2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Learning of Moving Cast Shadows for Dynamic Environments

Ajay J. Joshi and Nikolaos Papanikolopoulos
Department of Computer Science and Engineering
University of Minnesota-Twin Cities
{ajay,npapas}@cs.umn.edu

Abstract— **We propose a novel online framework for detecting moving shadows in video sequences using statistical learning techniques. In this framework, Support Vector Machines are applied to obtain a classifier that can differentiate between moving shadows and other foreground objects. The co-training algorithm of Blum and Mitchell is then used in an online setting to improve accuracy with the help of unlabeled data. We evaluate the concept of co-training and show its viability even when explicit assumptions made by the algorithm are not satisfied. Thus, given a small random set of labeled examples (in our application domain, shadow and foreground), the system gives encouraging generalization performance using a semi-supervised approach. In dynamic environments such as those induced by robot motion, the view changes significantly and traditional algorithms do not work well. Our method can handle such changing conditions by adapting online using a semi-supervised approach.**

I. Introduction

The problem of moving shadow detection has attracted significant interest in the vision community. In spite of its importance in various application domains, the problem remains unsolved because of several underlying difficulties. In robot vision in particular, the changing scene usually demands a technique that can adapt itself automatically. Even in transportation applications, the vision system is expected to run continuously for all times of the day. Shadow detection with fixed parameters tuned by humans for each different scene in such a scenario is infeasible. If manual tuning is not performed with changing scenes, it results in poor performance with traditional shadow detection techniques. This can be seen by how different tuning parameters are required by different videos [7]. One approach to tackle this difficulty is to use a learning method. In this paper, we approach the problem from a statistical machine learning perspective.

The entire literature on shadow detection techniques is too vast to review here. We mention a few representative methods for the reader. Prati et al. [1] present a comprehensive survey and analysis of most of the shadow detection algorithms published in the literature. They outline the strengths and limitations of different methods, and also study the conditions under which some techniques are better suited than others. See [2]-[7] for a variety of methods proposed. In most of these, shadow detection is done using either pixel-based parameters, or scene geometry information, or a combination of the two. There are three principal problems with the current methods:

1) Most methods require explicit tuning for each new video in large parameter spaces with multiple dimensions. This could be infeasible for real applications.
2) Most previous methods rely on assumptions made about image feature distributions, scene geometry, and other factors based on observations from a few videos. According to our experiments, these assumptions do not hold for general applicability.
3) Most techniques to detect shadows do not work in an online setting. System parameters once set, remain the same throughout, thus making the method error-prone to changes in the scene properties.

In this paper, we use a learning-based approach to address these problems. Instead of tuning the system for a new video, it is easy for a user to mark small regions belonging to shadow and foreground in one frame of any new video sequence. The system can then learn from these examples and "find" the right parameters for each scene in particular. Secondly, if a learning method can tune itself online, it can be of great help since the scene might change from time to time. We show how the learning algorithm can accomplish this using a semi-supervised approach.

Our work is similar in motivation to work by Porikli and Thornton [8]. They use Mixtures of Gaussians to model moving shadows. However they target only surveillance systems and use a predefined model that works well for that domain. Our paper paper provides a more general learning framework that can easily incorporate more features, and is applicable to other semi-supervised classification tasks as well.

II. Approach

The approach involves using effective pixel-based features from the frames of video to perform shadow detection. Specifically, we use four image features from [7], which have been shown to be effective in real world scenarios, indoors and outdoors. These features, computed for each pixel are summarized in the following: (1) The difference between edge gradient direction at the pixel in the current frame and that in the background model (f_1). Here, the background model is computed using the Mixtures of Gaussians technique [10], [11]; (2) The difference between edge strength at the pixel in the current frame and that in the background model (f_2); (3) The color distortion between the pixel in the current frame and that in the background model (f_3) [3], [7]. This distortion is computed from the 3D color model in the R, G, B space where each pixel is represented by a point; and (4) The ratio of intensity of the pixel in the current frame to that in the background model (f_4) [3], [7]. These four features (represented by feature vector f) attempt to extract useful information from the scene in terms of presence or absence of shadow on the particular pixel. See [7] for details.

