Online Detection and Classification of Moving Objects Using Progressively Improving Detectors

Omar Javed
Computer Vision Lab,
University of Central Florida
Orlando, FL, U.S.A
ojaved@cs.ucf.edu

Saad Ali
Computer Vision Lab,
University of Central Florida
Orlando, FL, U.S.A
sali@cs.ucf.edu

Mubarak Shah
Computer Vision Lab,
University of Central Florida
Orlando, FL, U.S.A
shah@cs.ucf.edu

Abstract

Boosting based detection methods have successfully been used for robust detection of faces and pedestrians. However, a very large amount of labeled examples are required for training such a classifier. Moreover, once trained, the boosted classifier cannot adjust to the particular scenario in which it is employed. In this paper, we propose a co-training *based approach to continuously label incoming data and use it for online update of the boosted classifier that was initially trained from a small labeled example set. The main contribution of our approach is that it is an* online *procedure in which separate views (features) of the data are used for co-training, while the combined view (all features) is used to make classification decisions in a single boosted framework. The features used for classification are derived from Principal Component Analysis of the appearance templates of the training examples. In order to speed up the classification, background modeling is used to prune away stationary regions in an image. Our experiments indicate that starting from a classifier trained on a small training set, significant performance gains can be made through online updation from the unlabeled data.*

1. Introduction & Related Work

The detection of moving objects, specifically people or vehicles, in a scene is of utmost importance for most surveillance system. In recent years, considerable progress has been made for detection of faces and pedestrians through supervised classification methods. In this context, a variety of approaches have been used including naive Bayes classifiers [9], Support Vector Machines [7] and Adaboost [10]. Specifically for surveillance related scenarios, Adaboost is particularly suitable since it has been demonstrated to give high detection rates using simple Haar-like features in real-time [10]. However, one problem in training such a classifier is that an extremely large number of training examples are required to ensure good performance in the test phase. For example, Zhang et al. [11] used around 11000 positive and a 100000 negative labeled images for face detection. Another issue related to the use of Boosted classifiers in the surveillance scenario is that the classifier parameters are fixed in the test stage. However it is preferable to have a system that automatically learns from the examples in a specific scenario.

One possible way around the requirement of a large labeled training set is the co-training approach proposed by Blum and Mitchell [1]. The basic idea is to train two classifiers on two independent "views" (features) of the same data, using a relatively small number of examples. Then to use each classifier's prediction on the unlabeled examples to enlarge the training set of the other. Blum and Mitchell prove that co-training can find a very accurate classification rule, starting from a small quantity of labeled data if the two feature sets are statistically independent. However, this assumption does not hold in many realistic scenarios [8].

Levin et al. [5] use the co-training framework, in the context of boosted binary classifiers. Two boosted classifiers are employed for co-training. If one classifier predicts a label for a certain example with a high confidence then that labeled example is added to the training set of the other, otherwise the example is ignored. One of the two boosted classifiers employed for co-training uses background subtracted image regions, while the other classifier is trained on the image grey-levels directly. Note that the features are closely related. However, their approach empirically demonstrates that co-training is still possible even in the case the independence assumption does not hold. The co-training based learning approach have also been used successfully for text retrieval and classification by Collins and Singer [2].

One important point to note is that co-training is not a classification framework. It is actually a training method from unlabeled data. Co-training requires two separate

views of the data for labeling, however a better classification decision can be made by combining the two views of data by a fully trained classifier. Thus, co-training is used to train a classifier in an *offline* setting. Once training is complete the combined view is used to make classification decisions. The principal contribution of our approach is that it is an *online* method, in which separate views (features) of the data are used for co-training, while the combined view is used to make classification decisions in a single framework. To achieve this, we have exploited the fact that the the boosted classifier is a linear combination of simpler 'base' classifiers and the adaptive boosting selection mechanism discourages high correlation among the selected features. The co-training is performed through the base classifiers, i.e., if a particular unlabeled example is labeled very confidently by a small subset of the base classifiers then it is used to update both the base classifiers and the boosting parameters using an online variant of the multi-class Adaboost.M1 algorithm [6]. Note that only few of the observed examples might qualify for co-training. Meanwhile the classification decision for each example is made by the boosted classifier, whose parameters have been updated from the labeled examples observed so far. The advantage of this approach is that the classifier is attuned to the characteristics of a particular scene. Note that, a classifier trained to give the best average performance in a variety of scenarios will usually be less accurate for a particular scene as compared to a classifier trained specifically for that scene. Obviously, the specific classifier would not perform well in other scenarios and thus it will not have widespread application. Our proposed approach tackles this dilemma by using a classifier trained on a general scenario that can automatically label examples observed in a specific scene and use them to fine tune its parameters online.

