of *supervised* deep-learning BGS algorithms have been developed [8, 1, 2, 30, 28, 15, 16, 20] with performance easily surpassing that of traditional methods. However, most of these algorithms have been tuned to either one specific video or to a group of similar videos, and their performance on unseen videos has not been evaluated. For example, FgSegNet [15] uses 200 frames from a test video for training and the remaining frames from the *same* video for evaluation. If applied to an unseen video, its performance drops significantly (Section 4.3).

In this paper, we introduce *Background Subtraction for Unseen Videos* (BSUV-Net), a fully-convolutional neural network for predicting foreground of an *unseen* video. A key feature of our approach is that the training and test sets are composed of frames originating from different videos. This guarantees that no ground-truth data from the test videos have been shown to the network in the training phase. By employing two reference backgrounds at different time scales, BSUV-Net addresses two challenges often encountered in BGS: varying scene illumination and intermittently-static objects that tend to get absorbed into the background. We also propose novel data augmentation which further improves our method's performance under varying illumination. Furthermore, motivated by recent work on the use of semantic segmentation in BGS [7], we improve our method's accuracy by inputting semantic information along with the reference backgrounds and current frame. The main contributions of our work are as follows:

1. **Supervised BGS for Unseen Videos:** Although supervised algorithms, especially neural networks, have significantly improved BGS performance, they are tuned to a specific video and thus their performance on *unseen* videos deteriorates dramatically. To the best of our knowledge, BSUV-Net is the first supervised BGS algorithm that is truly generalizable to *unseen* videos.
2. **Data Augmentation for Increased Resilience to Varying Illumination:** Changes in scene illumination pose a major challenge to BGS algorithms. To mitigate this, we develop a simple, yet effective, data augmentation technique. Using a simple additive model we vary the illumination of the current frame and the reference background frames that are fed into BSUV-Net during training. This enables us to effectively tackle various illumination change scenarios that may be present in test videos.
3. **Leveraging Semantic and Multiple Time-Scale Information:** BSUV-Net improves foreground-boundary segmentation accuracy by accepting semantic information as one of its inputs. This is unlike in an earlier BGS method [7] which used semantic information as a *post-processing* step. The other network inputs are the current frame (to be segmented) and a two-frame background model with data from different time scales. While one background frame, based on *distant* history, helps with the discovery of intermittently-static objects, the other frame, based on *recent* history, is key for handling dynamic factors such as illumination changes.

Based on our extensive experiments on the CDNet-2014 dataset [11], BSUV-Net outperforms state-of-the-art BGS algorithms evaluated on *unseen* videos

## 2. Related Work

A wide range of BGS algorithms have been developed in the past, each having some advantages and disadvantages over others. Since this is not a survey paper, we will not cover all BGS variants. Instead, we will focus only on recent top-performing methods. We divide these algorithms into 3 categories: (i) BGS by (unsupervised) background modeling, (ii) supervised BGS tuned to a single video or a group of videos, (iii) Improving BGS algorithms by post-processing.

### 2.1. BGS by Background Modeling

Nearly all traditional BGS algorithms first compute a background model, and then use it to predict the foreground. While a simple model based on the mean or median of a subset of preceding frames offers only a single background value per pixel, a probabilistic Gaussian Mixture Model (GMM) [24] allows a range of background values. This idea was improved by creating an online procedure for the update of GMM parameters in a pixel-wise manner [34]. Kernel Density Estimation (KDE) was introduced into BGS [10] as a non-parametric alternative to GMMs and was subsequently improved [18]. The probabilistic methods achieve better performance compared to single-value models for dynamic scenes and scenes with small background changes.

In [3], Barnich and Droogenbroeck introduced a sample-based background model. Instead of implementing a probabilistic model, they modeled the background by a set of sample values per pixel and used a distance-based model to decide whether a pixel should be classified as background or foreground. Since color information alone is not sufficient for complex cases, such as illumination changes, Bilodeau *et al.* introduced Local Binary Similarity Patterns (LBSP) to compare the current frame and background using spatio-temporal features instead of color [5]. St-Charles *et al.* combined color and texture information, and introduced a word-based approach, PAWCS [22]. They considered pixels as background words and updated each word's reliability by its persistence. Similarly, SuBSENSE by St-Charles *et al.* [23] combines LBSP and color features, and employs pixel-level feedback to improve the background model.

Recently, Isik *et al.* introduced SWCD, a pixel-wise, sliding-window approach leveraging a dynamic control system to update the background model [13], while Lee *et al.* introduced WisenetMD, a multi-step algorithm to eliminate false positives in dynamic backgrounds [14]. In [25], Sultana *et al.* introduced an unsupervised background estimation method based on a generative adversarial network (GAN). They use optical flow to create a motion mask and then in-paint covered regions with background values estimated by a GAN. The foreground is then computed by subtracting the estimated background from the current frame followed by morphological operations. They, however, do not achieve state-of-the-art results. Zeng *et al.* introduced RTSS [29] which uses deep learning-based semantic segmentation predictions to improve the background model used in SubSENSE [23].

### 2.2. Supervised BGS

Although background subtraction has been extensively studied in the past, the definition of a supervised BGS algorithm is still vague. Generally speaking, the aim of a supervised BGS algorithm is to learn the parameters (e.g., neural-network weights) of a complex function in order to minimize a loss function of the labeled training frames. Then, the performance of the algorithm is evaluated on a separate set of test frames. In this section we divide the supervised BGS algorithms into three groups namely, *video-optimized*, *video-group-optimized* and *video-agnostic* depending on which frames and videos they use during training and testing.