We wish to solve the following classification problem. Given a feature vector $f \in \mathbb{R}^4$, we want to find the corresponding label $l \in \{S, FG\}$ where S indicates shadow and FG indicates foreground.

A. Support Vector Machines (SVM)

In the supervised learning case, the algorithm uses labeled data to find the hyperplane in feature space that separates the data with maximum margin. This hyperplane is then used for subsequent classification of new data.

The standard Support Vector Machine formulation [14] can be directly applied to the shadow detection problem, since we have cast it as a two-class classification problem. Support Vector Machines can learn complex separating surfaces in feature space. We thus do not make restrictive assumptions on the distribution of the data unlike most of the previous works on shadow detection, in which either some distribution is assumed, or an approximation is made to the histograms obtained from the available data [7].

The distributions assumed may not be accurate enough for future pixel classification. Even if empirical data is used to approximate these, some limitations exist. One is that the distribution may not remain the same for all scenes in differing conditions. Secondly, even on looking at the histograms, the final approximation is usually based on simple functions for modeling ease. In our earlier work [7], this modeling is done using sigmoid-like and Laplacian distributions which provide reasonable matches to the empirical observations.

Our objective was to make the approach more general, without relying on observations based on a few scenes. If we let the Support Vector Machine (or any learning algorithm) learn the decision boundary (which can actually be quite complex, and different for each scene), the method does not make restrictive assumptions as do previous methods.

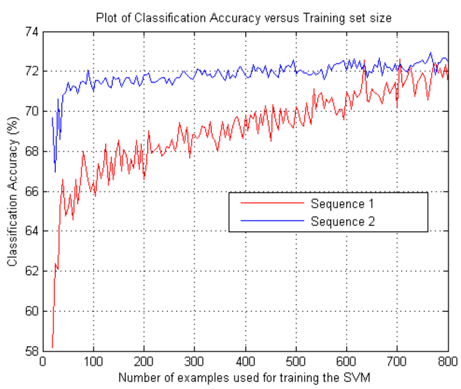


Fig. 1. The figure shows a plot of classification accuracy versus training set size for two datasets acquired from two different video sequences.

In Figure 1, we show a plot of classification accuracy using Support Vector Machines trained on different sizes of training sets (Sequence 1: 800 examples, Sequence 2: 10000 examples). The two results are for two different video sequences, one outdoor and one indoor. This is done to evaluate effectiveness across general scenes with differing illumination conditions and object sizes.

In all cases, we used the Radial Basis Function $K(X_i, X_j) = \exp\left(\frac{-\|X_i - X_j\|^2}{2}\right)$ as the kernel, and LIBSVM [15] was used to perform the tests. Accuracy values were obtained using 10-fold cross validation on each set.

B. Problems with using only one classifier

There are three main problems with using SVMs directly for shadow identification:

1) From Figure 1, it can be seen that as the training set size increases, classification accuracy improves for both datasets. This improvement continues for even larger sizes of training data. Also, improvement in accuracy is in varying degrees for both datasets. Thus, it is hard to predict what size of training data will provide "good" accuracy values for any new video sequence.
2) Because of the high computational cost associated with training Support Vector Machines with a large number of training examples, the use of a large training set might be infeasible.
3) Enough manually labeled data might not be available to learn from. In this case, accuracy will suffer since with smaller training set sizes, future classification accuracy is usually smaller as in Figure 1.

One solution is to use a semi-supervised learning technique, in which we can train the system using a small set of labeled data, and allow it to improve with the help of a possibly large number of unlabeled examples. This approach has been dealt with extensively, especially in the machine learning community [13], [16], [17]. The idea of co-training first proposed by Blum and Mitchell [13] is particularly attractive in our case for reasons described in the next section.

C. Co-training [13]

The original motivation to develop co-training came from the fact that labeled data is scarce, whereas unlabeled data is usually plenty and cheap to obtain. In this algorithm, two classifiers are trained using two different feature sets on the initial labeled data. Then each classifier is deployed on the unlabeled data, and at each round, it chooses the example which it can label most confidently from each class, and adds it to the pool of labeled examples. This is carried out iteratively until a fixed number of rounds, or until all the originally unlabeled data is labeled. Figure 2 outlines the algorithm. In this algorithm, F_1 and F_2 are two mutually exclusive feature sets which are part of the original feature vector f.

