Large-Scale Training of Shadow Detectors with Noisily-Annotated Shadow Examples

Tomás F. Yago Vicente(✉), Le Hou, Chen-Ping Yu,
Minh Hoai, and Dimitris Samaras

Computer Science Department, Stony Brook University, Stony Brook, USA
{tyagovicente,lehhou,cheyu,minhhoai,samaras}@cs.stonybrook.edu

Abstract. This paper introduces training of shadow detectors under the large-scale dataset paradigm. This was previously impossible due to the high cost of precise shadow annotation. Instead, we advocate the use of quickly but imperfectly labeled images. Our novel label recovery method automatically corrects a portion of the erroneous annotations such that the trained classifiers perform at state-of-the-art level. We apply our method to improve the accuracy of the labels of a new dataset that is 20 times larger than existing datasets and contains a large variety of scenes and image types. Naturally, such a large dataset is appropriate for training deep learning methods. Thus, we propose a semantic-aware patch level Convolutional Neural Network architecture that efficiently trains on patch level shadow examples while incorporating image level semantic information. This means that the detected shadow patches are refined based on image semantics. Our proposed pipeline can be a useful baseline for future advances in shadow detection.

Keywords: Shadow detection · Large scale shadow dataset · Noisy labels

1 Introduction

Shadows are ubiquitous in images of natural scenes. On one hand, shadows provide useful cues about the scene including object shapes [28], light sources and illumination conditions [23,30,31], camera parameters and geo-location [19], and scene geometry [21]. On the other hand, the presence of shadows in images creates difficulties for many computer vision tasks from image segmentation to object detection and tracking. In all cases, being able to automatically detect shadows, and subsequently remove them or reason about their shapes and sizes would usually be beneficial. Moreover, shadow-free images are of great interest for image editing, computational photography, and augmented reality, and the first crucial step is shadow detection.

Shadow detection in single images is a well studied, but still challenging problem. Early work focused on physical modeling of the illumination and shadowing phenomena. Such approaches, e.g., illumination invariant methods [8,9],

B. Leibe et al. (Eds.): ECCV 2016, Part VI, LNCS 9910, pp. 816–832, 2016.
DOI: 10.1007/978-3-319-46466-4_49

only work well for high quality images [24]. In contrast, for consumer-grade photographs and web quality images, the breakthrough in performance came with statistical learning approaches [12,17,24,49]. These approaches learn the appearance of shadows from images with ground-truth labels. The first sizable database with manually annotated shadows was the UCF shadow dataset [49], followed, soon after, by the UIUC shadow dataset [12]. These publicly available datasets with pixel-level annotations have led to important advances in the field. They enabled both systematic quantitative and qualitative evaluation of detection performance, as opposed to the prior practice of qualitative evaluation on a few selected images. In the past few years, several novel shadow detection methods (e.g., [14,43]), gradually advanced state-of-the-art performance in these datasets, to the point of saturation. However, shadow detection is still far from being solved. Due to their limited sizes, UIUC is biased by certain type of images such as objects in close range shots, whereas UCF is biased towards scenes with darker shadows. Thus their generality is limited, and as expected, cross-dataset performance (e.g., training on UIUC and testing on UCF) degrades significantly [13,44]. In order to facilitate the development of robust classifiers, a much larger and more general dataset is needed. However, creating a large shadow dataset would require enormous amount of effort, primarily for obtaining pixel-level annotation.

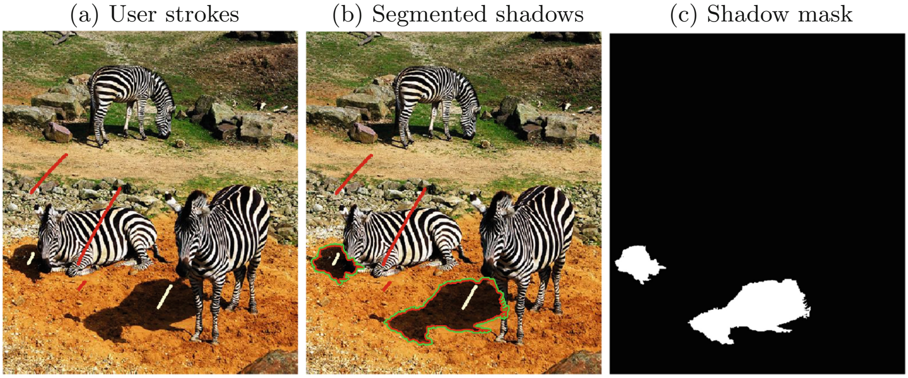

Fig. 1. Lazy labeling for shadow annotation [44]**.** (a) White strokes for shadows, red stokes for negative areas. (b) Automatically segmented shadow regions. (c) Corresponding annotation mask. (Color figure online)

Fortunately, pixel-level annotation might not be required after all, given the recently introduced *lazy labeling* approach [44]. Instead of pixel selection or boundary tracing, lazy labeling allows annotators to use a few brush strokes to roughly label a shadow image, as illustrated in Fig. 1. Lazy labeling significantly reduces annotation time, so now 3–4 images can be easily annotated per minute. The drawback of lazy labeling is that the obtained annotation can be noisy.

However, it is possible to recover the true value for a large portion of such noisy labels so that the noisy annotated shadow images are still useful [44].

In this work, we introduce an efficient framework for learning shadow detection from a large collection of noisy annotations. Our **first contribution** is the extension of our previous work [44] to yield a scalable kernelized method for noisy label recovery. Noisy label recovery is posed as an optimization problem, seeking to minimize the sum of squared leave-one-out errors for a Kernel Least Squares SVM [40]. Since the leave-one-out error is most meaningful for similar data instances, we propose to group similar images into small clusters and perform label recovery for each cluster independently. Hence, our method can be used for large-scale noisy label recovery. Our **second contribution** is a novel stacked Convolutional Neural Network (CNN) based method for structured shadow prediction that takes advantage of the wealth of cleaned-up data. Given a large dataset, we expect to learn not only local shadow cues, but also the discriminative global context. Our *semantics-aware* stacked CNN architecture combines an image level Fully Connected Network (FCN) and a patch-based CNN (patch-CNN). We train the FCN for semantically aware shadow prediction. We use the outputs of the FCN together with the corresponding input RGB images to train the patch-CNN from a random initialization. Thus, the output of the FCN functions as an image-level shadow prior that is further refined by the more local appearance focus of the patch-CNN. To validate our approach while addressing the need for a large-scale shadow dataset, we collected the largest ever shadow dataset. This is the **third contribution** of this paper. Our dataset of almost 5000 images covers a wide range of scenes and is 20 times bigger than UCF [49], bringing shadow detection to the large-data paradigm, and increasing the utility of deep learning approaches.