Several algorithms use some frames from a test video for training and all the frames of the *same* video for evaluating performance on that video. In such algorithms, parameter values are optimized separately for each video. We will refer to this class of algorithms as *video-optimized* BGS algorithms. In another family of algorithms, randomly-selected frames from a *group* of test videos are used for training and all the frames of the same videos are used for testing. Since some frames from *all* test videos are used for training, we will refer this class of algorithms as *video-group-optimized* algorithms. Note that, in both of these scenarios the algorithms are neither optimized for nor evaluated on *unseen* videos and to the best of our knowledge all of the top-performing supervised BGS algorithms to-date are either *video-optimized* or *video-group-optimized*. In this paper, we introduce a new category of supervised BGS algorithms, called *video-agnostic* algorithms, that can be applied to unseen videos with no or little loss of performance. To learn parameters, a *video-agnostic* algorithm uses frames from a set of training videos but for performance evaluation it uses a completely different set of videos.

In recent years, supervised learning algorithms based on convolutional neural networks (CNNs) have been widely applied to BGS. The first CNN-based BGS algorithm was introduced in [8]. This is a *video-optimized* algorithm which produces a single foreground probability for the center of each $27 \times 27$ patch of pixels. A method proposed in [28] uses a similar approach, but with a modified CNN which operates on patches of size $31 \times 31$ pixels.

Instead of using a patch-wise algorithm, Zeng and Zhu introduced the Multiscale Fully-Convolutional Neural Network (MFCN) which can predict the foreground of the entire input image frame in one step [30]. Lim and Keles proposed a triplet CNN which uses siamese networks to create features at three different scales and combines these features within a transposed CNN [15]. In a follow-up work, they removed the triplet networks and used dilated convolutions to capture the multiscale information [16]. In [2], Bakkay *et al.* used generative adversarial networks for BGS. The generator performs the BGS task, whereas the discriminator tries to classify the BGS map as real or fake. Although all these algorithms perform very well on various BGS datasets, it is important to note that they are all *video-optimized*, thus they will suffer a performance loss when tested on unseen videos. In [1], Babae *et al.* designed a *video-group-optimized* CNN for BGS. They randomly selected $5\%$ of CDNet-2014 frames [11] as a training set and developed a single network for all of the videos in this dataset. In [20], Sakkos *et al.* used a 3D CNN to capture the temporal information in addition to the color information. Similarly to [1], they trained a single algorithm using 70% of frames in CDNet-2014 and then used it to predict the foreground in all videos of the dataset. Note that even these approaches do not generalize to other videos since some ground truth data from each video exists in the training set. Table 1 compares and summarizes the landscape of supervised BGS algorithms and the methodology used for training and evaluation.

As discussed above, none of the CNN-based BGS algorithms to-date have been designed for or tested on unseen videos with no ground truth at all. This limits their practical utility since it is not possible to label some frames in *every* new video. Since the publication of our preprint [26], we have learned about a recent BGS algorithm named $3DFR$ [17], which uses $3D$ spatio-temporal convolution blocks in an encoder-decoder architecture to predict background in an unseen video. However, [17] only reports evaluation results on 10 out of the 53 videos of CDNet2014.

### 2.3. Improving BGS Algorithms by Post-Processing

Over the last few years, many deep-learning-based algorithms were developed for the problem of semantic segmentation and they achieved state-of-the-art performance. In [7], Braham and Droogenbroeck introduced a post-processing step for BGS algorithms based on semantic segmentation predictions. Given an input frame, they predicted a segmentation map using PSPNet [32] and obtained pixel-wise probability predictions for semantic labels such as person, car, animal, house etc. Then, they manually grouped these labels into two sets – foreground and background labels, and used this information to improve any BGS algorithm's output in a post-processing step. They obtained very competitive results by using SubSENSE [23] as the BGS algorithm.

Bianco *et al.* introduced an algorithm called IUTIS which combines the results produced by several BGS algorithms [4]. They used genetic programming to determine how to combine several BGS algorithms' outputs using a sequence of basic binary operations, such as logical "and/or", majority voting and median filtering. Their best result was achieved by using 5 top-performing BGS algorithms on the CDNet-2014 dataset at the time of publication. Zeng *et al.* followed the same idea, but instead of genetic programming, used a fully-convolutional neural network to fuse several BGS results into a single output [31], and outperformed IUTIS on CDNet-2014.

## 3. Proposed Algorithm: BSUV-Net

### 3.1. Inputs to BSUV-Net

Segmenting an unseen video frame into foreground and background regions without using any information about the background would be an ill-defined problem. In BSUV-Net, we use two reference frames to characterize the background. One frame is an "empty" background frame, with no people or other objects of interest, which can typically be extracted from the beginning of a video e.g., *via* median filtering over a large number of *initial* frames. This provides an accurate reference that is very helpful for segmenting intermittently-static objects in the foreground. However, due to dynamic factors, such as illumination variations, this reference may

Table 1. Training/evaluation methodologies of supervised BGS algorithms on CDNet-2014.

| Algorithm | Are some frames from test videos used in training? | | Training and evaluation methodology |
|---|---|---|---|
| Braham-CNN-BGS [8] | Yes | First half of the labeled frames of the test video | *video-optimized* |
| MFCNN [30] | Yes | Randomly selected 200 frames from the first 3000 labeled frames of the test video | *video-optimized* |
| Wang-CNN-BGS [28] FGSegNet [15, 16] BScGAN [2] | Yes | Hand picked 200 labeled frames of the test video | *video-optimized* |
| Babae-CNN-BGS [1] | Yes | 5% of the labeled frames of all videos | *video-group-optimized* |
| 3D-CNN-BGS [20] | Yes | 70% of the labeled frames of all videos | *video-group-optimized* |
| **BSUV-Net** (proposed) | No | No frame from test videos is used in training | *video-agnostic* |

not be valid after some time. To counteract this, we use another reference frame that characterizes *recent* background, for example by computing median of 100 frames preceding the frame being processed. However, this frame might not be as accurate as the first reference frame since we cannot guarantee that there will be no foreground objects in it (if such objects are present for less than 50 frames, the temporal median will suppress them). By using two reference frames captured at different time scales, we aim to leverage benefits of each frame type.