- Given two sets, one of labeled examples and one of unlabeled examples
- Loop for k rounds
  - Train a classifier C_1 using only F_1 part of the feature vector f with all labeled examples
  - Train a classifier C_2 using only F_2 part of the feature vector f with all labeled examples
  - Pick one unlabeled example in each class for which C_1 is most confident and add it to the set of labeled examples
  - Pick one unlabeled example in each class for which C_2 is most confident and add it to the set of labeled examples
- Return two classifiers C_1 and C_2. Labels for any new examples can be predicted by combining these classifiers, based on their confidence scores

---

Fig. 2. The Co-training Algorithm as proposed in [13], adapted from [9].

The idea behind co-training is the following. If the two classifiers are trained using conditionally independent feature sets, when one classifier labels an example, it is seen as a random training example by the other classifier. In this case, the other classifier benefits from this added example. In this way, different "views" of the target concept may help achieve better combined classification accuracy, even though individual classifier accuracy may be much weaker. We use the co-training approach to try and solve the problems caused by using SVMs alone as described in section II-B. This approach has two possible advantages:

1) It can use smaller training sets (with possibly large unlabeled data), so the running time of the algorithm is substantially reduced, and may be lower than the original, even if now, we train multiple classifiers.
2) For good future classification accuracy, enough training points might not be initially available to train on. In this scenario, performance will suffer when only SVMs are used. In the co-training setting, even if a small number of training examples are available, classification accuracy may be improved using the unlabeled examples. This is particularly useful for the application considered here, since smaller the manual labeling required, the easier for the user it is.

Although scarcity of labeled data is one consideration while choosing to use co-training, we also look for the added advantages of possibly improved speed of operation, and adaptive behavior (discussed later). Henceforth, the semi-supervised method refers to the one using co-training, and supervised refers to the method using only one SVM and no unlabeled data.

III. Setting the parameters for co-training

As mentioned previously, we use a feature vector f consisting of the four features for each pixel. The co-training algorithm performs best when the two feature sets used are conditionally independent, given the class label. Practically, in the best case, this would mean that the feature split should be done in such a way that the mutual information gain from the two sets is the minimum.

In our case, it is not obvious how to split the original feature vector to obtain a "good" split. Hence, we find a "good" feature split as required by co-training, empirically, using 50-fold cross validation tests on our vision data. The testing was done for various values of training set sizes and different number of co-training rounds. The results for two representative cases from our tests are summarized in Table I and Table II.

Table I shows classification accuracy results for a training set of size 100 (test set of about 800), and Table II shows classification accuracy on another dataset with a much larger (800 examples) training set size. The test set in this case was about 10000 examples. The two tables show behavior with vastly varying dataset sizes, and differing scenes, to eliminate any possible bias due to a particular size.

In the tables, Row 8 shows classification accuracy without using co-training (using only one classifier). The other rows show results with different feature splits. We note that in some cases (Rows 1,3,6), co-training degrades accuracy, while in others (Rows 2,4,5,7) it improves accuracy (as compared to Row 8) quite substantially. It is further vital to see that this improvement in accuracy comes for the same splits in both datasets. This observation confirms the fact that indeed, benefit to be achieved from co-training depends hugely on the different "views" of the example we obtain. Since in both cases the features used are the same, this throws some light on which features are 'more' independent of the other ones. It also emphasizes that the co-training algorithm can improve classification accuracy if we can find a "good" feature split. Now that "good" splits are found based on the tests, we choose one of them ([f_1 f_2] & [f_3 f_4]) for all experiments henceforth. Both splits giving good results are marked in boldface in the tables.