We demonstrate the performance of our classifier in the context of detection of pedestrians and vehicles observed through fixed cameras. In the first step of our detection framework, we use a background model [4] to select regions of interest in the scene. The boosted classifier searches within these regions and classifies the data into pedestrians, vehicles and non-stationary background. Co-training decisions are made at the base classifier level, and both the base classifiers and the boosting parameters are updated online.

In the next section we discuss the features used for object representation and the base classifiers learned from these features. In Section 3, we describe the co-training framework in the context of an online boosted classifier. In Section 4, we present the results and give the concluding remarks in Section 5.

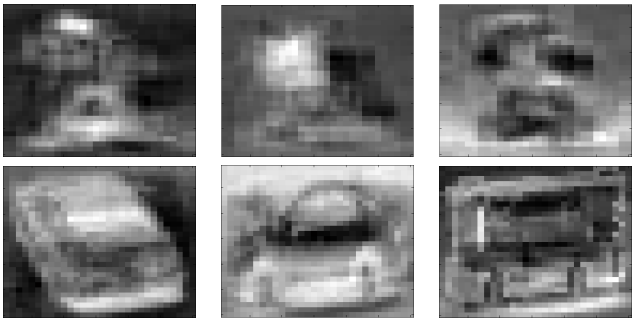

Figure 1. First Row: The top 3 eigenvectors for the pedestrian subspace. Second Row: The top eigenvectors for the vehicle subspace.

2. Feature Selection and Base Classifiers

One approach for object representation in boosted classifiers is to use local Haar like features. The advantage of using the Haar features is that they can be calculated very efficiently [10]. However, it has been shown, in the context of face detection, by Zhang et al. [11] that base classifiers trained from global features are more reliable and the resulting boosted classifier has a higher detection rate. The drawback is that global features are usually more expensive to compute. However, in our approach, background subtraction is used to discard most of the stationary regions in an image before further processing, therefore we can afford to use global features for classification and still handle real-time processing requirements.

We employ Principal Component Analysis (PCA) to obtain the global features. The principal component model is formed by taking m example images of dimensionality d in a column vector format, subtracting the mean, and computing the $d \times d$ dimensional covariance matrix C. The covariance matrix is then diagonalized via an eigenvalue decomposition $C = \Phi E \Phi^T$, where Φ is the eigenvector matrix and E is the corresponding diagonal matrix of its eigenvalues. Only m eigenvectors, corresponding to the m largest eigenvalues are used to form a projection matrix S_m to a lower dimension subspace.

We construct a pedestrian subspace with a $d \times m_1$ dimensional projection matrix $\mathbf{S}_{m_1}$ and a vehicle subspace with a $d \times m_2$ dimensional projection matrix $\mathbf{S}_{m_2}$ by performing PCA on the gradient magnitudes of respective training images. The parameters m_1 and m_2 are chosen such that the eigenvectors account for 99% of the variance in pedestrian and vehicle data respectively. The top three eigenvectors for the pedestrians and vehicles are shown in Figure 1. The features for the base learners are obtained by projecting each training example $\mathbf{r}$ in the two subspaces and obtaining a fea-

Online Co-Train

-if atleast k base classifiers confidently predict a label c_p for incoming example x, where $p \in \{1, \ldots, numclasses\}$, then

- if $\left((\sum_{n:h_n(x)=c_p} log(\frac{1}{\beta_n})) / (\sum_{n=1}^{N} log(\frac{1}{\beta_n})) \right) < T_{c_p}^{ada}$
  - $\beta_1, \ldots, \beta_N \leftarrow OnlineBoosting(H_N, x, c_p)$
  - add example with assigned label c_i to the validation set.
  - for $j = 1, \ldots, N$
    - for $i = 1, \ldots, numclasses$
      - T_{j,c_i}^{base}=max posterior probability, for class c_i by h_j, of a negative example in the validation set
  - for $i = 1, \ldots, numclasses$
    - $T_{c_i}^{ada}$=max H_N normalized score, for class c_i, of a negative example in the validation set

---

returns $\beta_1, \ldots, \beta_N$:$\mathbf{OnlineBoost}(\mathbf{H_N}, \mathbf{x}, \mathbf{label})$

-Set the example's initial weight $\lambda_x = 1$.