Braham *et al.* [7] have shown that leveraging results of semantic segmentation significantly improves the performance of a BGS algorithm, for example by using semantic segmentation results in a post-processing step. In BSUV-Net, we follow a different idea and use semantic information as an additional input channel to our neural network. In this way, we let our network learn how to use this information. To extract semantic segmentation information, we used a state-of-the-art CNN called DeepLabv3 [9] trained on ADE20K [33], an extensive semantic-segmentation dataset with 150 different class labels and more than 20,000 images with dense annotations. Let us denote the set of object classes in ADE20K as $C = \{c_0,\ c_1,\ \dots,\ c_{149}\}$. Following the same procedure as in [7], we divided these classes into two sets: foreground and background objects. As foreground objects, we used person, car, cushion, box, book, boat, bus, truck, bottle, van, bag and bicycle. The rest of the classes are used as background objects. The softmax layer of DeepLabv3 provides pixel-wise class probabilities $p_{c_j}$ for $c_j \in C$. Let $\mathbf{I}[m,n]$ be an input frame at spatial location $m,n$ and let $\{p_{c_j}[m,n]\}_{j=0}^{149}$ be the predicted probability distribution of $\mathbf{I}[m,n]$. We compute a foreground probability map (FPM) $\mathbf{S}[m,n] = \sum_{c_j \in F} p_{c_j}[m,n]$, where $F$ is the set of foreground classes.

We use the current, recent and empty frames in color, each along with its FPM, as the input to BSUV-Net (Figure 1). Clearly, the number of channels in BSUV-Net's input layer is 12 for each frame consists of 4 channels (R,G,B,FPM).

## 3.2. Network Architecture and Loss Function

We use a UNET-type [19] fully-convolutional neural network (FCNN) with residual connections. The architecture of BSUV-Net has two parts: encoder and decoder, and is shown in Figure 1. In the encoder network, we use $2 \times 2$ max-pooling operators to decrease the spatial dimensions. In the decoder network, we use up-convolutional layers (transposed convolution with a stride of 2) to increase the dimensions back to those of the input. In all convolutional and up-convolutional layers, we use $3 \times 3$ convolutions as in VGG [21]. The residual connections from the encoder to the decoder help the network combine low-level visual information gained in the initial layers with high-level visual information gained in the deeper layers. Since our aim is to increase the performance on unseen videos, we use strong batch normalization (BN) [12] and spatial dropout (SD) [27] layers to increase the generalization capacity. Specifically, we use a BN layer after each convolutional and up-convolutional layer, and an SD layer before each max-pooling layer. Since our task can be viewed as a binary segmentation, we use a sigmoid layer as the last layer in BSUV-Net. The operation of the overall network can be defined as a nonlinear map $\mathbf{G}(\mathbf{W}) : \mathbf{X} \rightarrow \widehat{\mathbf{Y}}$ where $\mathbf{X} \in \mathbb{R}^{w \times h \times 12}$ is a 12-channel input, $w$ and $h$ are its spatial dimensions, $\mathbf{W}$ represents the parameters of neural network $\mathbf{G}$, and $\widehat{\mathbf{Y}} \in [0,1]^{w \times h}$ is a pixel-wise foreground probability prediction. Note that since this is a fully-convolutional neural network, it does not require fixed input size; any frame size can be used, but some padding may be needed to account for max-pooling operations.

In most BGS datasets, the number of background pixels is much larger than the number of foreground pixels. This class imbalance creates significant problems for the commonly-used loss functions, such as cross-entropy and mean-squared error. A good alternative for unbalanced binary datasets is the Jaccard index. Since the network output is a probability map, we opted for a relaxed form of the Jaccard index as the loss function, defined as follows:

$$J_R(\mathbf{Y},\widehat{\mathbf{Y}}) = \frac{T + \sum\limits_{m,n} (\mathbf{Y}[m,n]\widehat{\mathbf{Y}}[m,n])}{T + \sum\limits_{m,n} \Big( \mathbf{Y}[m,n] + \widehat{\mathbf{Y}}[m,n] - \mathbf{Y}[m,n]\widehat{\mathbf{Y}}[m,n] \Big)}$$

where $\mathbf{Y} \in \{0,1\}^{w \times h}$ is the ground truth of $\mathbf{X}$, $T$ is a smoothing parameter and $m$, $n$ are the spatial locations.

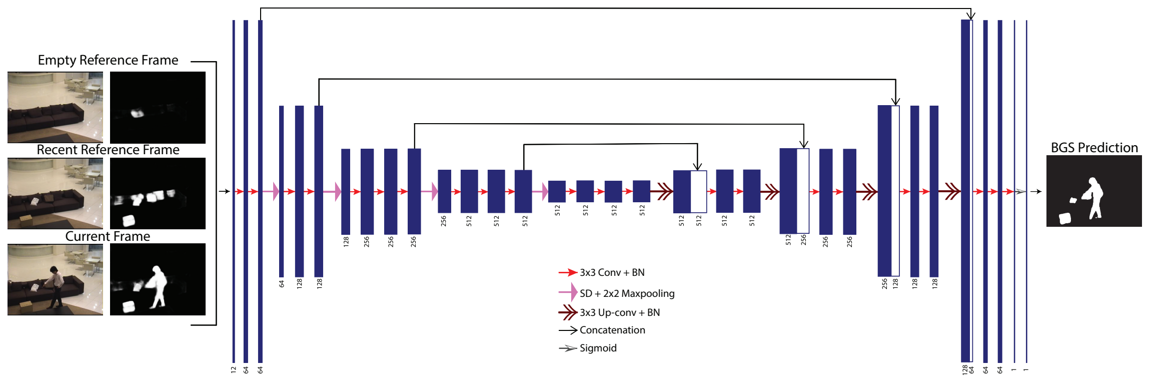

Figure 1. Network architecture of BSUV-Net. BN stands for batch normalization and SD stands for spatial dropout. Grayscale images at the network input show foreground probability maps (FPM) of the corresponding RGB frames.

