Full length article

Single image shadow detection via uncertainty analysis and GCN-based refinement strategy

Wen Wu*, Kai Zhou, Xiao-Diao Chen*

Computer and Software School, Hangzhou Dianzi University, Hangzhou, 310018, China

Abstract

Learning-based shadow detection methods have achieved an impressive performance, while these works still struggle on complex scenes, especially ambiguous soft shadows. To tackle this issue, this work proposes an efficient shadow detection network (ESDNet) and then applies uncertainty analysis and graph convolutional networks for detection refinement. Specifically, we first aggregate global information from high-level features and harvest shadow details in low-level features for obtaining an initial prediction. Secondly, we analyze the uncertainty of our ESDNet for an input shadow image and then take its intensity, expectation, and entropy into account to formulate a semi-supervised graph learning problem. Finally, we solve this problem by training a graph convolution network to obtain the refined detection result for every training image. To evaluate our method, we conduct extensive experiments on several benchmark datasets, i.e., SBU, UCF, ISTD, and even on soft shadow scenes. Experimental results demonstrate that our strategy can improve shadow detection performance by suppressing the uncertainties of false positive and false negative regions, achieving state-of-the-art results.

1. Introduction

Shadows generally exist in images. On the one hand, shadows can impede many visual tasks, such as visual tracking [1], object detection [2], and semantic segmentation [3], where shadows may be incorrectly marked as objects. On the other hand, shadows can provide us with useful clues, e.g., light source position [4,5], scene geometry [6–8], and exposure parameters [9]. Hence, shadow detection is an essential part of scene understanding.

Early shadow detection methods are mostly based on color chromaticity, illumination invariant assumptions, and hand-craft features, such as illumination [10–12], color [13,14], and others cues [15,16]. With deep learning [17–19] successfully applied to various vision tasks, recent data-driven models [20–33] use convolutional neural networks (CNNs) or generative adversarial networks (GANs) [34] to detect shadows. Compared with traditional methods, they have achieved substantial performance improvements. These mainstream methods mainly have two strategies: expanding training data [21,26,29] or combining global semantic information [23,24,30,31].

According to [35,36], a shadow image consists of the sunlit, umbra, and penumbra regions. A weaker light source or a longer distance between ground and occluders will cause wider penumbra and even less clear shadow boundaries (soft shadow). For hard shadows, we can employ simple filtering operations to obtain the corresponding shadow masks. While annotating for soft shadows is time-consuming and expensive since the edges of soft shadows are difficult to identify even by eyes. The primary source of inaccuracies in the soft shadow detection results are those ambiguous penumbra regions. In this work, these ambiguities are referred to as uncertain regions (uncertainties). Note that the uncertain regions in the soft shadow images are much larger than the hard shadow images. In particular, these uncertainties are mainly clustering at the shadow boundary. The state-of-the-art methods [30–33] often fail to detect these ambiguous regions due to most of benchmarks (i.e., UCF [16], SBU [21] and ISTD [37]) are mainly consist of simple hard shadow images, leading to soft shadow detection is still very challenging.

From the observations that humans can locate shadows well from various complex scenes. We argue that people tend to first identify the main part of shadows based on the prior of shadows are darker than other non-shadow regions and then compare the ambiguities at the boundary with near (local) and far (global) regions to make the final decision. On account of uncertainty regions can bring clues about potential errors in shadow detection, we need to take this knowledge into account for more accurate results. For this purpose, this work

* This paper has been recommended for acceptance by Zicheng Liu.
* Corresponding authors.

E-mail addresses: wuwen.hdu.cs@gmail.com (W. Wu), xiaodiao@hdu.edu.cn (X.-D. Chen).

https://doi.org/10.1016/j.jvcir.2021.103397
Received 4 May 2021; Received in revised form 20 September 2021; Accepted 25 November 2021
Available online 3 January 2022
1047-3263/© 2021 Elsevier Inc. All rights reserved.
will employ uncertainty analysis and a graph convolutional network (GCN) [38] based refinement strategy to suppress these uncertainties.