Row	Feature Split	Rounds of Co-training			
	$[Set1]$ $[Set2]$	20	50	100	200
1	$[f_1]$ $[f_2\ f_3\ f_4]$	61.96	60.56	59.91	60.20
2	$[f_2]$ $[f_1\ f_3\ f_4]$	74.27	73.48	73.28	72.89
3	$[f_3]$ $[f_1\ f_2\ f_4]$	57.00	57.89	59.21	60.41
4	$[f_4]$ $[f_1\ f_2\ f_3]$	**75.16**	**75.22**	**74.62**	**74.72**
5	$[f_1\ f_2]$ $[f_3\ f_4]$	**74.95**	**75.50**	**74.95**	**75.30**
6	$[f_1\ f_3]$ $[f_2\ f_4]$	61.72	60.95	61.22	61.63
7	$[f_1\ f_4]$ $[f_2\ f_3]$	73.59	73.09	73.01	73.24
8	No co-training	66.50			

TABLE I

Ppercentage classification accuracy with varying rounds of co-training and different feature splits on one dataset. Size - Training set:100, Test set:800.

Row	Feature Split	Rounds of Co-training			
	$[Set1]$ $[Set2]$	20	50	100	200
1	$[f_1]$ $[f_2\ f_3\ f_4]$	70.15	75.34	69.88	69.81
2	$[f_2]$ $[f_1\ f_3\ f_4]$	74.47	74.64	74.54	74.43
3	$[f_3]$ $[f_1\ f_2\ f_4]$	71.65	71.94	71.47	70.58
4	$[f_4]$ $[f_1\ f_2\ f_3]$	**76.63**	**76.82**	**76.55**	**76.26**
5	$[f_1\ f_2]$ $[f_3\ f_4]$	**76.06**	**75.86**	**75.52**	**75.19**
6	$[f_1\ f_3]$ $[f_2\ f_4]$	71.83	71.41	71.19	70.92
7	$[f_1\ f_4]$ $[f_2\ f_3]$	75.65	75.62	75.38	75.22
8	No co-training	72.34			

TABLE II

Percentage classification accuracy with varying rounds of co-training and different feature splits on another dataset. Size - Training set:800, Test set:10000.

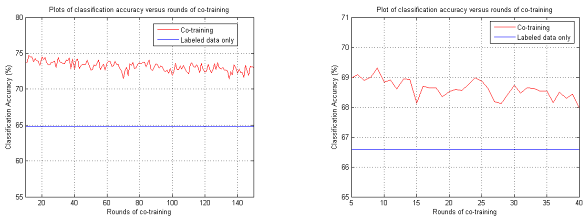
IV. Practical considerations in co-training

One important consideration is the size of the initial labeled set used for training. If this set is large, training time will be large for the resulting algorithm. In order for this framework to be useful, it is essential that the set of labeled examples be kept as small as possible. Secondly, co-training needs to run online, as new data becomes available. Smaller the number of co-training rounds, the better it is in terms of speed. Especially for vision applications, where frame processing rate is of significant importance, co-training should be done with as fewer rounds as possible.

In order to validate this, we ran tests with different number of rounds of co-training, starting from as low as 5, going up to 150 in some cases. Figure 3 (a) shows a plot of classification accuracy versus the number of rounds of co-training for the 'Intelligent Room' sequence (the 'Intelligent Room' sequence along with its ground truth data and other sequences used in Section VI are courtesy of the Computer Vision and Robotics Research Lab of UCSD). The training set size was 50 examples, while testing was done on 791 examples. Results are shown by averaging accuracy with 100-fold runs, with the training set chosen randomly at each fold. The curve in red shows accuracy with co-training, while the one in blue shows accuracy when only the initial set of labeled data is used. Figure 3 (b) shows a similar plot for the 'Highway' sequence. In this case, the training set size was 400, while test set size was 1000 examples. The number of rounds of co-training were varied from 5 to 40.

A. What happens with increasing co-training rounds?

It can be observed in both cases that increasing the number of rounds of co-training does not improve accuracy values much, if at all. At first, this seems contrary to our understanding of the algorithm, as well as the result obtained by Nigam and Ghani [9]. They show an experiment in which



(a) 'Intelligent Room'   (b) 'Highway'

Fig. 3. Variation of accuracy with varying rounds of co-training on two sequences. Note the improvement provided by co-training as opposed to using only labeled data.