### 3.3. Resilience to Illumination Change by Data Augmentation

Since neural networks have millions of parameters, they are very prone to overfitting. A widely-used method for reducing overfitting in computer-vision problems is to enlarge the dataset by applying several data augmentations such as random crops, rotations and noise addition. Since we are dealing with videos in this paper, we can also add augmentation in the temporal domain.

In real-life BGS problems, there might be a significant illumination difference between an empty background frame acquired at an earlier time and the current frame. However, only a small portion of videos in CDnet-2014 capture significant illumination changes which limits BSUV-Net's generalization performance. Therefore, we introduce a new data-augmentation technique to account for global illumination changes between the empty reference frame and the current frame. Suppose that $\mathbf{R_E} \in \mathbb{R}^{w,h,3}$ represents the RGB channels of an empty reference frame. Then, an augmented version of $\mathbf{R_E}$ can be computed as $\widehat{\mathbf{R}}_\mathbf{E}[m,n,c] = \mathbf{R_E}[m,n,c] + \mathbf{d}[c]$ for $c = 1,2,3$, where $\mathbf{d} \in \mathbb{R}^3$ represents a frame-specific global RGB change in our illumination model. By choosing $\mathbf{d}$ randomly for each example during training (see Section 4.2 for details), we can make the network resilient to illumination variations.

## 4. Experimental Results

### 4.1. Dataset and Evaluation Metrics

In order to evaluate the performance of BSUV-Net, we used CDNet-2014 [11], the largest BGS dataset with 53 natural videos from 11 categories including challenging scenarios such as shadows, night videos, dynamic background, etc. The spatial resolution of videos varies from $320 \times 240$ to $720 \times 526$ pixels. Each video has a region of interest labelled as either 1) foreground, 2) background, 3) hard shadow or 4) unknown motion. When measuring an algorithm's performance, we ignored pixels with unknown motion label and considered hard-shadow pixels as background. Our treatment of hard shadows is consistent with what is done in CDNet-2014 for the change-detection task.

In CDNet-2014 [11], the authors propose seven binary performance metrics to cover a wide range of BGS cases: recall ($Re$), specificity ($Sp$), false positive rate ($FPR$), false negative rate ($FNR$), percentage of wrong classification ($PWC$), precision ($Pr$) and F-measure ($F_1$). They also introduced two ranking-based metrics namely "average ranking" ($R$) and "average ranking accross categories" ($R_{cat}$) which combine all 7 metrics into ranking scores. The details of these rankings can be found at "changedetection.net".

### 4.2. Training and Evaluation Details

As discussed in Section 2.2, we use a *video-agnostic* evaluation methodology in all experiments. This allows us to measure an algorithm's performance on real-world-like tasks when no ground-truth labels are available. To evaluate BSUV-Net performance on all videos in CDNet-2014, we used 18 different combinations of training/test video sets. The splits are structured in such a manner that every video appears in the test set of exactly one split, but when it does so, it does not appear in the training set for that split. Detailed information about these sets can be found in the supplementary material. Let us denote the $m$-th combination by $(V^m_{train}, V^m_{test})$. Then, $\cup^{18}_{m=1} V^m_{test}$ is equal to the set of all 53 videos in CDNet-2014. During training, we used 200 frames suggested in [30] for each video in $V^m_{train}$.

When training on different sets $V^m_{train}$, we used *exactly the same* hyperparameter values across all sets to make sure that we are not tuning our network to specific videos. In all of our experiments, we used ADAM optimizer with a learning rate of $10^{-4}$, $\beta_1 = 0.9$, and $\beta_2 = 0.99$. The minibatch size was 8 and we trained for 50 epochs. As the empty background frame, we used the median of all foreground-free frames within the first 100 frames. In a few videos containing highway traffic, the first 100 frames did not contain a single foreground-free frame. For these videos, we hand-picked empty frames (e.g., in groups) and used their median as the empty reference. Although this may seem like a limitation, in practice one can randomly sample several hundred frames at the same time of the day across several days (similar illumination) and median filter them to obtain an empty background frame (due to random selection, a moving object is unlikely to occupy the same location in more than 50% of frames). Since there is no single empty background frame in videos from the pan-tilt-zoom (PTZ) category, we slightly changed the inputs. Instead of "empty background + recent background" pair we used "recent background + more recent background" pair, where the recent background is computed as the median of 100 preceding frames and the more recent background is computed as the median of 30 preceding frames.

Although BSUV-Net can accept frames of any spatial dimension, we used a *fixed* size of $224 \times 224$ pixels (randomly cropped from the input frame) so as to leverage parallel GPU processing in the training process. We applied random data augmentation at the beginning of each epoch. For illumination resilience, we used the data augmentation method of Section 3.3 with $\mathbf{d}[c] = \mathbf{I} + \mathbf{I}_c$, where $\mathbf{I}$ is the same for all $c$ and sampled from $\mathcal{N}(0, 0.1^2)$, while $\mathbf{I}_c$ is independently sampled from $\mathcal{N}(0, 0.04^2)$ for each $c$. We also added random Gaussian noise from $\mathcal{N}(0, 0.01^2)$ to each pixel in each color channel. For pixel values, we used double precision numbers that lie between 0 and 1.