Specifically, we first introduce a novel shadow detection framework based on MobileNet-V3 [39], namely ESDNet. In this model, we adopt a direction-aware spatial context (DSC) module [30] to aggregate global information from high-level features and formulate a feature refinement module (FRM) to harvest shadow detail in low-level features. Moreover, inspired by [40,41], Kendall and Gal demonstrate that a stochastic Gaussian process can be approximated using regular CNN dropout layers, a technique known as Monte Carlo dropout (MCDO). It allows us to estimate the uncertainty of one CNN-based segmentation model. CNN’s uncertainty has been demonstrated to be effective as an attention mechanism in semi-supervised learning [42].

As shown in Fig. 1, we first employ our ESDNet to obtain an initial prediction without MCDO and then append the MCDO to generate a series of predictions and compute corresponding expectation and entropy map. Next, we threshold this entropy map into high confident regions (background and the main body of shadows) and uncertain regions (shadow boundary regions). Furthermore, we utilize this information to define a semi-labeled graph to train a GCN in a semi-supervised learning manner. Finally, we re-evaluate a fully connected graph on the well-trained GCN to obtain the final detection result.

The main contributions of this work are following three aspects:

- We propose an efficient shadow detection network (ESDNet). In detail, we combine the MobileNet-V3, consider both low-level and high-level features for obtaining more accurate initial predictions.
- We apply the uncertainty analysis and semi-supervised GCN to introduce a GCN-based refinement strategy. Specifically, we construct graphs in the ROI to reduce the data dimension. Then, we consider local and global connections for label propagation. Finally, we encode abundant features into each node and edge to make it shadow-aware.
- The two parts, i.e., ESDNet and GCN-based refinement strategy, work in a complementary fashion. By combining them, we can obtain a remarkable improvement. Experimental results show that our proposed GCN-based refinement strategy as a post-processing step can also improve existing shadow detector obviously.

2. Related works

2.1. Single image shadow detection

Single image shadow detection has been extensively studied in the computer vision community. Early works mainly focus on illumination models or machine learning algorithms by exploring various hand-crafted shadow features, e.g., illumination [10–12], color [13,14] and texture [15,16]. However, these features have limited ability to distinguish between shadow and non-shadow regions, leading to these approaches do not generalize well to most real-world scenes.

Recently, deep learning-based methods demonstrate remarkable performance improvement on various benchmarks. For example, Khan et al. [20] employ CNNs to extract features at the super-pixel level and object boundaries and then apply a CRF to predict shadow contours. Shen et al. [43] use a structured CNN to predict the structures of shadow edges. Vicente et al. [21] present a semantic-aware stacked-CNN to detect shadows by recovering the noisy shadow annotations. Nguyen et al. [22] introduce a sensitive parameter to the loss function in a conditional generative adversarial network (ScGAN) to solve the unbalanced labels of shadow and non-shadow regions. Wang et al. [29] present a stacked cGAN (ST-cGAN) for jointly learning shadow detection and shadow removal. Le et al. [28] adopt a shadow attenuation network to generate adversarial training samples, further for training a shadow detection network (A+DNet). Hu et al. [30] explore the spatial context in a directional-aware (DSC) way and introduce an RNN-based module to learn the spatial context in four directions. Zhu et al. [24] propose a bidirectional recurrent (BDRAR) model to combine global and local contexts for shadow detection. Wang et al. [37] aggregate multi-scale context with the dense connection. Zheng et al. [31] present a distraction-aware shadow detection network (DSDNet) by explicitly revising the false negative and false positive regions found by other shadow detection methods. Very Recently, Chen et al. [44] propose a multi-task CNN and a multi-task mean teacher framework (MTMT-Net) to leverage unlabeled data for training. Although these methods have achieved high accuracy on current benchmarks, their performances are still limited for complex environments.
2.2. Graph convolutional network

As an extension of CNNs, GCNs [38,45–48] can handle data represented by an irregular graph. Recently, GCNs have effectively demonstrated in graph representation and learning. In the beginning, Kipf et al. [38] propose a simple GCN for graph semi-supervised learning. Hamilton et al. [49] present a general inductive representation and learning framework to represent unseen nodes. Velickovi et al. [45] introduce Graph Attention Network (GANs) for semi-supervised learning. We can observe that one of the main functions of GCN is to represent and label graph nodes by passing messages through the graph structure. Inspired by these works, we regard the pixels with high uncertainty in the initial prediction as nodes, build a graph, and then train a GCN to refine this shadow detection result.