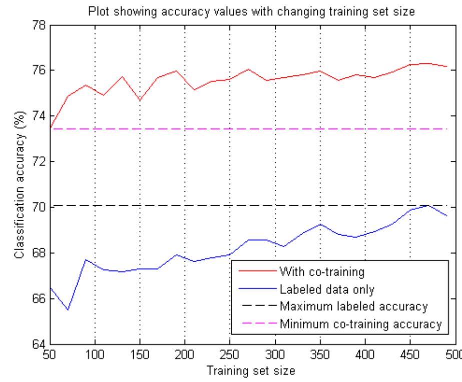


Fig. 4. Classification accuracy with changing training set sizes on the 'Intelligent Room' sequence. The co-training setting always performs better than the other even when the other uses much larger labeled set of examples.

increasing the rounds of co-training improves classification accuracy at every round. However, these results were only produced for a semi-artificial dataset they generate, keeping in mind the assumptions that co-training relies on. They combine two datasets to form one single dataset, such that the two separate feature sets they use are truly independent. We know from previous discussion that when this conditional independence assumption is satisfied, a new labeled example given by one classifier is seen as a random training example by the other, which can improve its performance. Therefore, it is very much expected that with more rounds of co-training (in effect more random labeled data for each classifier), the classifier accuracy improves.

The features we use are not independent in the first place. Secondly, our data comes from a vision system, which includes camera sensor noise, compression noise, and other artifacts expected from an ordinary camera system.

We believe that these (feature dependence and noise) are the reasons for the observation that increasing the number of rounds does not improve classifier accuracy at all times.

B. So how does this help us?

We can see from Figure 3 that even with a small number of co-training rounds, classification accuracy is better as compared to the accuracy obtained without using any unlabeled data. This is a very useful result, since we would like to use as small a number of rounds as possible, to speed up online classification.

These results, therefore show that with similar training set sizes, co-training does help improve classifier accuracy quite significantly. The question that remains is this: Can we use a larger initial labeled set (if available), and use supervised learning to achieve the same benefit in accuracy as that achieved by co-training?

We address this question in the following. Figure 4 shows a plot of classification accuracy in both settings with changing training set size. We observe that co-training always outperforms the supervised setting, with all data sizes. Furthermore, the figure also shows that accuracy obtained using co-training with only 50 labeled examples is better than accuracy obtained without co-training using up to 500 labeled examples. In other words, within this range, accuracy using only labeled examples for classification is always lower than the accuracy using unlabeled data. Similar results have been obtained on other datasets with extensive testing, and we show only a representative case for space constraints.

Two conclusions can be drawn from this:

1) By using unlabeled data, we achieve very good classification accuracy even with very small training set sizes. This backs the reduced computational cost argument made earlier. In this case, better classification accuracy was achieved even with a 10 times smaller training set.
2) Even if we can train classifiers with any large training set sizes, it is hard (if not impossible) to get comparable accuracy if we do not make use of unlabeled data.

These results justify the use of unlabeled examples for classification, in terms of speed as well as accuracy.

V. Results on Video Frames

In this Section, we provide some results on the actual video sequences. In the figures, blue indicates regions classified as foreground by the algorithm while red indicates regions classified as shadows.

Two important aspects should be considered when evaluating these results. One is that no global information is used for labeling the pixels. Secondly, we use an initial labeled set of examples sampled randomly from a labeled pool. This can lead to unbalanced sampling (a large difference between the number of training examples belonging to each label), which may deteriorate performance of the system. If a human manually labels a few pixels in both classes, the training data can be made much more balanced. We use a random sample so as to evaluate the concept of this method without relying on specific assumptions on the training set distribution.

In Figure 5, we use labeled data (100 points) from one frame of the video to construct the base classifiers. Then another frame is fed to the classifiers, and using 50 examples from the new frame, the combined classifier is built using co-training.

In the first 6 rows of Table III, we compare our results to other pixel-based methods (methods in which no post-processing is performed). The two metrics - Shadow Detection Accuracy (η) and Shadow Discrimination Accuracy (ξ), are defined in [1]. Detection accuracy aims to measure the percentage of true shadow pixels detected. Discrimination accuracy is an indicator of the false positive rate, higher the discrimination accuracy, lower is the false positive rate. We compare our results with those from [1] on the 'Intelligent Room' sequence. Ground truth data from this sequence was obtained from UCSD (109 manually marked frames), and it is the same that was used to obtain the results reported in [1], making the comparison fair (ground truth data for other sequences was not available). From Table III, we can see that Co-training gives favorable results compared to the

