Unsupervised Improvement of Visual Detectors using Co-Training

Anat Levin*
School of CS and Eng.
The Hebrew University
91904 Jerusalem, Israel
alevin@cs.huji.ac.il

Paul Viola
Microsoft Research
One Microsoft Way
Redmond, WA 98052
viola@microsoft.com

Yoav Freund
Computer Science Dept.
Columbia University
New York, NY 10027
freund@cs.columbia.edu

Abstract

One significant challenge in the construction of visual detection systems is the acquisition of sufficient labeled data. This paper describes a new technique for training visual detectors which requires only a small quantity of labeled data, and then uses unlabeled data to improve performance over time. Unsupervised improvement is based on the co-training framework of Blum and Mitchell, in which *two* disparate classifiers are trained simultaneously. Unlabeled examples which are confidently labeled by one classifier are added, with labels, to the training set of the other classifier. Experiments are presented on the realistic task of automobile detection in roadway surveillance video. In this application, co-training reduces the false positive rate by a factor of 2 to 11 from the classifier trained with labeled data alone.

1 Introduction

There are now a number of practical solutions for the problem of visual detection [14, 11, 9, 13, 17]. While the primary area of application is face detection, it has been shown that these approaches are general and can be applied to other objects such as pedestrians, face profiles, and automobiles [9, 13]. In the context of this success, it might be argued that the construction of new types of detectors is a straightforward process: select a detection technique, acquire a large training set, and train the detector. Of course many detection tasks are simply beyond the capabilities of current detection techniques. Yet even for the "solvable" tasks, the cost of data acquisition may be large enough to preclude a practical deployment.

We call this scenario "High Initial Expense". It arises because all the above techniques require a very large set of labeled training data. Typically several thousand scaled and aligned positive examples are required. The cost of compiling this positive data is high, since each example must be located by hand. In addition as many as 10^9 negative examples are also required, usually a collection of several thousand images which do not contain positive instances. This large number of negative examples ensures that the false positive rate is very low, perhaps 10^{-6}.

Another related scenario is that of "Narrow Application". Using the example of face detection because of its long history, achieving detection rates higher than 95% on realistic images has proven very difficult. Part of the difficulty clearly lies in the complexity of the appearance of faces. But another part of the difficulty arises because of the very wide variety of background images which are encountered, including indoor locations such as offices, living rooms, elevator lobbies, and conference rooms, and outdoor locations such as fields, mountains, and trees. There are many foreseeable applications of face detection which involve a fixed camera, or a camera with a limited area of application. For these cameras the range of background images is very limited. A face detector which has been trained for "broad application" will expend representational capacity to reject false positives which will never be encountered. Conversely, a face detector which is trained for "narrow application" in a particular location can achieve much lower false positives rates and higher detection rates. The cost of this approach, is that one must acquire a different training set, and train a different classifier, for each location.

The difficulty shared in both scenarios is the high cost of acquiring a large set of labeled examples. Of course, gathering a large number of *unlabeled* examples in most applications has much lower cost, as it requires no human intervention. The question is whether unlabeled examples are of any use when training a visual detector.

In this paper *co-training* is used to automatically improve visual detector for cars in traffic surveillance video [1]. Initially a small quantity of hand labeled data is used to train a *pair* of car detectors (using the approach of Viola and Jones[16]). Co-training then generates additional labeled training example from a large number of unlabeled images. Experiments demonstrate that co-training can generate an accurate car detector using a significantly smaller

*Research performed while all authors were at Mitsubishi Electric Research Labs, Cambridge, MA. Their support is gratefully acknowledged.

number of labels than would be required for the same algorithm when co-training is not used.

2 Learning from Unlabeled Data

The use of both labeled and unlabeled data for practical problems was popularized by Nigam et. al. in the area of information retrieval [8]. They use EM to infer the missing labels of the unlabeled data much in the same way that EM is typically used to infer missing cluster labels. During learning, EM assigns strong labels to those unlabeled examples which are unambiguous. These new examples sharpen the class density estimates, which then allows for the labeling of additional unlabeled examples.

The basic assumption underlying the success of EM for this task (as well as the more recent techniques [15, 7, 6]) is that the distribution of unlabeled data respects the class boundaries of the labeled data. Technical details vary, but the bottom line is that the density of unlabeled example must be low near the classification boundary. This makes good sense for two class problems where the classes are Gaussian (and in a significant number of other practical situations). This assumption often does not hold for detection tasks, where the density of the detected class is lost amid the thousands of other classes.

Support Vector Machines [3] and Adaboost [4, 5] are purely discriminative techniques for pattern recognition which have had a significant impact on applications. Neither method attempts to estimate the density of the classes. Instead, both methods use the notion of "classification margin" and attempt to maximize the margin of all (or most) training examples. The result is improved generalization performance from fewer examples.