We first validate our model trained on the newly collected training set performing shadow detection on the UCF test set. Experimental results show comparable performance to state of the art methods [14,43] trained on the UCF training set. This is remarkable as our training set does not overlap with the UCF dataset, proving the generality of our trained model and dataset. We carefully annotated shadow masks for 700 images to serve as a new benchmark for shadow detection. The test set covers a wide range of scenes. Our method achieves a Balanced Error Rate (BER) of 11 % in the new test set, setting the baseline for future comparisons. We observe that our label recovery method correctly retrieves most of the shadows missed by human annotators. Experiments training our network model with cleaned annotations show an improvement in classification performance by 9.1 %, thus proving the effectiveness of our label recovery framework. The dataset is available to the public at http://www3.cs.stonybrook.edu/~cvl/dataset.html.

2 Previous Work

A number of shadow detection methods have been developed in recent years. Guo *et al.* [12] proposed to model long-range interaction between pairs of regions of the same material, with two types of pairwise classifiers: same illumination condition and different illumination condition. Then, they combined the pairwise classifier and a shadow region classifier with a CRF. Similarly, Vicente *et al.* [45] proposed an MRF that combines a unary region classifier with a pairwise classifier and a shadow boundary classifier. These approaches achieved good shadow detection results, but required expensive ground-truth annotation. Khan *et al.* [14] were the first to use deep learning for shadow detection. They combined a CNN for shadow patches and a CNN for shadow boundaries with a CRF, achieving state-of-the-art results at the time. Vicente *et al.* [43] optimized a multi-kernel model for shadow detection based on leave-one-out estimates, obtaining even better shadow predictions than [14]. More recently, Shen *et al.* [35] proposed a CNN for structured shadow edge prediction.

Label noise, also known as class noise, may severely degrade classification performance [10,50]. Numerous methods seek robustness to noisy labels [6,22,27,39]. For instance, Stempfel and Ralaivola [38] deal with training a binary Support Vector Machine (SVM) when the probability of flipping a label is constant and only depends on the true class. For this, they replace the objective functional by a uniform estimate of the corresponding noise-free SVM objective. This becomes a non-convex problem that can be solved with Quasi-Newton BFGS. Biggio *et al.* [3] compensate noise in the labels by modifying the SVM kernel matrix with a structured matrix modeling the noise. This approach only models random flips with fixed probability per class and adversarial flips. That is, for a set number of labels to be flipped, the adversary tries to maximize the classification error. These methods are designed to be unaffected by label noise rather than to be effective in using noisy labels for training. Moreover, these methods focus on asymptotic behavior with unlimited training data. The label recovery method described in this paper is built on our previous work [44], addressing the scalability issues to handle a large amount of training samples.

3 Noisy Label Recovery with Kernel Least Squares SVM

In this section, we describe a method for noisy label recovery. We pose it as an optimization problem, where the labels of some training examples can be flipped to minimize the sum of squared leave-one-out errors. Our formulation is based on the fact that the leave-out-out error of kernel LSSVM is a linear function of the labels. Our method extends our previous work [44] by introducing a kernelized algorithm for noisy label recovery that allows the use of non-linear kernels, which have been shown to be important for shadow detection [43]. Our framework for recovering noisy annotation is based on Least-Squares Support Vector Machine (LSSVM) [33,41]. LSSVM has a closed-form solution, and once the solution of the LSSVM has been computed, the solution for a reduced training set obtained by removing any training data point can be found efficiently. This enables reusing training data for further calibration, e.g., [15,16], and for noisy label recovery.

Given a training set of n data points $\{\mathbf{x}_i\}_{i=1}^{n}$[1] and associated binary labels $\{y_i\}_{i=1}^{n}$, LSSVM optimizes the following:

$$\underset{\mathbf{w},b}{\text{minimize}} \;\; \lambda||\mathbf{w}||^2 + \sum_{i=1}^{n} s_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b - y_i)^2. \tag{1}$$

For high dimensional data (i.e., $\phi(\mathbf{x}_i)$ is large), it is more efficient to obtain the solution for $(\mathbf{w}, b)$ via the representer theorem, which states that $\mathbf{w}$ can be expressed as a linear combination of training data, i.e., $\mathbf{w} = \sum_{i=1}^{n} \alpha_i \phi(\mathbf{x}_i)$. Let $\mathbf{K}$ be the kernel matrix, $k_{ij} = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$. The objective function becomes:

$$\underset{\boldsymbol{\alpha},b}{\text{minimize}} \, \lambda\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} + \sum_{i=1}^{n} s_i(\mathbf{k}_i^T\boldsymbol{\alpha} + b - y_i)^2 \tag{2}$$

Here s_i is the instance weight, allowing the assignment of different weights to different training instances. Let $\overline{\boldsymbol{\alpha}} = [\boldsymbol{\alpha}, b], \overline{\mathbf{K}} = [\mathbf{K}; \mathbf{1}_n^T], \mathbf{R} = \begin{bmatrix} \lambda\mathbf{K} & \mathbf{0}_n \\ \mathbf{0}_n^T & 0 \end{bmatrix}$. Then Eq. (2) is equivalent to minimizing $\lambda\overline{\boldsymbol{\alpha}}^T\mathbf{R}\overline{\boldsymbol{\alpha}} + \sum_{i=1}^{n} s_i(\overline{\mathbf{k}}_i^T\overline{\boldsymbol{\alpha}} - y_i)^2$. This is an unconstrained quadratic program, and the optimal solution can be found by setting the gradient to zero. That is to solve:

$$(\mathbf{R} + \overline{\mathbf{K}}diag(\mathbf{s})\overline{\mathbf{K}}^T)\overline{\boldsymbol{\alpha}} = \overline{\mathbf{K}}diag(\mathbf{s})\mathbf{y} \tag{3}$$