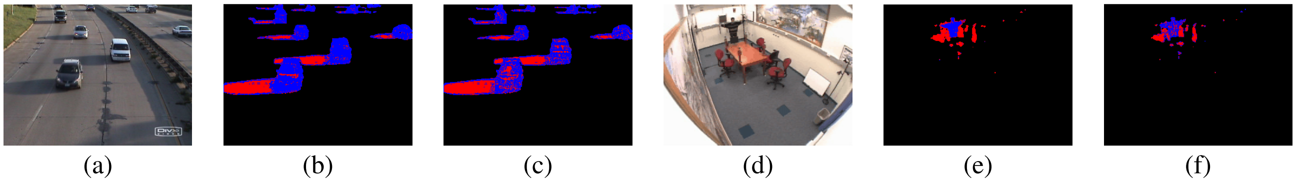


Fig. 5. Output of the combined classifier using co-training on different sequences. (a) and (d) show the original frames, (b) and (e) show output of the algorithm using co-training, and (c) and (f) show output using labeled data only. Note the drastic improvement in classification quality provided by the use of unlabeled data.

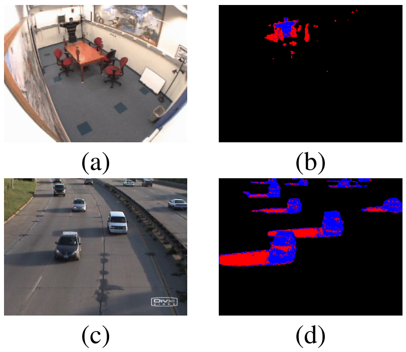


Fig. 6. (b) shows results when labeled data from the 'Highway' sequence is used, while (d) shows results when labeled data from the 'Intelligent Room' sequence is used.

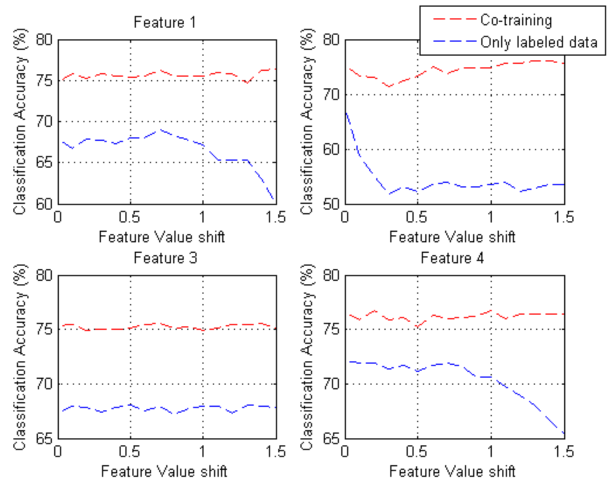


Fig. 7. Shows how co-training adapts to shifts in the feature values of the data. Supervised learning fails to capture this drift, however, with the help of unlabeled data, the classification boundary is adaptively altered.

best previous methods. The last 3 rows in the Table will be discussed in the following sections.

	η (%)	ξ (%)	Combined Score (Mean)
SNP [1]	72.82	88.90	80.86
SP [1]	76.27	90.74	83.50
DNM1 [1]	78.61	90.29	84.45
DNM2 [1]	62.00	93.89	77.94
Co-training	**86.49**	**92.27**	**89.38**
Co-training (Adaptive)	**81.23**	**85.12**	**83.18**
Method in [7] (post-processing)	87.99	97.08	92.54
Co-training + post-processing	**91.12**	**97.55**	**94.33**

TABLE III

COMPARISON OF OUR RESULTS WITH THOSE REPORTED BY PRATI ET AL. [1] AND JOSHI ET AL. [7]. OUR RESULTS ARE SHOWN IN BOLDFACE.