3. Efficient shadow detection network

In this section, we introduce an efficient shadow detection network (ESDNet) for fast shadow detection. Given an RGB image \( I(x) \), where \( x \in \mathbb{R}^3 \) represent the pixel position. Considering the efficiency, we design an light-weight network \( g(I(x); \theta) \) for fast shadow detection and produce a initial shadow map \( \hat{y}(x) = g(I(x); \theta) \), where \( \theta \) is the model’s parameters. Fig. 2 illustrates the detailed structure of our ESDNet. Its input is a shadow image and outputs a shadow mask in an end-to-end manner. Since it has only 4.39 M parameters, it can achieve real-time performance.

3.1. Network architecture

In detail, we first apply the MobileNet-V3 [39] as the backbone of our ESDNet, which combine depthwise separable convolutions (from MobileNet-V1) [50], inverted residual with linear bottleneck (from MobileNet-V2) [51] and squeeze and excitation (from MnasNet) [52]. This backbone follows by a direction-aware spatial context (DSC) module [30], which aims to harvest the DSC features and store global context cues for recognizing shadows. Inspired by ResNet [53], they connect the input and output of a convolution block (i.e., shortcut connection) to avoid gradient vanishing and gradient exploding, which is referred to as “Residual Learning”. In this work, we also connect the input and output of the DSC module for robust training.

Considering that low-level feature maps contain rich fine details which is beneficial for us to locate shadow boundaries and tiny shadows. So we further introduce a feature refinement module (FRM) to enhance shadow details when the distance between the DSC feature and low-level feature is large. Finally, we concatenate the refined low-level feature and high-level feature, and then use a series of convolutional layers to output a shadow mask.

3.2. Feature refinement module

As shown in Fig. 3, FRM’s input is low-level feature \( F_L \) and DSC feature \( F_D \). We first compress the \( F_D \)’s channels through a series of \( 1 \times 1 \) convolution kernel and then upsample it to the same resolution as \( F_L \). Then, we calculate the gated map \( G \) based on the distance between the DSC feature and the low-level feature to measure the importance of detail structures:

\[
G = a \log(1 + (F_L - F_D)^2),
\]

where \( (F_L - F_D)^2 \) represents the distance between two features, we use a logarithmic function to rescale it. Inspired by [54], we then introduce a learnable parameter to adjust the scale of the gate map. Finally, we multiply the gate map \( G \) with input low-level feature \( F_L \) to enhance the spatial detail and generate refined low-level feature \( F_R \).

4. Shadow-aware GCN refinement

In this section, we aim to refine the prediction \( \hat{y} \) by our shadow-aware GCN. Essentially, this step operates as a post-processing step (one image at a time) and assumes no ground truths are available. Thus, in principle, we can use any shadow detection model as the pre-processing step for generating an initial shadow detection result.

4.1. Uncertainty analysis

At this point, we aim to find incorrect predictions at pixel-level from our ESDNet. Inspired by [40,41], Kendall and Gal show that adding a dropout layer to the neural network model is equivalent to approximating the probabilistic deep Gaussian process. Like these works, we also utilize the MCDO approximation to evaluate the uncertainty of ESDNet. In principle, any model with a dropout layer can apply this method without changing the model architecture and re-training. This property makes it an ideal choice for post-processing refinement algorithms.