Let $\mathbf{C} = \mathbf{R} + \overline{\mathbf{K}}diag(\mathbf{s})\overline{\mathbf{K}}^T, \mathbf{d} = \overline{\mathbf{K}}diag(\mathbf{s})\mathbf{y}$. The solution for kernel LSSVM is: $\overline{\boldsymbol{\alpha}} = \mathbf{C}^{-1}\mathbf{d}$. Now suppose we remove the training data $\mathbf{x}_i$, let $\mathbf{C}_{(i)}, \mathbf{d}_{(i)}, \overline{\boldsymbol{\alpha}}_{(i)}$ be the corresponding values when removing $\mathbf{x}_i$. We have $\overline{\boldsymbol{\alpha}}_{(i)} = \mathbf{C}_{(i)}^{-1}\mathbf{d}_{(i)}$. Note that, even though we remove $\mathbf{x}_i$ from the training data, we can still write $\mathbf{w}$ as the linear combination of $\phi(\mathbf{x}_1), \cdots, \phi(\mathbf{x}_n)$ without excluding the term $\phi(\mathbf{x}_i)$. The matrices $\mathbf{K}, \overline{\mathbf{K}}, \mathbf{R}$ remain the same, and the only change is the removal of $s_i(\mathbf{k}_i^T\boldsymbol{\alpha} + b - y_i)^2$ from the objective function. Thus we have $\mathbf{C}_{(i)} = \mathbf{C} - s_i\overline{\mathbf{k}}_i\overline{\mathbf{k}}_i^T$ and $\mathbf{d}_{(i)} = \mathbf{d} - y_i s_i \overline{\mathbf{k}}_i$. Using the Sherman-Morrison formula, we have:

$$\mathbf{C}_{(i)}^{-1} = (\mathbf{C} - s_i\overline{\mathbf{k}}_i\overline{\mathbf{k}}_i^T)^{-1} = \mathbf{C}^{-1} + \frac{\mathbf{C}^{-1}s_i\overline{\mathbf{k}}_i\overline{\mathbf{k}}_i^T\mathbf{C}^{-1}}{1 - s_i\overline{\mathbf{k}}_i^T\mathbf{C}^{-1}\overline{\mathbf{k}}_i} \tag{4}$$

Using the above equations to develop $\overline{\boldsymbol{\alpha}}_{(i)} = \mathbf{C}_{(i)}^{-1}\mathbf{d}_{(i)}$, and let $\mathbf{M} = \mathbf{C}^{-1}\overline{\mathbf{K}}$ and $\mathbf{H} = \mathbf{M}^T\overline{\mathbf{K}}$, we obtain the following formula for the LOO weight vector:

$$\overline{\boldsymbol{\alpha}}_{(i)} = \overline{\boldsymbol{\alpha}} + \frac{(\overline{\boldsymbol{\alpha}}^T\overline{\mathbf{k}}_i - y_i)s_i}{1 - s_i h_{ii}}\mathbf{m}_i$$

[1] Bold uppercase letters denote matrices (e.g., $\mathbf{K}$), bold lowercase letters denote column vectors (e.g., $\mathbf{k}$). $\mathbf{k}_i$ represents the i^{th} column of the matrix $\mathbf{K}$. k_{ij} denotes the scalar in the row j^{th} and column i^{th} of the matrix $\mathbf{K}$ and the j^{th} element of the column vector $\mathbf{k}_i$. Non-bold letters represent scalar variables. $\mathbf{1}_n \in \Re^{n\times 1}$ is a column vector of ones, and $\mathbf{0}_n \in \Re^{n\times 1}$ is a column vector of zeros.

The LOO error can therefore be computed efficiently: $\overline{\boldsymbol{\alpha}}_{(i)}^T \overline{\mathbf{k}}_i - y_i = \frac{\overline{\boldsymbol{\alpha}}^T \overline{\mathbf{k}}_i - y_i}{1 - s_i h_{ii}}$.

Substituting $\overline{\boldsymbol{\alpha}} = \mathbf{M} diag(\mathbf{s})\mathbf{y}$ into the above, the leave-one-out error becomes:

$$\frac{\overline{\mathbf{k}}_i^T \mathbf{M} diag(\mathbf{s})\mathbf{y} - y_i}{1 - s_i h_{ii}} \tag{5}$$

Let $\mathbf{P} = diag(\mathbf{s})\mathbf{H}$ and recall that $\mathbf{H} = \mathbf{M}^T \overline{\mathbf{K}}$. The leave-one-out error can be shown to be: $\frac{\mathbf{p}_i^T \mathbf{y} - y_i}{1 - p_{ii}}$. Let $\mathbf{e}_i$ be the i^{th} column of the identity matrix of size n, and let $\mathbf{a}_i = \frac{\mathbf{p}_i - \mathbf{e}_i}{1 - p_{ii}}$, then the leave-one-out error becomes $\mathbf{a}_i^T \mathbf{y}$. Because the vector $\mathbf{a}_i$ only depends on the data, the leave-one-out error is a linear function of the label vector $\mathbf{y}$.

Let $\mathcal{P}, \mathcal{N}$ be the indexes of (noisy) positive and negative training instances respectively, i.e. $\mathcal{P} = \{i | y_i = 1\}$ and $\mathcal{N} = \{i | y_i = 0\}$. We pose noisy label recovery as the optimization problem that minimizes the sum of squared leave-one-out errors:

$$\underset{y_i \in \{0,1\}}{\text{minimize}} \sum_{i=1}^{n} (\mathbf{a}_i^T \mathbf{y})^2, \text{s.t.} \sum_{i \in \mathcal{P}} y_i \geq \alpha |\mathcal{P}| \quad \text{and} \quad \sum_{i \in \mathcal{N}} y_i \leq (1 - \beta)|\mathcal{N}|. \tag{6}$$

In the above $|\mathcal{P}|, |\mathcal{N}|$ are the original number of positive and negative training instances respectively, and α, β are parameters of the formulation $(0 \leq \alpha, \beta \leq 1)$. The constraint of the above optimization problem requires that the proportion of original positive training instances that remains positive must be greater than or equal to α. It also limits the proportion of flipped negative data points to be at most $1 - \beta$. If $\alpha = \beta = 1$, none of the training labels can be flipped.