Applied to discriminative classifiers, the direct analog of Nigam et al's approach is to assign labels to those unlabeled examples which have a large margin (and are therefore unambiguous). This is *not* an effective technique, since labeled examples with large margin are not informative and have little effect on the final classifier. For discriminative classifiers one must find unlabeled examples which can be unambiguously labeled *AND* have a negative (or small) margin.

Co-training was proposed by Blum and Mitchell[1] as a method for training a *pair* of learning algorithms. The basic assumption is that the two learning algorithms use two different "views" of the data. For example, it is not hard to believe that one can discriminate between apples and bananas using either features of their shape *or* features of their color. Since the margins assigned by the classifiers are not directly related, there may exist a set of examples with high margin based on shape and small or negative margin based on color. The key property is that some examples which would have been *confidently* labeled using one classifier would be *misclassified* by the other classifier. The classifiers can therefore train each other, by providing additional *informative* labeled examples. See Figure 2 for experimental evidence that such informative examples do exist.

Given two "views" of the data, one might be tempted to avoid training altogether and simply combine the views in order to improve the classification performance. Why then does co-training operate on the views separately, since it reduces classification performance? Co-training is a *training process* not a classification process. After co-training the final classifiers, which are trained on labeled and unlabeled data, are significantly improved. These improved classifiers are easily combined in order to maximize classification performance.

In fact, Blum and Mitchell prove under a set of formal assumptions, that co-training finds a very accurate rule from a very small quantity of labeled data. This error rate is far smaller than what would be achieved by simply combining the initial classifiers. The required assumptions include: a reasonable learning algorithm, an underlying distribution which satisfies "conditional independence", and an initial weak classification rule. The main drawback of their theorem is the assumption of conditional independence, which requires the two feature sets be statistically independent. In most real world cases, this assumption is not likely to hold.

Nevertheless, co-training based methods for real world problems have been developed and used successfully by several groups, especially in the context of text processing[2, 10]. In general, the approach used was to add a new term to the training cost function penalizes the number of disagreements that the two classifiers have on the unlabeled examples.

While this seems like a reasonable approach, it overlooks one important aspect of the co-training idea, which is that each learner labels only those unlabeled examples on which it can make a *confident* prediction. In this paper we suggest a different co-training algorithm, which is based on the known relationship between prediction confidence and prediction margins[12].

2.1 Co-training Using Confidence Rated Predictions

While the general notion of co-training was defined by Blum and Mitchell, there are many potential algorithmic instantiations. We propose a new algorithm for co-training which is explicitly applicable to margin based classifiers. Given two classifiers trained on a small data set, estimate margin thresholds above which (or below which) all training examples are correctly labeled. Thresholds can be estimated from the training set, or a validation set. Using these thresholds assign labels to unlabeled examples, and then add these examples to the training set and re-train the classifier. New thresholds are then estimated. This process can be repeated many times.

In the context of Adaboost we can analyze the co-training process in the following way. Let us denote the outputs from the weak classifiers by the vector $\vec{h} \in [-1,+1]^n$ and the weights associated with these classifiers by $\vec{\alpha} \in [-1,+1]^n$, $\sum_{j=1}^n \vec{\alpha}_j = 1$. The large margins assumption is that there is some real number $\theta > 0$ such that the probability that $\vec{h} \cdot \vec{\alpha} > \theta$ is significant and the conditional probability that $\vec{h}$ corresponds to a detection given that $\vec{h} \cdot \vec{\alpha} > \theta$ is close to one.

Can we estimate this conditional probability reliably from our training set? On its face, there seems to be no reason to believe that this is possible. After all, the weights $\vec{\alpha}$ depend on the training set and were chosen to maximize the margins of the training examples. However, Schapire et al [12] have shown, both experimentally and theoretically, that large margins on the training set imply correct classification on test data *even when the dimensionality n is extremely high*. Specifically, Theorem 2 in [12] shows that for any convex combination f of weak rules from a class with VC dimension d, and for any $\theta > 0$, the following inequality holds with probability $1-\delta$ over the random choice of a training set of size m:

$$P\left[f(\vec{h}_i) \leq 0\right] \leq \frac{1}{m}\sum_{i=1}^{m} \mathbf{1}\left[f(\vec{h}_i) \leq \theta\right] + O\left(\frac{1}{\sqrt{m}}\left(\frac{d\log^2(m/d)}{\theta^2} + \log(1/\delta)\right)^{1/2}\right).$$

where y is the $\{+1,-1\}$ label, $f(\vec{h}_i)$ is the convex combination of weak classifiers for the i-th example. The first sum on the right hand side of the equation corresponds to the fraction of the training set examples with margin smaller than θ. The left hand side is proportion of *testing* examples which are in error. The most important aspect of this theorem is that the bound does not depend on n the number of weak functions that are combined in the convex combination f.