The use of MCDO is very simple. In the training stage, the MCDO makes it an ideal choice for post-processing refinement algorithms. In the testing stage, we still maintain the dropout layers instead of turning them off, which is the only difference from the conventional method. “MC” in MCDO denotes that we need to forward the same input multiple times to get
the output from “different network structures”. Note that this process can be parallelized. So we perform $T$ (i.e., the times of forwarding prediction) stochastic passes in our ESDNet to get the expectation of the model prediction result:

$$E(x) \approx \frac{1}{T} \sum_{t=1}^{T} g(I(x), \theta_t),$$  \hspace{1cm} (2)

where $\theta_t$ is the parameter in pass $t$. The model uncertainty $U$ is given by the entropy, which can be computed as:

$$U(x) = -\sum_{c} P(x) \log P(x),$$ \hspace{1cm} (3)

where $P(x)$ is the true probability of pixel $x$ belongs to the class $c$, and $N$ is the total number of classes ($N = 2$ in our task). In this work, we use the expectation $E(x)$ to approximate this probability. Finally, we define the potential incorrect pixels by employing a binary threshold on the entropy map:

$$U_b(x) = U(x) > r,$$ \hspace{1cm} (4)

where the uncertainty threshold $r$ determines whether a pixel $x \in \hat{y}$ is an uncertainty element. These uncertain elements will be refined in the rest of the sections.

4.2. GCN-based refinement strategy

At this point, we have obtained a binary mask $U_b$, which indicates pixels with high uncertainty. Notably, uncertainty analysis can only tell us that the model is not confident about its prediction. That means some pixels in $U_b$ may be correct, so its label should not be changed. Fortunately, we can use high confident pixels to train a model for reclassifying our ESDNet’s output.

Using the information from uncertainty analysis, we can define a partially-labeled graph, where pixels correspond to nodes in the graph. By doing this, we can define this refinement task as a semi-supervised graph learning problem. In the rest of sections, we will describe the node labeling, graph connectivity, edge weighting, and loss function in detail.

4.2.1. Node labeling

Given a graph $G$, in the stage of inference, we aim to obtain the output of our shadow-aware GCN model $\hat{y}$ as the refined result $y'$:

$$y' = \Gamma(G); \phi),$$ \hspace{1cm} (5)

where $G$ is constructed from certain and uncertain pixels based on $I$, $\hat{y}$, $E$ and $U$. $\phi$ are parameters of our shadow-aware GCN.

Since most of pixels in $\hat{y}$ (certain regions) are irrelevant. For reducing data dimensionality and memory requirements, we define a working region ROI = dilation($U_b(x)$) u $E_b(x)$, where $E_b$ is the expectation binarized by a threshold of 0.5. Following the priority of entropy value at shadow boundary is higher, we introduce the dilation($U_b(x)$) to ensure ROI is bigger enough to cover shadows. Note that the ROI allows us to include high confident background ($\hat{y}(x) = 0$) and high confident foreground ($\hat{y}(x) = 1$) for training the GCN model. The pixel $x \in ROI$ correspond to the nodes of $G$. To be specific, we represent each node by a feature vector containing intensity $I(x)$, expectation $E(x)$, and entropy $U(x)$. Finally, we label each node follow the rule:

$$l(x) = \begin{cases} \hat{y}(x) & \text{if} \; U_b(x) = 0 \\ \text{unlabeled} & \text{if} \; U_b(x) = 1. \end{cases}$$ \hspace{1cm} (6)

4.2.2. Graph connectivity

We do not consider the most straightforward connection strategy ($n4$ or $n8$) for the following two main reasons: (1) these schemes can only propagate information in a local manner leading to the neglect of global context; (2) pixels with high uncertainty tend to shape contiguous clusters. Therefore, a high uncertain pixel is more likely to be surrounded by other high uncertain pixels, reducing the connection with low uncertain points. While $n4$ or $n8$ connection strategy may limit message passing from confident pixel to uncertain one.

A fully connected graph seems to utilize well both local and global information. However, its main disadvantage is the prohibitive memory requirements. To tackle this problem, we choose a compromise solution. For a node (or pixel), we create a connection with its four neighbors ($n4$) and then connect randomly $k$ additional nodes in ROI. This connectivity defines a sparse representation that considers the local and global connections between high uncertainty and low uncertainty elements. We find that $k = 8$ offers a balance in efficiency and memory requirements.

4.2.3. Edge weighting

The weight on edge is intended to measure the similarity between two adjacent nodes, so we consider color, space, illumination, tone, and initial prediction to make it shadow-aware. This allows us to assign the same label to those similar nodes in the stage of label propagation.