4 Large-Scale Noisy Label Recovery

The presence of label noise is known to deteriorate the quality of training data. To address this problem, we use the method described in Sect. 3. However, this method requires solving a binary quadratic program in which the number of variables is the same as the number of image regions. This full-scale optimization problem is too big for the optimization algorithm developed in our previous work [44]. To circumvent this issue, we propose here a simple but effective approach. We divide images into clusters of similar images, and perform label recovery for each cluster independently. This approach is motivated by the fact that our label recovery algorithm is based on optimizing the leave-one-out errors. Perhaps the wrong label of a region can be corrected because the region is similar to other regions with correct labels. As such, for label recovery, dissimilar regions do not have much impact on each other. Hence, it makes sense to recover labels within clusters of similar images.

The ability to perform label recovery in smaller clusters leads to large-scale label recovery. Using our approach, we can recover the labels of hundreds of thousands of image regions. This approach allows us to consider superpixels rather than larger regions as in our previous work [44]. We oversegment images using Linear Spectral Clustering [48]. The oversegmentation minimizes frequent inaccuracies in shadow segmentation where small shadow areas "leak" into large non-shadow regions. After all shadows are well known to confound segmentation.

For image clustering, we use a modified version of the Parametric Graph Partitioning method (PGP) [46], which has been shown to work well for image and video segmentation [47]. Here we use PGP instead of the more popular k-means clustering because PGP does not require setting the number of clusters. The details of the image clustering algorithm are provided below (Fig. 2).

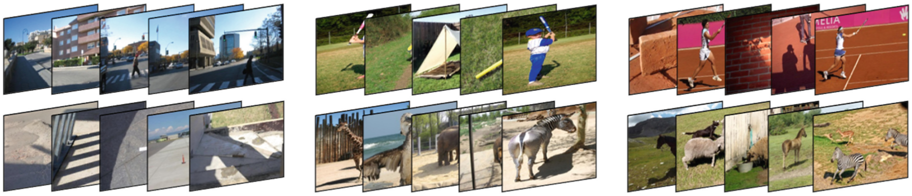

Fig. 2. Examples of clusters of similar shadow images.

Image clustering details. We aim to create clusters of images that depict similar scenes and therefore similar shadows (the appearance of shadows depends on scene properties, including illumination, the color, and the texture of materials). For feature representation, we use GIST [29], and the a and b components of the Lab color space. We compute histograms of a and b from the shadow areas and their surroundings. For this, we use the initial annotated shadow mask and dilate it with an area ratio of 3:2 (shadow vs non shadow). We used a 30-bin histogram for the a and b features separately, and the original 512-bin histogram for the GIST feature.

PGP [46] groups data into clusters by finding and removing between-cluster edges from a weighted graph, where the graph nodes are the data points and the edges define neighborhood relationships where the pair-wise similarity distances are the edge weights. Given the graph, a two-component Weibull Mixture Model is fitted over the edge weights. Then, we use the cross-point of the two Weibull components as the critical value that represents the cut-off between the within-cluster edge weights and the between-cluster edge weights. After the critical value is computed, the edges with weights higher than the critical value are identified as between-cluster edges and removed, with the subsequent disjoint sets of sub-graphs as the final clustering result.

For the shadow image clustering problem, initial neighborhood relationships are not explicitly defined. Therefore, we construct the data graph by linking data nodes with their k nearest neighbors. Each node represents an image. We use Earth Mover's Distance (EMD) as the distance metric for the a and b color histograms, and Euclidean (L_2) distance for the GIST features. Given the three similarity distances per node pair, we normalize the EMD and L_2 distance values to have zero mean and unit variance, perform PCA, and take the first principal

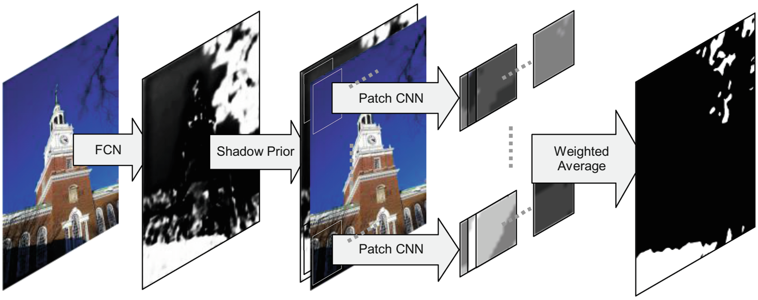

Fig. 3. The proposed pipeline for shadow segmentation. An FCN takes an RGB image and outputs an image level shadow prior map. Then a patch level CNN with structured output takes the RGBP (P is the image level shadow Prior channel) image and outputs a local shadow prediction map. Finally, the probability of each pixel being a shadow pixel is computed by averaging results from different patches.

component as the combined similarity distance for constructing the k nearest neighbor data graph.

Once the clusters are computed by applying PGP on the graph. We add a post-processing step to enforce the size of each cluster to be between $n_{min} = 10$ to $n_{max} = 60$ images. We iteratively merge small clusters (with less than n_{min} images) into the closest cluster. That is, the cluster that has the member with the lowest combined similarity distance to a member of the small cluster. Finally, we re-apply PGP to the clusters with sizes larger than n_{max} until the sizes of all resulting clusters fall within the desired range.

5 Shadow Segmentation Using Stacked-CNN

Most previous methods for shadow detection are based on classification of image regions using local color and texture cues. This approach, however, ignores global semantic information, which is useful for disambiguation. For example, without reasoning about global semantics, a dark cloud in the sky might be misclassified as a shadow region. In this section, we describe a semantics-aware patch level CNN, a method that combines global semantics with local cues for shadow detection.

Our method is based on the combination of two neural networks. Combining multiple neural networks has been successfully used in many applications [5,18,20,32,34,37]. One approach is to train multiple neural networks separately then combine their predictions [5,18,37]. Another approach is to combine the feature maps of neural networks instead of the final predictions [20]. These approaches, however, require the networks to share the same input/output structure and learning objective. Instead we propose to stack two CNNs into a single stream, as shown in Fig. 3. The two networks can have heterogeneous input/output representation and learning objectives.