- For each base model h_n,in the boosted classifier

1. Set z by sampling Poisson(λ_x).
2. Do z times : $h_n \leftarrow OnlineBase(h_n, x, label)$
3. if $h_n(x)$ is the correct label,
   - $\lambda_n^{sc} = \lambda_n^{sc} + \lambda_x$, $\epsilon_n = \frac{\lambda_n^{sw}}{\lambda_n^{sc} + \lambda_n^{sw}}$, $\lambda_x = \lambda_x(\frac{1}{2(1-\epsilon_n)})$
4. else
   - $\lambda_n^{sw} = \lambda_n^{sw} + \lambda_x$, $\epsilon_n = \frac{\lambda_n^{sw}}{\lambda_n^{sc} + \lambda_n^{sw}}$,$\lambda_x = \lambda_x(\frac{1}{2(\epsilon_n)})$
5. calculate $\beta_n = log(\frac{1-\epsilon_n}{\epsilon_n})$

Figure 2. The co-training method. Note that both T^{base} and T^{ada} are automatically computed from the validation set. The subfunction $onlineboost()$ was proposed in [6]. λ_n^{sc} are sum of weights for examples that were classified correctly by the base model at the stage n while λ_n^{sw} is sum for incorrectly classified examples.

ture vector $\mathbf{v} = [v_1, \ldots, v_{m_1}, v_{m_1+1}, \ldots, v_{m_1+m_2}]$, where

$$\begin{aligned} \left[v_1, \ldots, v_{m_1}\right] &= \mathbf{r}^T \mathbf{S}_{m_1}, \\ \left[v_{m_1+1}, \ldots, v_{m_1+m_2}\right] &= \mathbf{r}^T \mathbf{S}_{m_2}. \end{aligned}$$

We construct each base classifier from a single subspace coefficient. Thus we will have a total of $m_1 + m_2$ base classifiers.

We use the Bayes classifier as our base classifier. Let c_1, c_2 and c_3 represent the pedestrian, vehicle and the non-stationary background classes respectively. The classification decision by the q^{th} base classifier is taken as c_i if $P(c_i|v_q) > P(c_j|v_q)$ for all $j \neq i$. The posterior is given by the Bayes rule, i.e., $P(c_i|v_q) = \frac{p(v_q|c_i)P(c_i)}{p(v_q)}$. The pdf $p(v_q|c_i)$ is approximated through smoothed 1D histogram of the of the q^{th} subspace coefficients obtained from the training data. The denominator $p(v_q)$ is calculated as $\Sigma_{i=1}^{3} p(v_q|c_i)p(c_i)$. Note that the sum of posterior probabilities over all classes for a particular coefficient instance is one, i.e., for the three class case, $\Sigma_{i=1}^{3} P(c_i|v_q) = 1$.

Once the base classifiers are learned, the next step is to train the boosted classifier from the initial set of labeled data. We use the Adaboost.M1 algorithm [3] for learning the boosted classifier. In the next section, we discuss the co-training framework for augmenting the initial training set.

3. The Co-Training Framework

Boosting is an iterative method of finding a very accurate classifier by combining many base classifiers, each of which may only be moderately accurate. In the training phase of the Adaboost algorithm , the first step is to construct an initial distribution of weights over the training set. Then the boosting mechanism selects a base classifier that gives the least error, where the error is proportional to the weights of the misclassified data. Next, the weights associated with the data misclassified by the selected base classifier are increased. Thus the algorithm encourages the selection of another classifier that performs better on the misclassified data in the next iteration. If the base classifiers are constructed such that each classifier is associated with a different feature, then the boosting mechanism will tend to select features that are not completely correlated. Note that, for co-training we require two classifiers trained on separate features of the same data. Therefore, we propose to label the unlabeled data by using the base classifiers selected by Adaboost. Basically, if a base classifier selected through the boosting mechanism confidently predicts the label of the data, then we can add this data to our training set to update the rest of the classifiers. The confidence thresholds for the base classifiers can be determined through the training data or by using a small validation set.

Suppose H_N is the boosted (strong) classifier learned through the Adaboost.M1 [3] algorithm. Let h_j, where $j \in \{1, \ldots, N\}$, be the base classifiers selected by the boosting algorithm. In order to set confidence thresholds on the labels given the base classifiers, we use a validation set of labeled images. For the class c_i, the confidence threshold T_{j,c_i}^{base} is set to be the highest posterior probability achieved by a negative example. This means that all examples in the validation set labeled as c_i by h_j with a probability higher than T_{j,c_i}^{base} actually belong to the class c_i. Thus during the online phase of the classifier, any example which has a probability higher than T_{j,c_i}^{base} is very likely to belong to the class c_i. The thresholds for all base classifiers selected by the boosting algorithm are similarly calculated.