**Color relevance.** Two nodes with similar colors are more likely to have the same label. The color relevance $D_c$ between two nodes is defined as:

$$D_c = \exp(-\frac{\|I_i - I_j\|^2}{2\sigma_c^2}),$$ \hspace{1cm} (7)

where $I_i$ is the intensity at node $x_i$ in RGB space and $I_j$ is the intensity at node $x_j$.

**Space relevance.** Although shadows or uncertain nodes can occur anywhere in the image, two closer nodes in Euclidean space will more likely belong to the same category since uncertain nodes tend to shape contiguous clusters (i.e., shadow boundaries). The space relevance $D_s$ between two nodes is defined as:

$$D_s = \exp(-\frac{\|S_i - S_j\|^2}{2\sigma_s^2}),$$ \hspace{1cm} (8)

where $S_i$ and $S_j$ are the spatial coordinates of node $x_i$ and node $x_j$, respectively.

**Illumination relevance.** As the light source is obscured, the shadow region’s illumination is often lower than in the non-shadow
region. The illumination relevance $D_i$ between two nodes is defined as:

$$D_i = \exp(-\frac{(L_i - L_j)^2}{2\sigma_i^2}),$$  \hspace{1cm} (9)$$

where $L_i$ and $L_j$ are the illumination value at node $x_i$ and node $x_j$, which are represented by the L channel in Lab color space.

**Tone relevance.** Generally, shadow and non-shadow regions vary in tone. Thus it can be used as a feature to distinguish shadow from non-shadow areas. The tone relevance $D_h$ between two nodes is defined as:

$$D_h = \exp(-\frac{(H_i - H_j)^2}{2\sigma_h}),$$  \hspace{1cm} (10)$$

where $H_i$ and $H_j$ are the tone value of node $x_i$ and node $x_j$, which are represented by the L channel in Lab color space.

Using these above relevance, we define the weight (similarity) between node $x_i$ and $x_j$ as:

$$w_j(x_i, x_j) = a \cdot \text{div}(x_i, x_j) + b \cdot (D_i + D_j + D_h),$$  \hspace{1cm} (11)$$

where $a$ and $b$ are balance factors, and $\text{div}()$ is the diversity between nodes [55], defined as $\text{div}(x_i, x_j) = \sum_{c=1}^{C} (P^j(x_i) - P^j(x_i)) \log \frac{P^j(x_i)}{P^j(x_i)}$.

Next, we re-train them on the ISTD training set for more comparison. For the fair competition, we first train all competitive methods on the SBU training set and test on the SBU and UCF testing set. We use a stochastic gradient descent algorithm with Adam to optimize the network by minimizing binary cross-entropy (BCE) loss between the ground truths and predicted shadow masks. We set the initial learning rate to 0.0004 with a batch size of 24, and horizontally and vertically flip the training images for data augmentation.

It is worth mentioning that the ESDNet is used for initial shadow detection, and other models can also be instead, mainly because our

$$\ell_e' = -\frac{1}{N} \sum_{x_i \in S} \sum_{c \in \{0,1\}} (y'(x_i)_c \log (y'(x_i))_c + (1 - y'(x_i))_c \log (1 - y'(x_i))_c),$$  \hspace{1cm} (19)$$

where $N$ refers to the number of unlabeled images in current graph. In addition, since similar pixel usually shares the same label (Section 4.2.3), we introduce a Laplacian loss $\ell_p$ to ensure the consistency of the class prediction with the image contents. We incorporate prior knowledge $\omega(x_i, x_j)$ into the training procedure:

$$\ell_p = \frac{1}{2N} \sum_{x_i \in S} \sum_{x_j \in N} \omega(x_i, x_j) \cdot \|y'(x_i) - y'(x_j)\|_2^2,$$  \hspace{1cm} (20)$$

where $N$ is the number of node in the current graph.

### 5. Experiments

In this section, we first present the shadow detection benchmark datasets and evaluation metrics and then compare the proposed ESDNet and ESDNet+ (append shadow-aware GCN) with the state-of-the-art methods and finally report ablation study results. Our code will soon be released at [http://www.deeplab.club](http://www.deeplab.club).