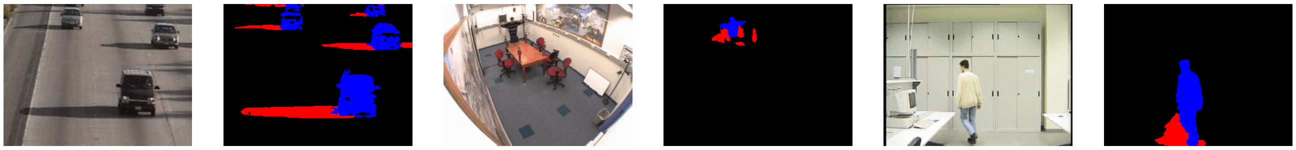
A. *Co-training is adaptive*

Figure 6 shows results obtained by using an initial labeled set from the 'Intelligent Room' sequence and testing on the 'Highway sequence' and vice versa.

Observing the intensity of shadow and foreground regions in the two sequences, we see that they differ widely. One of the features we use is the ratio of intensity of the pixel in the current frame to the intensity of the pixel in the background model. For the 'Highway' sequence, shadow regions have this value in the range of 0.5 to around 0.75, whereas for the other video, this ratio ranges from 0.65 to about 0.95. Even with this difference in the feature values, training on the labeled set of one, and then using co-training on the other produces good results. This is surprising, and very useful at the same time. Other methods need to explicitly tune the parameters for each particular video. We do not need to do that in this co-training setting, which shows an interesting adaptive behavior of the algorithm.

One row of Table III (Co-training (Adaptive)) shows Detection and Discrimination Accuracy on the 'Intelligent Room' sequence, when trained on 100 examples from an outdoor sequence. This still shows comparable values to other methods, showing adaptive behavior. Note that the other methods have been fine-tuned to work well for the particular sequence explicitly. This observation is extremely significant, since it has important implications in vision systems. As per our knowledge, none of the previous works have reported this behavior.

In order to see if this behavior is exhibited for a broader spectrum of data from different sources, we carried out more comprehensive experiments. Figure 7 shows results for a frame of video extracted from our vision system. In each of the four plots, the following is done. The data is split into two parts. We train classifiers on one part of the data. Then, we modify one of the feature values for all examples in the other part, keeping the same class labels. The classifiers are then deployed on this part, and accuracy values are obtained. On the X-axis, the amount of feature shifts are noted. This feature shift is not absolute. The values mentioned are the amount by which a feature value is shifted scaled by its mean value. For example, 1.2 on the X-axis represents a shift of 1.2 times the mean of feature value for all examples ($f_i \leftarrow f_i \pm 1.2 \cdot \text{mean}(f_i)$). Hence, note that a value of 1.5 is a large shift in feature space. The four plots show one feature shifted in each. The plots on the top left and bottom right show that with increasing shift of feature values, the classifier trained using SVM alone deteriorates in accuracy smoothly. However, the plot at the top right of the figure



(a) (b) (c) (d) (e) (f)

Fig. 8. Shadow detection results with semi-supervised learning and post-processing. (a), (c), and (e) show the original frames. (b), (d), and (f) show detection results using our technique

shows a drastic lowering in accuracy at one point, and it stays at that low value since. We get some understanding of the underlying feature distribution from the behavior of these plots. In case there is a high population of examples in feature space near the classification boundary, shifting feature values will possibly impact the classifier performance acutely. This could be the cause of the sudden drop of accuracy when a particular value of shift is made. On the other hand, if population of examples in feature space is small near the classifier boundary, shifting feature values might not affect the classifier accuracy values much. The plots at the top left and bottom right in Figure 7 show such a behavior.

In all cases, note that the classifier that uses co-training maintains high accuracy values for the entire range of feature shifts. In a sense, it adapts to the new classification boundary with the help of unlabeled examples. Intuitively, this could be attributed to the two distinct "views" of the example obtained from the two separate classifiers. This aspect is very important, especially for computer vision systems. In many vision tasks, we know what features best represent the knowledge based on which we wish to distinguish between different classes. Even though these features are the same, their values might change from one scene to another (based on illumination conditions, scene geometry, sensor variation, background etc.). This is countered by tuning the system manually, usually by trial and error, or exhaustive search. A semi-supervised framework like the one presented in this paper holds tremendous potential for similar vision problems, in which the system needs to adapt online to scene changes.