In the evaluation step, we did not apply any scaling or cropping to the inputs. To obtain binary maps, we applied thresholding with threshold $\theta = 0.5$ to the output of the sigmoid layer of BSUV-Net.

## 4.3. Quantitative Results

Table 2 compares BSUV-Net against state-of-the-art BGS algorithms in terms of the seven metrics and two rankings discussed in Section 4.1. All quantitative results shown in this paper are computed by "changedetection.net" evaluation servers to reflect the real performance on test data. Since BSUV-Net is *video-agnostic*, comparing it with *video-optimized* or *video-group-optimized* algorithms would not be fair and we omit them. Instead, we compare BSUV-Net with state-of-the-art unsupervised algorithms, namely SWCD [13], WisenetMD [14] and PAWCS [22] , which, by definition, are *video-agnostic*. We exclude RTSS [29] and $3DFR$ [17] in Table 2 since their results on the test frames are not available on "changedetection.net". We include the results of IUTIS-5 [4] and SemanticBGS [7], but we list them separately because these are *post-processing* algorithms. Note that, both IUTIS-5 and SemanticBGS can be applied to any BGS algorithm from Table 2, including BSUV-Net, to improve its performance. To show this, we also report the result of BSUV-Net post-processed by SemanticBGS. In the *self-contained algorithms* category, we also list FgSegNet v2 [16] since it is currently the best performing algorithm on CDNet-2014. However, since FGSegNet v2's performance reported on "changedetection.net" has been obtained in a *video-optimized* manner, we trained it anew in a *video-agnostic* manner using the same methodology that we used for BSUV-Net. As expected, this caused a huge performance decrease of FgSegNet v2 compared to it's *video-optimized* training. As is clear from Table 2, BSUV-Net outperforms its competitors on almost all of the metrics. The F-measure performance demonstrates that BSUV-Net achieves excellent results without compromising either recall or precision. Table 2 also shows that the performance of BSUV-Net can be improved even further by combining it with SemanticBGS. The combined algorithm outperforms all of the *video-agnostic* algorithms that are available on "changedetection.net".

Table 3 compares the per-category F-measure performance of BSUV-Net against state-of-the-art BGS algorithms. For RTSS [29], the values of performance metrics shown in Table 3 are as reported in their paper. Columns 2-12 report the F-measure for each of the 11 categories from "changedetection.net", while the last column reports the mean F-measure across all categories. Similarly to Table 2, we divided this table into *post-processing* and *self-contained algorithms*. It can be observed that BSUV-Net achieves the best performance in 5 out of 11 categories. It has a striking performance advantage in the "night" category. All videos in this category are traffic-related and many cars have headlights turned on at night which causes significant local illumination variations in time. BSUV-Net's excellent performance in this category demonstrates that the proposed model is indeed largely illumination-invariant.

BSUV-Net performs poorer than other algorithms in "camera jitter" and "dynamic background" categories. We believe this is related to the empty and recent background frames we are using as input. The median operation used to compute background frames creates very blurry images for these categories since the background is not static. Thus, BSUV-Net predicts some pixels in the background as foreground and increases the number of false positives.

## 4.4. Visual Results

A visual comparison of BSUV-Net with SWCD [13] and WisenetMD [14] is shown in Figure 2. Each column shows a sample frame from one of the videos in one of the 8 categories. It can be observed that BSUV-Net produces visually the best results for almost all categories.

In the "night" category, SWCD and WisenetMD produce many false positives because of local illumination changes. BSUV-Net produces better results since it is designed to be illumination-invariant. In the "shadow" category, BSUV-Net performs much better in the shadow regions. Results in the "intermittent object motion" and "baseline" categories show that BSUV-Net can successfully detect intermittently-static

Table 2. Comparison of methods for **unseen videos** from CDNet-2014. For fairness, we separated the post-processing and self-contained algorithms.

| Method | $R$ | $R_{cat}$ | $Re$ | $Sp$ | $FPR$ | $FNR$ | $PWC$ | $Pr$ | $F_1$ |
|---|---|---|---|---|---|---|---|---|---|
| *Post-processing algorithms* | | | | | | | | | |
| **BSUV-net** + SemanticBGS* | **9.00** | 14.00 | **0.8179** | 0.9944 | 0.0056 | **0.1821** | 1.1326 | **0.8319** | **0.7986** |
| IUTIS-5* + SemanticBGS* | 9.43 | 11.45 | 0.7890 | **0.9961** | **0.0039** | 0.2110 | **1.0722** | 0.8305 | 0.7892 |
| IUTIS-5* | 11.43 | **10.36** | 0.7849 | 0.9948 | 0.0052 | 0.2151 | 1.1986 | 0.8087 | 0.7717 |
| *Self-contained algorithms* | | | | | | | | | |
| **BSUV-net** | **9.29** | **13.18** | **0.8203** | 0.9946 | 0.0054 | **0.1797** | **1.1402** | **0.8113** | **0.7868** |
| SWCD | 15.43 | 19.00 | 0.7839 | 0.9930 | 0.0070 | 0.2161 | 1.3414 | 0.7527 | 0.7583 |
| WisenetMD | 16.29 | 15.18 | 0.8179 | 0.9904 | 0.0096 | 0.1821 | 1.6136 | 0.7535 | 0.7668 |
| PAWCS | 14.00 | 15.45 | 0.7718 | **0.9949** | **0.0051** | 0.2282 | 1.1992 | 0.7857 | 0.7403 |
| FgSegNet v2 | 44.57 | 44.09 | 0.5119 | 0.9411 | 0.0589 | 0.4881 | 7.3507 | 0.4859 | 0.3715 |