Ideally, if a single base classifier confidently predicts a label with a probability higher than the established thresh-

old then we should assume that the label is correct and use that example for further training the classifier. However, training from only a few wrongly labeled examples can severely degrade the performance of the classifier. Therefore, we choose to be more conservative and only select an unlabeled example if k, where $k \approx .1N$, base classifiers confidently label an example.

It would be very inefficient to use every confidently labeled example for online training. The example labeled through co-training will improve the performance of the boosted classifier only if it has a small or negative margin, i.e., if the example lies close to the decision boundary in the solution space. If the example has been labeled unambiguously by the boosted classifier, i.e., it has a large margin, then using it for training will have little effect on the boosted classifier. Thus, we need unlabeled examples which have a small (or negative) margin and are also confidently labeled by the base classifiers. The limits on the score of the boosted classifier can also be established through the validation set. The score of an example for the label c_i is computed by Adaboost.M1 as $\Sigma_{n:h_n(x)=c_i} log(\frac{1}{\beta_n})$, where β_n is the coefficient of the n^{th} classifier selected by the algorithm. The label that gets the highest score is assigned to the example. For the class c_i, the threshold to determine the usefulness of employing the example for retraining, i.e., $T_{c_i}^{ada}$, is set to be the highest normalized score achieved by a negative example. Thus an example, assigned the label c_i by base classifiers should only be used for retraining if it gets a score of less than $T_{c_i}^{ada}$ by the boosted classifier.

Once an example has been labeled and if it has a small margin, the next issue is to use this example for updating the boosting parameters and the base classifiers online. The co-training and online updation algorithm is given in Figure 2.

3.1 Online Learning

Note that an online algorithm does not need to 'look at' all the training data at once, rather it process each training instance without the need for storage and maintains a current hypothesis that has been learned from the training examples encountered so far. To this end we use an online boosting algorithm proposed by Oza and Russel [6]. The inputs to the algorithm are the current boosted classifier H_N, the constituent base classifiers, and parameters λ_n^{sc} and λ_n^{sw}, where $n = 1, \ldots, N$. λ_n^{sc} and λ_n^{sw} are the sums of the weights of the correctly classified and misclassified examples, respectively, for each of the N base classifiers.

The main idea of the algorithm is to update each base classifier and the associated boosting parameter using the incoming example. The example is assigned a weight λ at the start of the algorithm. For the first iteration, the base classifier is updated z times, where $z = Poisson(\lambda)$. Then, if h_1 misclassifies the example, λ_1^{sw} is updated which is the sum of weight of all incorrectly classified examples by h_1. The weight of the example λ is increased and it is presented to the next base classifier. Note that in the regular 'batch' Adaboost method the weight of the example is also increased in case of misclassification. However, all the weights are assumed to be known at the next iteration. In the online boosting method only the sums of weights of correctly classified and misclassified examples (seen so far) are available. The boosting parameters, $\beta_1, \ldots, \beta_N$, are updated using these weights. Also, since the boosted classifier continues to learn online, no base classifiers are actually permanently discarded. however, to classify a new example, we only use the first k base classifiers such that classifier $k+1$ has the normalized error greater than random guess on the training data observed till then [6]. The algorithm also needs to update the base classifiers online. Since our base classifiers are represented as normalized histograms, they can easily be updated, i.e., the training example is added to the histogram representing the probability distribution of the feature, and the histogram is re-normalized. The online learning algorithm is shown in the bottom half of Figure 2.

4. Results

For the initial training of the multi-category classifier, we used 50 training images per class. Images of pedestrians and vehicles from a variety of poses were used. For the non-stationary background class, we selected the scenarios where the background modeling is likely to fail, for example sporadically moving tree branches, or waves in a pond. All extracted objects were scaled to the same size (30x30 pixels). Features were obtained by projecting gradient magnitudes of the regions in the pedestrian and vehicle subspaces. The base and boosted classifier thresholds were determined for a validation set consisting of 20 images per class for a total of 60 images.