VI. Results with post-processing

So far, we have used only pixel-based cues for classification. For better performance, it is important to use more surrounding information for classification, to preserve homogeneity between pixels belonging to the same region. We use the region post-processing technique from our previous work [7]. This helps remove noise in the classification, and also uses neighborhood information in case the pixel labels are ambiguous. Results with region-based post-processing are shown in Figure 8. These results show a drastic improvement in performance over previous methods [1], [7]. See the last 2 rows of Table III for details.

In summary, our framework is general enough to accommodate new features that might be found useful for classification. Also, in most cases, our algorithm is completely parameter-free because of automatic tuning of the learning technique. This contrasts with heavy dependence on parameters as in recent approaches [7].

VII. Conclusions

In this paper, we presented and evaluated a new framework based on semi-supervised learning for detecting moving shadows. This is especially applicable in demanding scenarios like vision systems mounted on mobile robots. In applications like Intelligent Transportation systems as well, such a scheme is highly effective because of its ability to adapt to changing conditions.

Acknowledgments

This work has been supported in part by MnDOT, the ITS Institute at the University of Minnesota, and the National Science Foundation through grants #IIS-0219863, #CNS-0224363, #CNS-0324864, #CNS-0420836, #IIP-0443945, #IIP-0726109, and #CNS-0708344.

References

[1] A. Prati, I. Mikic, M. M. Trivedi, and R. Cucchiara, "Detecting Moving Shadows: Algorithms and Evaluation", In *IEEE TPAMI*, vol. 25, no. 7, pp. 918–923, July 2003.

[2] J. Stauder, R. Mech, and J. Ostermann, "Detecting of Moving Cast Shadows for Object Segmentation", In *IEEE Trans. Multimedia*, vol. 1, no. 1, pp. 65–76, March 1999.

[3] T. Horprasert, D. Harwood, and L. S. Davis, "A Statistical Approach for Real-Time Robust Background Subtraction and Shadow Detection", In *Proc. ICCV Frame-Rate Workshop*, 1999.

[4] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting Objects, Shadows, and Ghosts in Video Streams by Exploiting Color and Motion Information ", *Proc. Eleventh Int'l. Conf. Image Analysis and Processing*, pp. 360–365, September 2001.

[5] I. Mikic, P. Cosman, G. Kogut, and M. M. Trivedi, "Moving Shadow and Object Detection in Traffic Scenes", *Proc. ICPR*, vol. 1, pp. 321–324, September 2000.

[6] M. Kilger, "A Shadow-Handler in a Video-Based Real-Time Traffic Monitoring System", *Proc. IEEE Workshop on Applications of Computer Vision*, pp. 11–18, 1992.

[7] A. J. Joshi, S. Atev, O. Masoud, and N. Papanikolopoulos, "Moving Shadow Detection with Low- and Mid-Level Reasoning", In *Proc. ICRA*, 2007.

[8] F. Porikli and J. Thornton, "Shadow Flow: A recursive method to learn moving cast shadows", In *Proc. ICCV*, 2005.

[9] K. Nigam and R. Ghani, "Analyzing the effectiveness and applicability of co-training", *Proc. 9th Int'l Conf. Information and Knowledge Management*, pp. 86–93, 2000.

[10] C. Stauffer and W. E. L. Grimson, "Adaptive Background Mixture Models for Real-Time Tracking", In *Proc. CVPR*, vol. 2, 1999.

[11] S. Atev, O. Masoud, and N. Papanikolopoulos, "Practical Mixtures of Gaussians with Brightness Monitoring", *Proc. IEEE Seventh Int'l Conf. ITS*, pp. 423–428, October 2004.

[12] V. Vapnik, "Statistical Learning Theory", *Wiley*, 1995.

[13] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training", *Proc. 11th annual Conf. Computational learning theory*, pp. 92–100, 1998.

[14] J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition", *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[15] C. Chang and C. Lin, "LIBSVM: a library for support vector machines", 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[16] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using EM", *Machine Learning*, vol. 39, no. 2/3, pp. 103–134, 2000.

[17] S. A. Goldman and Y. Zhou, "Enhancing Supervised Learning with Unlabeled Data", *Proc. ICML*, pp. 327–334, 2000.

[18] D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz, "UCI Repository of machine learning databases", University of California, Department of Information and Computer Science. [http://www.ics.uci.edu/ mlearn/MLRepository.html].