From this we can conclude that there exists θ, estimated on the training or validation set, for which the risk of misclassification on test data is very low. We also rely on the fact that the margins of the two classifiers are only weakly related, and that the classifiers are not perfect. As a result, unlabeled data can be used to generate new informative training examples for which the predicted label is highly accurate.

3 Co-training for Visual Detection

There are a number of serious issues that arise in the domain of visual detection which must be addressed. One difficultly arises due to the highly unequal class probabilities. A second is the alignment of automatically labeled positive examples.

For the experiments in this paper we will co-train two detectors for automobiles as seen from traffic surveillance video cameras. These cameras are typically used to gauge the traffic volume and delays in large urban areas. Many of these cameras are setup as simple web-cams for the benefit of commuters (see Figure 1). Other, more specialized cameras, are used to count cars which pass a particular point on the road. These cameras are used to replace "traffic loops", electromagnetic systems which are expensive to install and calibrate. The traffic loop cameras are quite specialized and must be placed in very particular locations and carefully calibrated. One application of our system would be to replace traffic-loops and traffic-loop cameras with less expensive web cam cameras.

In this application one classifier detects cars in the original grey level images. The second classifier detects cars in images where the background has been subtracted (called BackSub in the rest of the paper). These classifier are well suited to the available data, since the images are monocular and grey level. Nevertheless, the input images are somewhat related, and as a result classifiers are not quite ideal for co-training. We must emphasize that while "conditional independence" is a sufficient condition for co-training to succeed, it is widely believed that it never holds in practice. This paper demonstrates that even two closely related classifiers can be co-trained effectively.

For our experiments 50 image patches containing positive examples and 6 validation images are used for the initial training. While this a very small quantity of data, the resulting classifier is far better than random. In addition 22,000 entirely unlabeled images are made available for training. After co-training, error rates are reduced by a factor of 2 to 11 across the entire ROC (receiver operating characteristic) curve. The final classifier is quite effective.

4 Detection Framework

The detectors used in the co-training framework are based on the work of Viola and Jones.

As is typical for detection, the input image is scanned across location and scale. At each location an independent decision is made regarding the presence of the target object. In a 320x240 image there are over 50,000 independent locations.

A collection of features are used to classify the training set. These features are selected using the sequential logistic

Figure 1: Example images used to test and train the car detection system. On the left are the original images. On the right are background subtracted images.

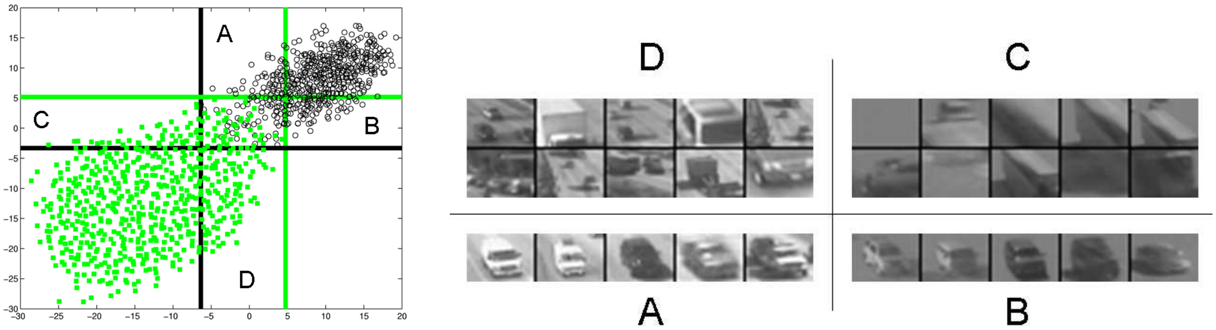


Figure 2: Left: A scatter plot of the joint distribution of margins for the two classifiers. These results are shown on test data, and therefore represents the distribution on unlabeled data (positive examples are circles, negative are grey/green). For each classifier two threshold are also shown, the threshold above which no negative is found, θ_p, and the threshold below which no positive is found, θ_n. The regions labeled A,B,C, and D contain informative examples. Right: Particular examples taken from A, B, C, or D; images which are mislabeled by one classifier (or have small margin) which are confidently labeled by the other classifier. E.G. Set B contains images confidently labeled positive by the Grey classifier but are misclassified by the BackSub classifier. These examples are added to the training set of the BackSub classifier during co-training.

regression algorithm of Collins et. al.(which we will call LogAdaBoost in this paper). In each round the feature selected is that with the lowest weighted error. Each feature is a simple linear function made up of rectangular sums followed by a threshold. In the final classifier, the selected feature is assigned a weight based on its performance on the current task. As in all variants of AdaBoost, examples are also assigned a weight. In subsequent rounds incorrectly labeled examples are given a higher weight while correctly labeled examples are given a lower weight.