#### 5.1. Datasets and evaluation metrics

**Benchmark datasets.** We evaluate our method on three widely used benchmark datasets: SBU, UCF, and ISTD. (1) SBU is the largest annotated shadow dataset, which contains 4089 training images and 638 testing images; (2) UCF consists of 145 training images and 76 testing images, which is mainly used for conventional machine learning algorithms since this dataset is small; (3) ISTD is a recent dataset for both shadow detection and shadow removal, it has 1870 triples of shadow, shadow map, and shadow-free images, 540 of them are used for testing. For the fair competition, we first train all competitive methods on the SBU training set and test on the SBU and UCF testing set. Next, we re-train them on the ISTD training set for more comparison.

**Evaluation metrics.** We follow [30–33] and use balance error rate (BER) to evaluate the shadow detection performance quantitatively. The BER equally considers shadow and non-shadow regions, which is given by:

$$\text{BER} = 1 - \frac{1}{2} \left( \frac{TP}{TP + FN} + \frac{TN}{TN + FP} \right).$$  \hspace{1cm} (21)$$

We also consider separated per pixel error rates per class (shadow and non-shadow) marked as “S” and “N”. A small BER value indicates a better shadow detection performance.

#### 5.2. Implementation details

**ESDNet.** We pre-train MobileNet-V3 on ImageNet [56] and then perform transfer learning on the shadow datasets. We use a stochastic gradient descent algorithm with Adam to optimize the network by minimizing binary cross-entropy (BCE) loss between the ground truths and predicted shadow masks. We set the initial learning rate to 0.0004 with a batch size of 24, and horizontally and vertically flip the training images for data augmentation.
refinement method uses the model-independent MCDO analysis. Specifically, we use MCDO to calculate the expectation and entropy with a dropout rate of 0.3 and $T = 20$ in stochastic passes.

Shadow-aware GCN. Similar to [38], our GCN model is a two-layered network, and its hidden layer contains 16 feature maps and a single output neuron for binary classification. We train our shadow-aware GCN for 300 epochs with a learning rate of 0.01 and the Adam optimizer.

In our experiment, we set $\alpha = 0.5$, $\beta = 1$ in edge weighting function, the $\sigma_1$, $\sigma_2$, $\sigma_3$ and $\sigma_4$ in Eqs. (7) to Eq. (10) are set to 100, the $\lambda_1$ and $\lambda_2$ in Eq. (16) are set to 10 and 0.1, respectively.

After training, we build a fully connected graph among all pixels in ROI and then feed this graph into our GCN model to obtain the refinement prediction. In the end, we only need to replace the initial prediction results with the GCN’s predictions for those uncertain pixels.

5.3. Evaluation of the network design

At this point, we evaluate the effectiveness of the main components in ESDNet+ on the three benchmarks, i.e., SBU, UCF, and ISTD.

- First, we build a “basic” model using the MobileNet-V3 network to predict shadow masks directly. This model is built by removing the DSC, FRM, skip connections of low-level features from our ESDNet.
- Then, we add the DSC module to aggregate global features, namely “basic+DSC”.
- Further, we consider the low-level features refined by the FRM and set up another network model, namely “ESDNet w/o FRM”, by removing the FRM from the whole architecture and directly concatenating the low-level and high-level features.
- Next, we add back all components in Fig. 2, namely “ESDNet”.
- To evaluate the effectiveness of the shadow-aware GCN refinement, we further build “ESDNet+” by adding the refined strategy shown in Fig. 4.

From the quantitative evaluation results shown in Table 1 and Fig. 5, we can observe that the major components help improve the results and contribute to the entire pipeline.