Table 3. Comparison of methods according to the per-category F-measure for **unseen videos** from CDNet-2014.

| Method | Bad weather | Low framerate | Night | PTZ | Thermal | Shadow | Int. obj. motion | Camera jitter | Dynamic backgr. | Base-line | Turbu-lence | Overall |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *Post-processing algorithms* | | | | | | | | | | | | |
| **BSUV-net** + SemanticBGS* | **0.8730** | 0.6788 | **0.6815** | **0.6562** | **0.8455** | **0.9664** | 0.7601 | 0.7788 | 0.8176 | **0.9640** | 0.7631 | **0.7986** |
| IUTIS-5* + SemanticBGS* | 0.8260 | 0.7888 | 0.5014 | 0.5673 | 0.8219 | 0.9478 | **0.7878** | **0.8388** | **0.9489** | 0.9604 | 0.6921 | 0.7892 |
| IUTIS-5* | 0.8248 | **0.7743** | 0.5290 | 0.4282 | 0.8303 | 0.9084 | 0.7296 | 0.8332 | 0.8902 | 0.9567 | **0.7836** | 0.7717 |
| *Self-contained algorithms* | | | | | | | | | | | | |
| **BSUV-net** | **0.8713** | 0.6797 | **0.6987** | **0.6282** | **0.8581** | 0.9233 | 0.7499 | 0.7743 | 0.7967 | **0.9693** | 0.7051 | 0.7868 |
| RTSS | 0.8662 | 0.6771 | 0.5295 | 0.5489 | 0.8510 | **0.9551** | **0.7864** | **0.8396** | **0.9325** | 0.9597 | 0.7630 | **0.7917** |
| SWCD | 0.8233 | **0.7374** | 0.5807 | 0.4545 | **0.8581** | 0.8779 | 0.7092 | 0.7411 | 0.8645 | 0.9214 | 0.7735 | 0.7583 |
| WisenetMD | 0.8616 | 0.6404 | 0.5701 | 0.3367 | 0.8152 | 0.8984 | 0.7264 | 0.8228 | 0.8376 | 0.9487 | **0.8304** | 0.7535 |
| PAWCS | 0.8152 | 0.6588 | 0.4152 | 0.4615 | 0.8324 | 0.8913 | 0.7764 | 0.8137 | 0.8938 | 0.9397 | 0.6450 | 0.7403 |
| FgSegNet v2 | 0.3277 | 0.2482 | 0.2800 | 0.3503 | 0.6038 | 0.5295 | 0.2002 | 0.4266 | 0.3634 | 0.6926 | 0.0643 | 0.3715 |

objects. It is safe to say that BSUV-Net is capable of simultaneously handling the discovery of intermittently-static objects and also the dynamic factors such as illumination change.

An inspection of results in the “dynamic background” category shows that BSUV-Net has detected most of the foreground pixels but failed to detect the background pixels around the foreground objects. We believe this is due to the blurring effect of the median operation that we used in the computation of background frames. Using more advanced background models as an input to BSUV-Net might improve the performance in this category.

## 4.5. Ablation Study

One of the contributions of BSUV-Net is its multi-channel input composed of two background frames from different time scales and a foreground probability map (FPM). Another contribution is temporal data augmentation tailored to handling illumination changes. In Table 4, we explore their impact on precision, recall and F-measure. Each column on the left represents one characteristic and each row represents a different combination of these characteristics. RGB channels of the current frame are used in all of the combinations. “Empty BG” and “Recent BG” refer to the use of empty and\or recent background frames, respectively, in addition to the current frame. “Data aug.” refers to temporal data augmentation described in Section 3.3 and “FPM” refers to the use of semantic FPM channel in addition to the RGB channels for all input frames. It is clear that all these characteristics have a significant impact on the overall performance. Using only the current frame as input results in very poor metrics. The introduction of empty or/and recent background frames leads to a significant improvement. Adding temporal data augmentation or/and FPM channels further improves the performance and the final network achieves state-of-the-art results.

Table 4. Impact of background frames, data augmentation for temporal illumination change and FPM on BSUV-Net performance.

| Current frame | Empty BG | Recent BG | Data aug. | FPM | $Pr$ | $Re$ | $F_1$ |
|---|---|---|---|---|---|---|---|
| ✓ | | | | | 0.3615 | 0.5509 | 0.3476 |
| ✓ | ✓ | | | | 0.6994 | 0.7686 | 0.6819 |
| ✓ | | ✓ | | | 0.6976 | 0.7064 | 0.6351 |
| ✓ | ✓ | ✓ | | | 0.7658 | 0.7606 | 0.7156 |
| ✓ | ✓ | ✓ | ✓ | | 0.7574 | 0.8159 | 0.7447 |
| ✓ | ✓ | ✓ | | ✓ | 0.7807 | 0.7747 | 0.7450 |
| ✓ | ✓ | ✓ | ✓ | ✓ | 0.8113 | 0.8203 | 0.7868 |

In this paper, we proposed to add semantic FPM channel as input in order to improve our algorithm’s performance. However, if the background and foreground object categories are chosen carefully, FPM can be used as a BGS algorithm by itself. This would require prior information about the video (to compute FPM) and, therefore, would not qualify as a *video-agnostic* method. In our algorithm, however, we combine FPM information with RGB channels and background frames. When applying DeepLabv3 [7] to compute