We first train a Fully Connected Network (FCN) [26] on images with annotated shadow segmentation masks to predict a shadow probability map. Subsequently, the map predicted by the FCN for a training image is attached to the original RGB image as an additional channel. We refer to this channel as the image level shadow Prior channel P. Finally we train a CNN on RGBP patches to predict local shadow pixels, which will be referred to as patch-CNN. The final prediction of a pixel being a shadow pixel is a weighted average over the prediction outputs for all patches containing this pixel. The use of a patch-CNN in addition to an FCN has a "resolution" advantage. Although the deep layers of an FCN can extract semantic information, the spatial resolution is poor due to several max-pooling and down-sampling operations. Therefore, a local patch-CNN is necessary to refine the segmentation result. Furthermore, the patch-CNN learns from millions of training samples, leading to a more robust shadow classifier. By including the image level shadow prior channel in the input, we incorporate semantic information into the patch-CNN to generate improved shadow masks as shown in Fig. 4.

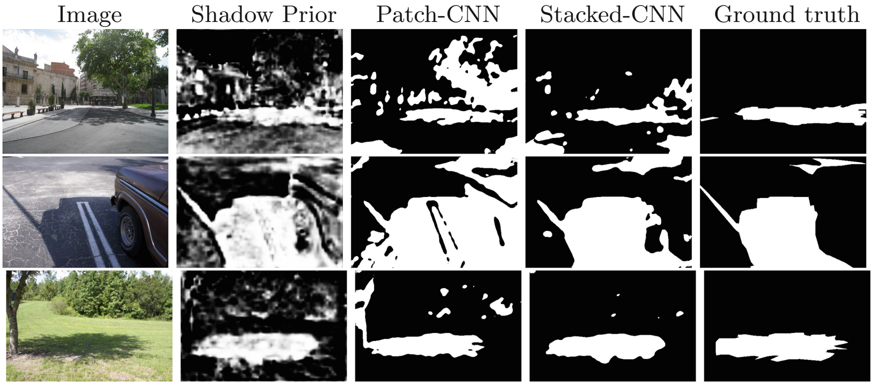

Fig. 4. Shadow segmentation examples. Qualitative results using patch-CNN on RGB images, and on RGBP (P is the image level shadow prior) images (stacked-CNN). The stacked-CNN achieves the best results by incorporating both semantic and subtle local texture and color information. For example, in the first image, although the color and texture of the tree is shadow-like, we can exclude the tree pixels thanks to the FCN generated shadow prior.

Semantic FCN details. We train a FCN [26] on images of various sizes to generate the image level shadow prior. We use the VGG-16 network [36], a CNN trained on a large scale object classification dataset, to initialize the semantic FCN. We fine-tune the semantic FCN using the given shadow masks. Because the initial FCN was trained for object classification, the resulting shadow probability maps contain semantic information.

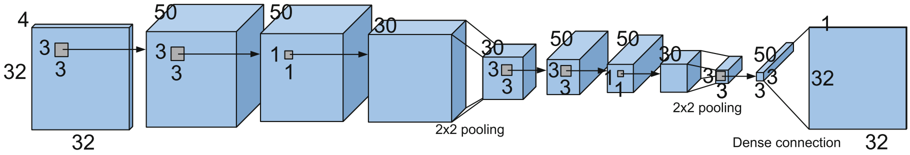

Fig. 5. Patch-CNN with structured output. The input is a 32×32 RGBP (RGB + image level shadow Prior) image, the output is a 32×32 shadow probability map.

Patch-CNN details. We build a patch level CNN with structured output for local shadow segmentation, as shown in Fig. 5. The loss function is the average negative log-likelihood of the prediction of every pixel. We extract image patches for training in three ways. Twenty-five percent of the patches are extracted at random image locations to include patches of various textures and colors. Fifty percent are extracted on Canny edges [4] to include hard-to-classify boundaries. Twenty-five percent are extracted at shadow locations to guarantee a minimum percent of positive instances. This results in an overall balanced number of shadow pixels and non-shadow pixels in the training batches for stochastic gradient descent. During testing, we feed all overlapping patches of each image to the patch-CNN. Thus every pixel has a maximum of $32 \times 32 = 1024$ predicted values from different patches. We use a weighted average to fuse multiple predictions. More precisely, suppose there are n patches containing the pixel, the distances between the pixel and the center of those patches are $d_1, d_2, \ldots, d_n$, and the predicted shadow probabilities are $p_1, p_2, \ldots, p_n$ respectively. Then the fused shadow probability is taken as: $p = (\sum_i G(d_i;\sigma)p_i)/\sum_i G(d_i;\sigma)$, where $G(d_i;\sigma)$ is a zero-mean Gaussian with variance σ^2. In our experiments we use $\sigma^2 = 8$.

6 A Large-Scale Shadow Dataset

We have collected a new shadow dataset, one that is significantly larger and more diverse than the existing datasets [12,49], and use lazy annotation [44] to quickly annotate the images. In this section we describe the details.

Image collection. To compile our dataset, we collected almost 5,000 images containing shadows. A quarter of the images came from the MS COCO dataset [25]. The rest were collected from the web. This image collection is significantly larger than the existing UCF [49] and UIUC [12] datasets, which contain less than 400 images combined. This image collection is also more diverse than existing datasets, which consist of images from a few specific domains (e.g., close shots of objects predominate in UIUC, whereas the majority images in UCF are scenes with darker shadows and objects). The image collection covers a wide range of scenes including urban, beach, mountain, roads, parks, snow, animals, vehicles, and houses. It also contains different picture types including aerial, landscape, close range, and selfies. We split the images into two subsets for training and testing. The training subset contains about 85 % of the images.