We evaluated our algorithm for person and vehicle detection in three different locations. In each location, the view consisted of the road, with walkways near by. The pedestrian and vehicular traffic along the paths was fairly consistent. We demonstrated the improvement through online co-training at each location in two different ways. Firstly, we divided the sequences in equal size chunks and show that classification accuracy improves with time through online learning. Figure 4 shows classification results over two minute subsets for the three sequences. Note that with the exception of one interval in the second sequence, the performance either consistently improves with time or remains stable. The performance measure was the classification accuracy, i.e., the percentage of the number of valid vehicle and pedestrian detections to the total number of detections.

For further analysis of the method, we divided each se-

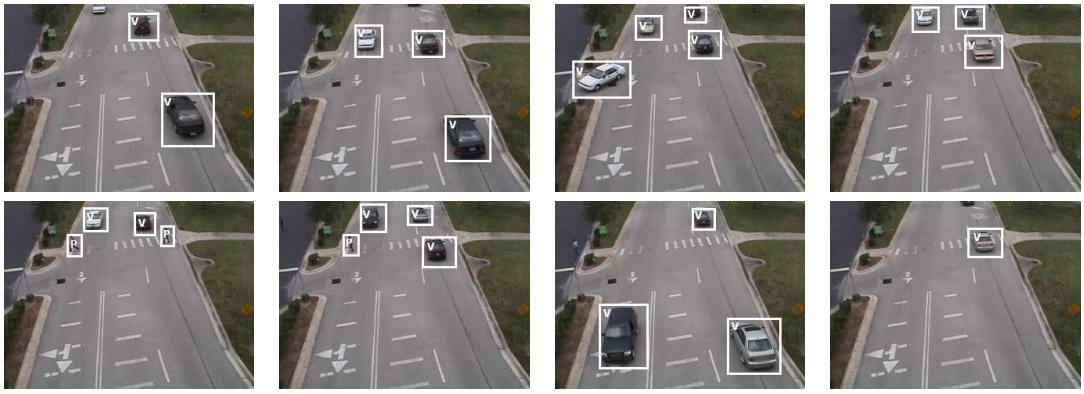

Figure 3. Some classification results from sequence 1.

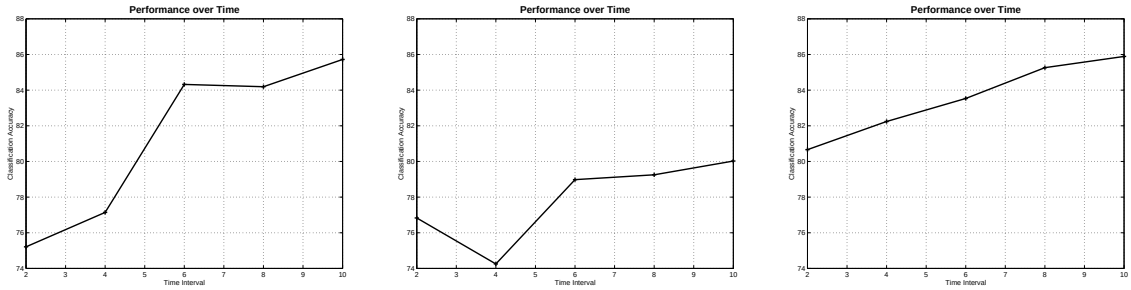

Figure 4. Change in performance with increase in time for sequence 1,2 and 3 respectively. The performance was measured over two minute intervals. Approximately 150 to 200 possible detections of vehicles or pedestrians were made in each time interval.

quence into two sets. In the first set the classification results were obtained using the multi-class Adaboost.M1 classifier without co-training. Then the other set was run with the co-trainable classifier, stopping when a pre-determined number of labeled examples had updated the classifier parameters. Once the updated parameters were obtained, the boosting algorithm was re-run on the first sequence with the classifier parameters frozen and the change in performance was measured. The improvement in the performance of the algorithm in the first setup is shown in Figure 6. The horizontal axis shows the number of examples obtained through co-training from the second sequence, and the vertical axis shows the detection rates on the test sequence. The detection rates improve significantly even with a small number of new training examples. Since the automatically labeled training examples are from the specific scene on which the classifier is being evaluated on, only a few co-trained examples are sufficient to increase the detection accuracy. Some detection results are shown in Figures 3 and 5.

Upon analysis of the examples selected for co-training by the base classifiers we found out that approximately 98% of these were correctly labeled. The small number of misclassification were caused mainly by occlusion. One important point in the use of examples obtained through co-training for update of classifier parameters is that, if the examples are misaligned, or the target object is only partially visible, then updating the classifier parameters with that example can lower the classification accuracy. We reduce the likelihood of such a scenario by forcing the detected region to be within the foreground regions as determined by the background modeling algorithm. Moreover we only select those examples that are at peaks of the (boosted) classifier scoring function, as suggested in [5].