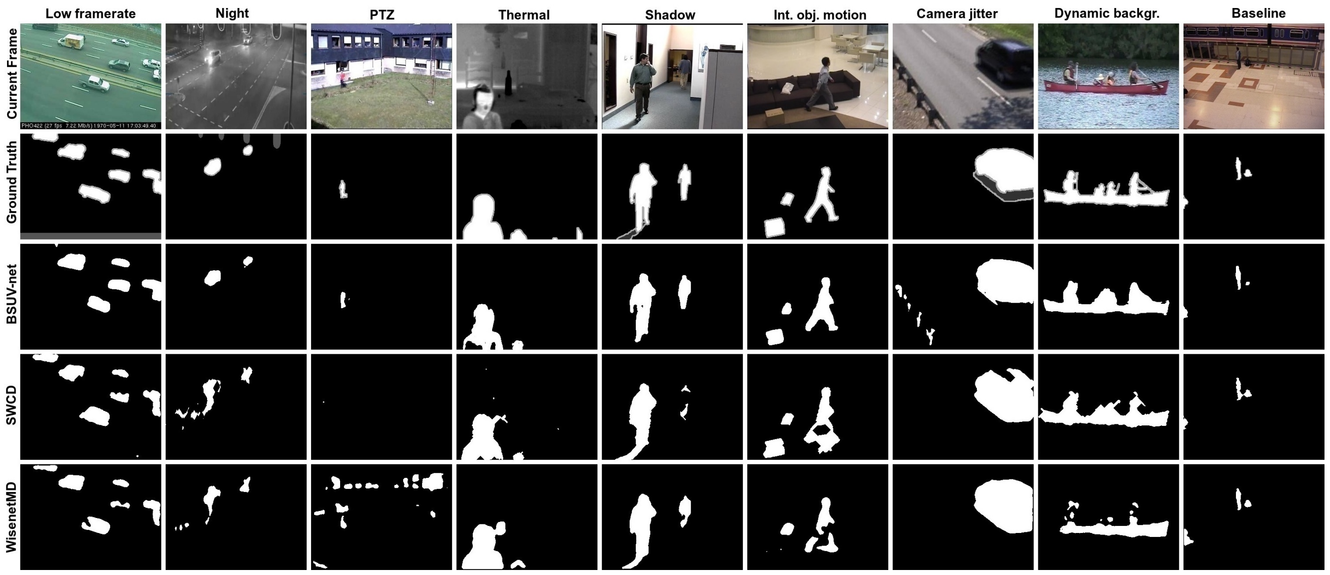

Figure 2. Visual comparison of sample results from BSUV-Net, SWCD and WisenetMD on **unseen videos** from CDNet-2014.

FPM frames, we pre-defined global background and foreground class categories which might be wrong for some of the videos. We did not optimize the selection of these class categories but instead used those suggested in [7]. To demonstrate that our algorithm is not replicating FPM but leverages its semantic information to boost performance, we compared BSUV-Net with thresholded FPM used as a BGS result (Table 5). It is clear that FPM alone is not a powerful tool for BGS as it is significantly outperformed by BSUV-Net.

While in BSUV-Net we assume that the *empty background frame* is foreground-free, CDNet-2014 does not provide empty background frames. Therefore, in some videos, we manually selected empty background frames from among the initial frames as explained in Section 4.2. In Table 6, we show the impact of this manual process by comparing the manual selection strategy with an automatic one, that is using the median of all frames in the test video as a the *empty background frame*. Clearly, the manual selection slightly improves precision while significantly decreasing recall. We believe this is due to the increase of false negatives caused by the appearance of some of the foreground objects in the empty background. Since videos in CDNet2014 are rather short (at most 10 minutes), in some cases the median of all frames does not result in an empty background. However, for stationary surveillance cameras in a real-life scenario it is often possible to compute an empty background, for example by taking the median of frames at the same time of the day (when it is expected to be empty) over many days.

## 5. Conclusions and Future Work

We introduced a novel deep-learning algorithm for background subtraction of unseen videos and proposed a *video-agnostic* evaluation methodology that treats each video in a dataset as unseen. The input to BSUV-Net consists of the current frame and two reference frames from different time-scales, along with semantic information for all three frames (computed using Deeplabv3 [9]). To increase the generalization capacity of BSUV-Net, we formulated a simple, yet effective, illumination-change model. Experimental results on CDNet-2014 show that BSUV-Net outperforms state-of-the-art *video-agnostic* BGS algorithms in terms of 7 out of 9 performance metrics. Its performance can be further improved by adding SemanticBGS [7] as a post-processing layer. This shows great potential for deep-learning BGS algorithms designed for unseen or unlabeled videos.

In the future, we are planning further work on temporal data-augmentation techniques to improve performance for challenging categories, such as "dynamic background" and "camera jitter". We will also investigate different background models for the reference frames. In this work, we kept our focus on designing a high-performance, supervised BGS algorithm for unseen videos without considering the computational complexity. To bring BSUV-Net closer to real-time performance, we are also planning to study a shallow-network implementation designed for fast inference.

Table 5. Comparison of BSUV-Net with thresholded FPM used as a BGS result (probability threshold equals 0.5).

| Method | $Pr$ | $Re$ | $F_1$ |
|---|---|---|---|
| FPM | 0.6549 | 0.6654 | 0.5846 |
| BSUV-net | 0.8113 | 0.8203 | 0.7868 |

Table 6. Comparison of manual and automatic selection of empty background frames in BSUV-Net.

| Empty background selection | $Pr$ | $Re$ | $F_1$ |
|---|---|---|---|
| Automatic | 0.8207 | 0.7812 | 0.7639 |
| Manual | 0.8113 | 0.8203 | 0.7868 |

## References

[1] M. Babaee, D. T. Dinh, and G. Rigoll. A deep convolutional neural network for video sequence background subtraction. *Pattern Recognition*, 76:635–649, 2018.