Shadow image annotation. We divided the image collection into disjoint train and test subsets and used two different approaches for annotation. For 700 test images, we carefully annotated the images, aiming for pixel accuracy to ensure the validity of numerical evaluation. We will refer to this test set as **SBU-Test**. For training images, we used *lazy labeling* to quickly annotate a large set of images. For lazy labeling, we drew a few strokes on shadow areas and a few other strokes on non-shadow areas. These strokes were used as shadow and non-shadow seeds for geodesic convexity image segmentation [11]. Figure 1 illustrates this procedure. With lazy labeling, we were able to annotate the dataset quickly, at the rate of 3 to 4 images per minute. However, the obtained annotation was noisy. In particular, there were many "dirty negatives" — shadow regions that were incorrectly labeled as negative. This was due to misclassification of shadow regions or poor segmentation (image regions contain both shadow and non-shadow pixels). Dirty negatives are more prevalent than "dirty positives". Since we focused on drawing strokes on major shadow areas, the chosen shadow areas were generally well segmented. The final dataset contains images with shadow labels that have been "cleaned" using the method described in Sect. 3. Hereafter, we refer to the dataset with noisy labels as **SBU-Train-Noisy** and the dataset with recovered labels as **SBU-Train-Recover**.

7 Experiments

We conducted experiments to evaluate our shadow detection method, the generalization ability of the proposed training dataset, and the effectiveness of the noisy shadow label recovery approach. Our newly collected dataset, SBU-Train-Recover contains 4085 training images. The dataset contains no images from existing shadow UCF and UIUC datasets.

For performance evaluation we compared the predicted shadow masks with the high quality annotation masks, measuring classification error rates at pixel level. The main performance metric is the Balanced Error Rate (BER). We avoid an overall error metric because shadow pixels are considerably less than non-shadow pixels, hence classifying all pixels as non-shadow would yield a low overall error.

CNN training details. We apply data augmentation: for the FCN training, we downsample the training images by six different factors: 1.0, 0.9, 0.8, 0.7, 0.6, 0.5 and perform left-right flip. For the patch-CNN training, we store original images in memory and randomly extract patches on the fly. Patches are randomly rotated and flipped. We use the implementation of the FCN provided by Long *et al.* [26]. We implement the patch-CNN using Theano [1,2]. The total training time of the stacked-CNN is approximately 10 h on a single Titan X GPU.

7.1 Shadow Segmentation Method Evaluation

We evaluate our shadow segmentation method on the UCF dataset [49]. We trained and tested on the original UCF dataset (255 images), using the split given by Guo *et al.* [12]. Measuring performance in terms of BER, our proposed method (stacked-CNN) performs comparably to several state-of-the-art methods[2]. Table 1 (left) shows that our method achieves lower BER than ConvNets+CRF [14], and the kernel optimization method (LooKOP+MRF) [43]. We also evaluate separately the different components of our architecture. As can be seen in Table 1 (right), the proposed stacked-CNN outperforms both the FCN and the patch-CNN. The 12 % reduction in BER compared to the patch-CNN confirms the benefits of using the FCN result as an image level shadow prior in our stacked-CNN architecture.

Table 1. Evaluation of shadow detection on UCF [49]. All methods are trained and tested on UCF training and test subsets. Our method stacked-CNN achieves better performance than state-of-the-art methods.

Method	BER	Sha.	Non.
Convnets+CRF [14]	17.7	27.5	7.9
LooKOP+MRF [43]	13.2	20.0	6.4
Stacked-CNN (ours)	**11.6**	10.4	12.8

Method	BER	Sha.	Non.
FCN	13.4	17.3	15.3
Patch-CNN on RGB	13.3	9.8	16.8
Stacked-CNN	**11.6**	10.4	12.8

Table 2. Experiments across datasets. Training on our dataset generalizes well on the UCF testing set, while the model trained on the UCF training set does not

Training Set	Methods	UCF Test			SBU-Test		
		BER	Sha.	Non-sha	BER	Sha.	Non-sha
UCF Train	LooKOP+MRF [43]	13.2	20.0	6.4	-	-	-
UCF Train	Stacked-CNN	**11.6**	10.4	12.8	13.9	13.1	14.7
SBU-Train-Recover	Stacked-CNN	13.0	9.0	17.1	**11.0**	9.6	12.5

7.2 Experiments with the SBU Datasets

We first evaluate the generalization ability of a classifier trained on our proposed dataset. We train the stacked-CNN on SBU-Train-Recover and test on UCF. As can be seen from Table 2, the stacked-CNN trained on SBU-Train-Recover achieves lower error than LooKOP+MRF [43] trained on UCF. Furthermore, training on SBU-Train-Recover slightly decreases the performance of the stacked-CNN as compared to training on UCF. This suggests that our stacked-CNN

[2] [35] cannot be directly compared because it used an extended version of the UCF dataset that is not publicly available.

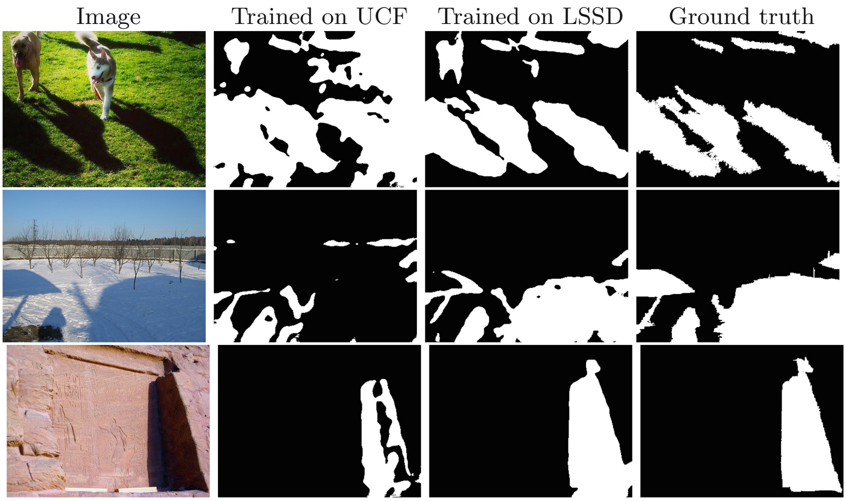

Fig. 6. Comparison of Stacked-CNN trained on UCF and SBU-Train-Recover. A stacked-CNN trained on a larger dataset shows improved shadow segmentation compared to a stacked-CNN trained on the UCF training set. Because SBU-Train-Recover contains a variety of scenes, the classifier trained on it is more robust on a general test set.

classifier trained on SBU-Train-Recover generalizes well to a totally different dataset. We also evaluate the performance of our proposed method on the newly collected testing set (SBU-Test). Our stacked-CNN achieves 11.0 % BER. In Fig. 6 we show qualitative results comparing the performance of out stacked-CNN trained on UCF and SBU-Train-Recover datasets.