Another problem that might arise during co-training is that if examples of one class are observed in much greater numbers than other classes. Updating the classifier parameters by training through examples of one class only can bias the classifier. This problem always occurs in a scenario when the background has to be distinguished from the object by the classifier. In this case, the examples of the background class outnumber by far the examples of the object class. Since, we are removing most of the background region by background subtraction, this scenario is less likely to occur. To avoid this problem completely, if examples of one class are being confidently labeled in much greater number than others, then one can store the examples and

Figure 5. Moving object classification results from sequence 2.

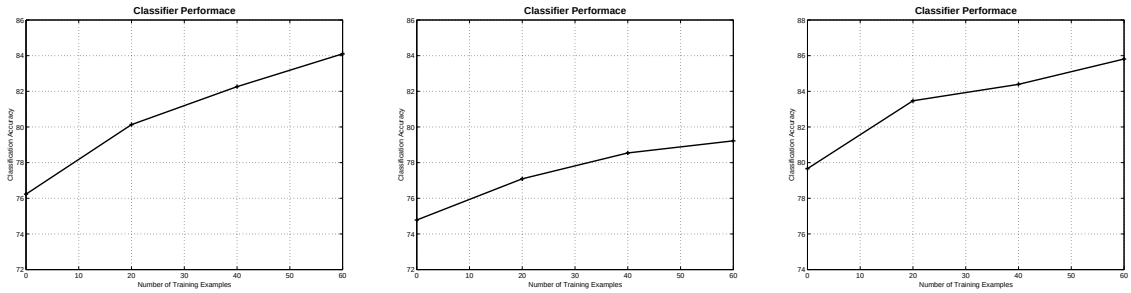

Figure 6. Performance vs. the number of co-trained examples, for sequences 1,2 and 3 respectively. The graphs for each sequence show the improvement in performance with the increase in the use of examples labeled by the co-training method. Note that, relatively few examples are required for improving the detection rates since these examples are from the same scene in which the classifier is being evaluated. The classification accuracy was relatively low for sequence 2 since there was persistent occlusion between vehicles.

sample them in numbers comparable to other classes, rather than using all of them for training.

5. Concluding Remarks

In this paper, we presented a unified boosting based framework for online training and classification of objects. The examples that were confidently labeled by a small subset of base classifiers were used to update both the boosting coefficients and the base classifiers. We have demonstrated that a classifier's performance can be significantly improved just by using a small numbers of examples from the specific scenario in which the classifier is employed. This is because the variation in the poses of objects, backgrounds and illumination conditions in a specific scene is far less than the possible variation in all possible detection scenarios. The use of co-training in an online classification framework allows us to focus on the specific subset of poses and backgrounds likely to be viewed in each scenario.

Acknowledgements

This material is based upon work funded in part by the U. S. Government. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Government.

References

[1] A. Blum and T. Mitchell. "Combining labeled and unlabeled data with co-training". In *11th Annual Conference on Computational Learning Theory*, 1998.

[2] M. Collins and Y. Singer. "Unsupervised models for named entity classification". In "*Empirical Methods in Natural Language Processing"*, 99.

[3] Y. Freund and R. Schapire. "Experiments with a new boosting algorithm". In *International Conference on Machine Learning*, 1996.

[4] O. Javed, K. Shafique, and M. Shah. "A hierarchical approach to robust background subtraction using color and gradient information". In *Workshop on Motion and Video Computing*, pages 22–27, 2002.

[5] A. Levin, P. Viola, and Y. Freund. "Unsupervised improvement of visual detectors using co-training". In *ICCV*, 2003.

[6] N. Oza. "Online ensemble learning". In *Ph.D. dissertation*, 2002.

[7] C. Papageorgiou and T. Poggio. "Trainable pedestrian detections". In *ICIP*, 1999.

[8] D. Pierce and C. Cardie. "Limitations of co-training for natural language learning from large datasets". In *Conference on Empirical Methods in Natural Language Processing*, 2001.

[9] H. Schneiderman and T. Kanade. "A statistical method for 3d object detection applied to faces and cars". In *CVPR*, 2000.

[10] P. Viola, M. Jones, and D. Snow. "Detecting pedestrians using patterns of motion and appearance". In *ICCV*, 2003.

[11] D. Zhang, S. Z. Li, and D. Perez. "Real-time face detection using boosting in hierarchical feature spaces". In *Int. Conf. on Image Processing*, 2004.