[2] M. Bakkay, H. Rashwan, H. Salmane, L. Khoudour, D. Puigtt, and Y. Ruichek. BSCGAN: Deep background subtraction with conditional generative adversarial networks. In *Proc. IEEE Int. Conf. Image Processing*, pages 4018–4022. IEEE, 2018.

[3] O. Barnich and M. Van Droogenbroeck. ViBe: A universal background subtraction algorithm for video sequences. *IEEE Trans. Image Process.*, 20(6):1709–1724, 2011.

[4] S. Bianco, G. Ciocca, and R. Schettini. How far can you get by combining change detection algorithms? In *Intern. Conf. on Image Analysis and Process.*, pages 96–107. Springer, 2017.

[5] G.-A. Bilodeau, J.-P. Jodoin, and N. Saunier. Change detection in feature space using local binary similarity patterns. In *Intern. Conf. on Computer and Robot Vision*, pages 106–112. IEEE, 2013.

[6] T. Bouwmans, S. Javed, M. Sultana, and S. K. Jung. Deep neural network concepts for background subtraction: A systematic review and comparative evaluation. *Neural Networks*, 2019.

[7] M. Braham, S. Piérard, and M. Van Droogenbroeck. Semantic background subtraction. In *Proc. IEEE Int. Conf. Image Processing*, pages 4552–4556. IEEE, 2017.

[8] M. Braham and M. Van Droogenbroeck. Deep background subtraction with scene-specific convolutional neural networks. In *Intern. Conf. on Systems, Signals and Image Processing*, pages 1–4. IEEE, 2016.

[9] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[10] A. Elgammal, R. Duraiswami, D. Harwood, and L. S. Davis. Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. *Proc. IEEE*, 90(7):1151–1163, 2002.

[11] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar. A novel video dataset for change detection benchmarking. *IEEE Trans. Image Process.*, 23(11):4663–4679, 2014.

[12] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[13] Ş. Işık, K. Özkan, S. Günal, and Ö. N. Gerek. SWCD: A sliding window and self-regulated learning-based background updating method for change detection in videos. *Journal of Electronic Imaging*, 27(2), 2018.

[14] S.-H. Lee, S.-C. Kwon, J.-W. Shim, J.-E. Lim, and J. Yoo. Wisenetmd: Motion detection using dynamic background region analysis. *arXiv preprint arXiv:1805.09277*, 2018.

[15] L. A. Lim and H. Y. Keles. Foreground segmentation using convolutional neural networks for multiscale feature encoding. *Pattern Recognition Letters*, 112:256–262, 2018.

[16] L. A. Lim and H. Y. Keles. Learning multi-scale features for foreground segmentation. *arXiv preprint arXiv:1808.01477*, 2018.

[17] M. Mandal, V. Dhar, A. Mishra, and S. K. Vipparthi. 3dfr: A swift 3d feature reductionist framework for scene independent change detection. *IEEE Signal Process. Lett.*, 26(12):1882–1886, 2019.

[18] A. Mittal and N. Paragios. Motion-based background subtraction using adaptive kernel density estimation. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, volume 2. IEEE, 2004.

[19] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Intern. Conf. on Medical Image Computing and Computer-assisted Intervention*, pages 234–241. Springer, 2015.

[20] D. Sakkos, H. Liu, J. Han, and L. Shao. End-to-end video background subtraction with 3d convolutional neural networks. *Multimedia Tools and Applications*, pages 1–19, 2017.

[21] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[22] P.-L. St-Charles, G.-A. Bilodeau, and R. Bergevin. A self-adjusting approach to change detection based on background word consensus. In *Proc. IEEE Winter Conf. on Appl. of Computer Vision*, pages 990–997. IEEE, 2015.

[23] P.-L. St-Charles, G.-A. Bilodeau, and R. Bergevin. Subsense: A universal change detection method with local adaptive sensitivity. *IEEE Trans. Image Process.*, 24(1):359–373, 2015.

[24] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, volume 2, pages 246–252. IEEE, 1999.

[25] M. Sultana, A. Mahmood, S. Javed, and S. K. Jung. Unsupervised deep context prediction for background estimation and foreground segmentation. *Machine Vision and Appl.*, 30(3):375–395, 2019.

[26] M. O. Tezcan, J. Konrad, and P. Ishwar. A fully-convolutional neural network for background subtraction of unseen videos. *arXiv preprint arXiv:1907.11371*, 2019.

[27] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler. Efficient object localization using convolutional networks. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 648–656, 2015.

[28] Y. Wang, Z. Luo, and P.-M. Jodoin. Interactive deep learning method for segmenting moving objects. *Pattern Recognition Letters*, 96:66–75, 2017.

[29] D. Zeng, X. Chen, M. Zhu, M. Goesele, and A. Kuijper. Background subtraction with real-time semantic segmentation. *IEEE Access*, 2019.

[30] D. Zeng and M. Zhu. Background subtraction using multiscale fully convolutional network. *IEEE Access*, 6:16010–16021, 2018.

[31] D. Zeng, M. Zhu, and A. Kuijper. Combining background subtraction algorithms with convolutional neural network. *Journal of Electronic Imaging*, 28(1), 2019.

[32] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 2881–2890, 2017.

[33] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Scene parsing through ADE20K dataset. In *Proc. IEEE*

*Conf. Computer Vision Pattern Recognition*, pages 633–641, 2017.

[34] Z. Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *Proc. Int. Conf. Pattern Recognition*, volume 4, pages 28–31. IEEE, 2004.