7.3 Noisy Label Recovery Performance

For label recovery, PGP clusters SBU-Train-Noisy into 224 subsets of 10–60 images. To perform label recovery we allow up to 5 % negative and up to 1 % positive labels to be flipped ($\alpha = 0.99$, $\beta = 0.95$). We use our label recovery framework with $\mathcal{X}^2$ kernel as shadow region classifier. We choose the scaling parameter of the $\mathcal{X}^2$ kernel that minimizes the leave-one-out error on the noisy training set. We oversegment the training images into superpixels using Linear Spectral Clustering [48]. For each superpixel we compute intensity, color and texture features. We use 30 bin histograms for each of the channels of the CIELab color space. For texture, we use texton histograms. We run the full MR8 [42] filter bank on the input images and on the image density map [7]. Textons from density maps were shown to work well for shadow detection [7]. We cluster the filter responses, sampling 2,000 locations per image (balancing shadow and non shadow pixels), to build two 128-word dictionaries. Our method is able to flip labels and correct some annotation mistakes. Figure 7 shows examples of label recovery. New shadow boundaries are depicted in cyan.

Table 3. Label recovery influence on CNNs. We show the BER of the FCN, the patch-CNN, and the stacked-CNN trained on SBU-Train-Noisy and SBU-Train-Recover, and tested on the UCF testing set and SBU-Test.

Labels	FCN		Patch-CNN		Stacked-CNN	
	UCF test	SBU-Test	UCF test	SBU-Test	UCF test	SBU-Test
SBU-Train-Noisy	20.0	17.7	14.1	12.6	14.0	12.1
SBU-Train-Recover	**16.5**	**13.0**	**13.6**	**12.0**	**13.0**	**11.0**

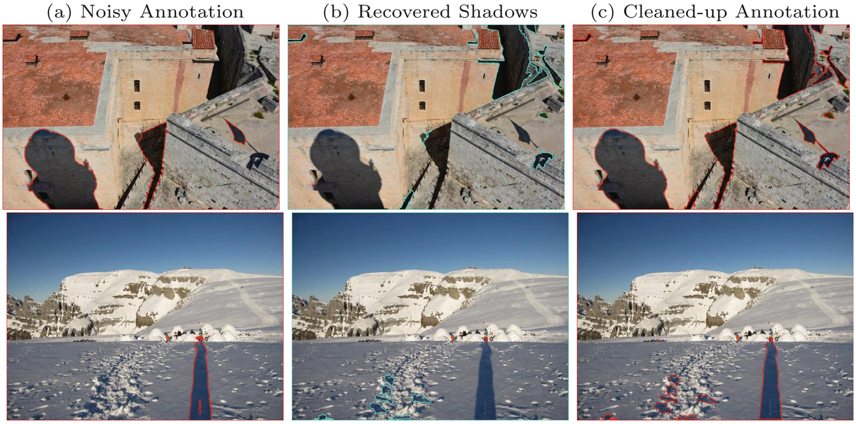

Fig. 7. Recovery from noisy annotations. Example of shadow region label recovery. (a) Original shadow annotation depicted with red boundaries. (b) Recovered shadows depicted with blue boundaries. (c) Resulting cleaned-up shadow annotation: shadow boundaries depicted in red. (Color figure online)

Since we could not quantitatively evaluate the proposed label recovery in a direct way, we measured the influence of training with noisy versus recovered labels in terms of classification performance. To expedite these experiments, we resized the training input images and corresponding shadow masks (for recovered and noisy) to be no bigger than 650 by 480 pixels. Then, we retrained our models using both recovered and noisy labels.

In Table 3, we compare the performance of the FCN, the patch-CNN, and the stacked-CNN when trained on SBU-Train-Noisy and SBU-Train-Recover and tested on the UCF testing set and the proposed SBU-Test. As can be seen, the models trained with recovered labels outperform models trained with noisy labels. Using recovered labels reduces the error rate of the stacked-CNN by 7 % and 9 % respectively, when testing in UCF and SBU-Test. Similarly, label recovery reduces the error rate of the FCN by 17.5 % and 26.5 %.

8 Conclusions

We have proposed a novel method for large-scale label recovery of noisily annotated shadow regions. This allowed us to create a new shadow dataset that is 20 times bigger than existing datasets. This dataset is well suited for deep-learning, and we proposed a novel deep learning framework to take advantage of the new dataset. Our deep learning architecture operates at the local patch level, but it can incorporate the global semantics. This leads to a shadow classifier that performs well across different datasets. We expect this new dataset to become the benchmark for large scale shadow detection.

Acknowledgments. Partially supported by NSF IIS-1161876, FRA DTFR5315-C00011, the Stony Brook SensonCAT, the Subsample project from DIGITEO Institute, France. The authors would like to thank Amazon for providing EC2 credits and NVIDIA for donating GPUs.