Table 1

<table>
<thead>
<tr>
<th>Method</th>
<th>SBU BER</th>
<th>UCF BER</th>
<th>ISTD BER</th>
</tr>
</thead>
<tbody>
<tr>
<td>basic</td>
<td>11.44</td>
<td>10.15</td>
<td>12.77</td>
</tr>
<tr>
<td>basic+DSC</td>
<td>12.24</td>
<td>11.37</td>
<td>15.16</td>
</tr>
<tr>
<td>ESDNet w/o FRM</td>
<td>11.15</td>
<td>18.19</td>
<td>6.09</td>
</tr>
<tr>
<td>ESDNet</td>
<td>2.55</td>
<td>11.12</td>
<td>18.48</td>
</tr>
<tr>
<td>ESDNet+</td>
<td>2.59</td>
<td>7.38</td>
<td>12.33</td>
</tr>
<tr>
<td>ESDNet+ w/o FRM</td>
<td>7.23</td>
<td>9.98</td>
<td>4.54</td>
</tr>
</tbody>
</table>

Fig. 5. Shadow detection results for ablation study.

5.4. Comparison with state-of-the-art methods

We make comparisons with seven recent deep learning-based shadow detectors, including ScGAN [22], ST-CGAN [29], DSC [30], A+D Net [28], BDRAR [24], DSDNet [31], and MTMT-Net [44]. To make a fair comparison, we use the same training data with the same input size of images (512 x 512) to train all the learning-based methods on the same hardware.

Hard shadow detection: As shown in Table 2, we summarize the quantitative results of different shadow detectors on the three benchmark datasets. The BER score is the average of shadow and non-shadow BER scores. Where “FPS” stands for “frames per second”, which is evaluated on a GeForce RTX 3080 GPU, and “P” stands for “the number of parameters”.

Among the deep learning-based shadow methods, MTMT-Net [44] is the second-best performing method, which presents a multi-task mean teacher model for semi-supervised shadow detection by leveraging unlabeled data and exploring the learning of multiple information of shadows simultaneously. Compared to the MTMT-Net, our ESDNet+ have 4.12%, 3.21%, and 2.32% lower BER values on SBU, UCF, and ISTD, respectively. Besides, our method has fewer parameters and higher efficiency. Unlike other four competitor (i.e., DSC, BDRAR, DSDNet and MTMT-Net), they use CRF [57] as post-processing. Our ESDNet+ only uses GCN refinement as our post-processing strategy. The results indicate using the GCN refinement strategy can obtain an obvious improvement based on ESDNet.

To further explain the performance of all methods, we also provide some visualization results in Figs. 6–8. We can see that our ESDNet+ has the best performance among all the shadow detectors. It can effectively locate different shadows under various backgrounds and successfully discriminates natural shadows from those non-shadow regions with shadow appearance. For example, ESDNet+ can accurately detect the shadow regions, while the others mistakenly recognize the road, the sky, and the dark ground as shadows. What is more, for high-contrast objects in a large shadow region, ESDNet+ can still recognize them correctly.

Soft shadow detection: The rare soft shadow scenes in current benchmarks leading to soft shadow detection is more challenging than conventional hard shadow detection. Moreover, the performance of soft shadow detection is also an important indicator of the model’s generalizability. We select various soft shadow images with different complex scenes from the internet and then perform extensive experiments as shown in Fig. 9.

We can observe that: (1) most shadow detection methods can only detect the main body of soft shadows; (2) although a few methods have better results than ScGAN, they cannot detect fine-grained shadow edges; (3) our shadow detection refinement method can overcome shadow detection uncertainty in the soft shadow boundary and obtain a complete shadow boundary.
Table 2
Comparing with state-of-the-art methods in terms of BER.