References

1. Bastien, F., Lamblin, P., Pascanu, R., Bergstra, J., Goodfellow, I.J., Bergeron, A., Bouchard, N., Bengio, Y.: Theano: new features and speed improvements. In: NIPS Workshop (2012)
2. Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., Bengio, Y.: Theano: a CPU and GPU math expression compiler. In: SciPy (2010)
3. Biggio, B., Nelson, B., Laskov, P.: Support vector machines under adversarial label noise. In: ACML (2011)
4. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **6**, 679–698 (1986)
5. Ciresan, D., Meier, U., Schmidhuber, J.: Multi-column deep neural networks for image classification. In: CVPR (2012)
6. Crammer, K., Lee, D.D.: Learning via Gaussian herding. In: NIPS (2010)
7. Ecins, A., Fermller, C., Aloimonos, Y.: Shadow-free segmentation in still images using local density measure (2014)
8. Finlayson, G., Hordley, S., Lu, C., Drew, M.: On the removal of shadows from images. IEEE Trans. Pattern Anal. Mach. Intell. **28**(1), 59–68 (2006)
9. Finlayson, G., Drew, M., Lu, C.: Entropy minimization for shadow removal. Int. J. Comput. Vis. **85**(1), 35–57 (2009)
10. Frenay, B., Verleysen, M.: Classification in the presence of label noise: a survey. IEEE Trans. Neural Netw. Learn. Syst. **25**(5), 845–869 (2014)
11. Gulshan, V., Rother, C., Criminisi, A., Blake, A., Zisserman, A.: Geodesic star convexity for interactive image segmentation (2010)
12. Guo, R., Dai, Q., Hoiem, D.: Single-image shadow detection and removal using paired regions (2011)
13. Guo, R., Dai, Q., Hoiem, D.: Paired regions for shadow detection and removal (2012)
14. Hameed Khan, S., Bennamoun, M., Sohel, F., Togneri, R.: Automatic feature learning for robust shadow detection (2014)
15. Hoai, M.: Regularized max pooling for image categorization (2014)

16. Hoai, M., Zisserman, A.: Improving human action recognition using score distribution and ranking (2014)
17. Huang, X., Hua, G., Tumblin, J., Williams, L.: What characterizes a shadow boundary under the sun and sky? (2011)
18. Ji, S., Xu, W., Yang, M., Yu, K.: 3D convolutional neural networks for human action recognition. PAMI **35**(1), 221–231 (2013)
19. Junejo, I.N., Foroosh, H.: Estimating geo-temporal location of stationary cameras using shadow trajectories. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008. LNCS, vol. 5302, pp. 318–331. Springer, Heidelberg (2008)
20. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: CVPR (2014)
21. Karsch, K., Hedau, V., Forsyth, D., Hoiem, D.: Rendering synthetic objects into legacy photographs. ACM Trans. Graph. **30**(6), 157 (2011)
22. Khardon, R., Wachman, G.: Noise tolerant variants of the perceptron algorithm. J. Mach. Learn. Res. **8**, 227–248 (2007)
23. Lalonde, J.F., Efros, A., Narasimhan, S.: Estimating natural illumination from a single outdoor image (2009)
24. Lalonde, J.-F., Efros, A.A., Narasimhan, S.G.: Detecting ground shadows in outdoor consumer photographs. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6312, pp. 322–335. Springer, Heidelberg (2010)
25. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Heidelberg (2014)
26. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR (2015)
27. Natarajan, N., Dhillon, I.S., Ravikumar, P.K., Tewari, A.: Learning with noisy labels. In: NIPS (2013)
28. Okabe, T., S.I., Sato, Y.: Attached shadow coding: estimating surface normals from shadows under unknown reflectance and lighting conditions (2009)
29. Oliva, A., Torralba, A.: Modeling the shape of the scene: a holistic representation fo the spatial envelope. Int. J. Comput. Vis. **42**(3), 145–175 (2001)
30. Panagopoulos, A., Samaras, D., Paragios, N.: Robust shadow and illumination estimation using a mixture model (2009)
31. Panagopoulos, A., Wang, C., Samaras, D., Paragios, N.: Simultaneous cast shadows, illumination and geometry inference using hypergraphs. IEEE Trans. Pattern Anal. Mach. Intell. **35**(2), 437–449 (2013)
32. Park, E., Han, X., Berg, T.L., Berg, A.C.: Combining multiple sources of knowledge in deep cnns for action recognition. In: WACV (2016)
33. Saunders, C., Gammerman, A., Vovk, V.: Ridge regression learning algorithm in dual variables (1998)
34. Sharkey, A.J.: Combining Artificial Neural Nets: Ensemble and Modular Multi-net Systems. Springer Science & Business Media (2012)
35. Shen, L., Chua, T.W., Leman, K.: Shadow optimization from structured deep edge detection (2015)
36. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR (2014)
37. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos (2014)
38. Stempfel, G., Ralaivola, L.: Learning SVMs from sloppily labeled data. In: Alippi, C., Polycarpou, M., Panayiotou, C., Ellinas, G. (eds.) ICANN 2009. LNCS, vol. 5768, pp. 884–893. Springer, Heidelberg (2009). doi:10.1007/978-3-642-04274-4_91

39. Stempfel, G., Ralaivola, L.: Learning kernel perceptrons on noisy data using random projections. In: Hutter, M., Servedio, R.A., Takimoto, E. (eds.) ALT 2007. LNCS, vol. 4754, pp. 328–342. Springer, Heidelberg (2007)
40. Suykens, J.A.K., Gestel, T.V., Brabanter, J.D., Moor, B.D., Vandewalle, J.: Least Squares Support Vector Machines. World Scientific, Singapore (2002)
41. Suykens, J.A.K., Vandewalle, J.: Least squares support vector machine classifiers. Neural Process. Lett. **9**(3), 293–300 (1999)
42. Varma, M., Zisserman, A.: Classifying images of materials: achieving viewpoint and illumination independence. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002. LNCS, vol. 2352, pp. 255–271. Springer, Heidelberg (2002)
43. Yago Vicente, T.F., Hoai, M., Samaras, D.: Leave-one-out kernel optimization for shadow detection (2015)
44. Yago Vicente, T.F., Hoai, M., Samaras, D.: Noisy label recovery for shadow detection in unfamiliar domains (2016)
45. Yago Vicente, T.F., Yu, C.P., Samaras, D.: Single image shadow detection using multiple cues in a supermodular MRF (2013)
46. Yu, C.P., Hua, W.Y., Samaras, D., Zelinsky, G.: Modeling clutter perception using parametric proto-object partitioning. In: NIPS (2013)
47. Yu, C.P., Le, H., Zelinsky, G., Samaras, D.: Efficient video segmentation using parametric graph partitioning. In: ICCV (2015)
48. Zhengqin, L., Jiansheng, C.: Superpixel segmentation using linear spectral clustering (2015)
49. Zhu, J., Samuel, K., Masood, S., Tappen, M.: Learning to recognize shadows in monochromatic natural images (2010)
50. Zhu, X., Wu, X.: Class noise vs. attribute noise: a quantitative study. Artif. Intell. Rev. **22**(3), 177–210 (2004)