<table>
<thead>
<tr>
<th>Method</th>
<th>FPS</th>
<th>P(M)</th>
<th>SBU</th>
<th>BER</th>
<th>S</th>
<th>N</th>
<th>UCF</th>
<th>BER</th>
<th>S</th>
<th>N</th>
<th>ISTD</th>
<th>BER</th>
<th>S</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>ScGAN</td>
<td>64.06</td>
<td>28.24</td>
<td>9.04</td>
<td>8.39</td>
<td>9.69</td>
<td>11.52</td>
<td>7.74</td>
<td>15.3</td>
<td>4.7</td>
<td>3.22</td>
<td>6.18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST-CGAN</td>
<td>29.83</td>
<td>62.58</td>
<td>8.14</td>
<td>3.75</td>
<td>12.53</td>
<td>4.94</td>
<td>11.23</td>
<td>3.85</td>
<td>2.14</td>
<td>5.55</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DSC</td>
<td>8.91</td>
<td>79.03</td>
<td>5.59</td>
<td>9.76</td>
<td>1.42</td>
<td>10.54</td>
<td>18.08</td>
<td>3.01</td>
<td>3.42</td>
<td>3.85</td>
<td>3.01</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A+DNet</td>
<td>122.44</td>
<td>54.41</td>
<td>5.37</td>
<td>4.45</td>
<td>6.32</td>
<td>9.25</td>
<td>8.37</td>
<td>10.14</td>
<td>3.23</td>
<td>3.76</td>
<td>2.82</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BDRAR</td>
<td>32.76</td>
<td>42.46</td>
<td>3.64</td>
<td>3.40</td>
<td>3.89</td>
<td>7.81</td>
<td>9.69</td>
<td>5.94</td>
<td>2.69</td>
<td>0.52</td>
<td>4.87</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DSDNet</td>
<td>39.53</td>
<td>58.16</td>
<td>3.45</td>
<td>3.33</td>
<td>3.58</td>
<td>7.59</td>
<td>9.74</td>
<td>5.44</td>
<td>2.17</td>
<td>1.36</td>
<td>2.98</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTMTNet</td>
<td>21.88</td>
<td>169.62</td>
<td>3.15</td>
<td>3.73</td>
<td>2.57</td>
<td>7.47</td>
<td>10.31</td>
<td>4.63</td>
<td>1.72</td>
<td>1.36</td>
<td>2.08</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ESDNet(ours)</td>
<td>139.54</td>
<td>4.39</td>
<td>3.31</td>
<td>3.78</td>
<td>2.59</td>
<td>7.38</td>
<td>12.33</td>
<td>3.94</td>
<td>2.08</td>
<td>1.21</td>
<td>2.41</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ESDNet+(ours)</td>
<td>117.38</td>
<td>4.58</td>
<td>3.02</td>
<td>3.64</td>
<td>2.43</td>
<td>7.23</td>
<td>9.98</td>
<td>4.54</td>
<td>1.68</td>
<td>1.19</td>
<td>2.32</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5.5. Ablation study

In this section, we aim to explore how the choice of the hyper-parameters can affect the performance. For this purpose, we evaluated the effects of different threshold value $\tau$, dropout ratio $\theta$, stochastic times $T$ and the number of random nodes $k$.

We can observe from Table 3, there is no significant impact on parameter selection. We assume that there is a clear separation between
high and low uncertainty elements. Therefore, changing $\tau$ may only add (remove) a small number of nodes that are not important for GCN training. Our result suggests that intermediate values are preferred since they lead to a lower number of nodes and edges, in sequence, to a lower memory requirement.

From Table 4, we find that a too high or too low dropout ratio and too many or too few stochastic times may inhibit the appearance of uncertain pixels, leading to poor shadow detection results.

From Table 5, we can see that although too many random nodes will slightly increase the shadow detection performance. However, at the same time, it will greatly reduce the efficiency of shadow detection and increase memory requirements. On the other hand, too few random nodes will reduce the effect of label propagation.

5.6. Applications

In this section, we apply the shadow-aware GCN refinement strategy on several state-of-the-art detectors. As shown in Table 6, compared to Table 2, different methods have a certain degree of improvement.

Note that our refinement strategy can only deal with pixels with high uncertainty, and we do not further perform pixels belong to shadow or non-shadow regions with high confidence, even when these predictions are wrong. In this way, those models with poor generalizability cannot obtain a good enough result in the stage of GCN refinement.

6. Conclusion

In this work, we design ESDNet, an efficient deep neural network architecture, and formulate the feature refinement module to bring in more shadow details from the low-level features. Moreover, we show the construction of a sparse semi-labeled graph based on the output of our ESDNet and uncertainty analysis. We have shown that graph semi-supervised learning can be used to obtain refined detection results. Comparing with the state-of-the-art models, our method is more accurate and efficient. In the future, we will apply GCN to video shadow detection and explore more efficient methods for node connectivity to overcome the high memory requirement.
References