In order to reduce the false positive rate while preserving efficiency, classification is divided into a cascade of classifiers. The early classifiers are constrained to use few features (and are therefore efficient) while achieving a very high detection rate. Constraints on the later classifiers are relaxed: they contain more features and have a lower detection rate. Later cascade stages are trained only on the true and false positives of earlier stages.

LogAdaBoost is used to train each stage in the cascade to achieve low error on a training set. Due to the asymmetric structure of the detection cascade, each stage in the cascade must achieve a very low false negative rate. The false negative rate of the trained classifier is adjusted, post hoc, using a set of validation images in which positives have been identified. These images are scanned and the threshold is set so that the required detection rate is achieved on these validation positives.

In order to train a full cascade to achieve very low false positive rates, a large number of examples are required, both positive and negative. The number of required negative examples is especially large. After 5 stages the false positive rate is often well below 1%. Therefore over 99% of the negative data is rejected and is unavailable for training subsequent stages.

5 Experiments and Algorithms

Data was acquired from a Washington State Department of Transit web site. The cameras selected provide 15 second video clips once every 5 minutes. Data from a total of 8 cameras was used for experiments. The cameras were similar, in that they were placed by the same authority. They did however vary in height and angle to the roadway. Data was acquired over a period of three weeks and randomly sampled.

Two types of classifiers were constructed: a grey image classifier (Grey) and a background difference (BackSub) classifier. The Grey classifier uses the grey scale images directly for detection. The input to the BackSub classifier is the difference between the video images and the average background computed from each video clip. Otherwise the feature set and training algorithms were identical for both classifiers.

The labeled training data made up a tiny subset of the total dataset. The training data includes images from 3 of the cameras in which 50 cars were identified. In each case a box was drawn around the car which contained a small percentage of the background and had an aspect ratio of 1.4. For training the car images where cropped and scaled to 20x28 pixels. The training data was limited to those car images which were equal to or larger than 20x28 pixels. The same 50 labeled cars were used to train both the image classifier and the background difference classifier.

In addition to the labeled positive data, a set of six additional images were used as a validation set to adjust classifier thresholds. The validation set contains images in which every car is located and labeled. In practice images are selected so that there are as many positives as possible so that both the false positive and false negative rate can be estimated. If there were unlabeled positive examples in these images, it is likely that these positives would appear as falsely labeled negative data.

Using this training and validation data, two cascaded detectors are constructed, one for the grey images and one for the background subtracted images. The input to the cascade construction algorithm is the target detection and false positive rates, a training set, and a validation set. The cascade is built incrementally by adding features and classifier stages until the detection and false positive rate targets are achieved. The target detection rate is 100% (to ensure that no positive example is lost) and the target false positive rate was 0.2% (the same as the true positive rate). A 5 stage cascade was learned with a total of 20 features. At this point the co-training process is begun.

5.1 Co-training the Detectors

The first step is to retrain the final stage for the cascade so that it has a larger number of features (in our experiments 30 features). This expanded stage provides a more accurate estimate for the confidence (or margin) of new examples. By comparison, a 4 feature stage would provide a smaller range of confidence values.

Classification scores, a signed measure of confidence computed by this final stage, are used to label a set of 22,000 unlabeled images (containing billions of sub windows). Each patch in these images is first passed through the cascade. If it labeled as potentially positive by the first 5 stages, then the score computed by the final stage is recorded.

For this final stage two thresholds are used to collect and label unlabeled data. These thresholds are set using the validation set. The positive threshold $\theta_p$is the maximum score achieved by the negative patches in the validation set. Any example which falls above $\theta_p$is very likely to be positive. Similarly $\theta_n$is the minimum score achieved by the positive patches in the validation set. Note that $\theta_p$and $\theta_n$are estimated separately for the Grey and BackSub classifiers. Since most data is negative, we can afford to be extremely conservative. The conservative negative threshold is set to 1.3 times θ_n.

New examples are labeled and sampled using $\theta_p$and θ_n. These are added to the training set which includes the original labeled data. Using this new training set, AdaBoost is used to add three new features the classifier. The unlabeled data is then resampled again. After 12 rounds of resampling there are $66 = 30 + 3 \times 12$ features.

Note that construction of the initial cascaded detectors act to solve two critical problems facing the co-training process: asymmetry and efficiency. Recall that in each image there are at most a few dozen positive examples and as many as 30,000 negative examples. Co-training predicts labels for examples where one classifier is “confident” – unlikely to be in error. In order to have a small percentage of errors on positive labels, the false positive rate must be less than 1 in 10,000. This can be difficult to achieve. After 5 cascade stages, the problem is much closer to symmetric, with approximately one positive for every 10 negatives. The cascade also provides a very significant boost in performance, since only 1 in 1000 examples are accepted by the early cascade stages.

5.2 Sampling Unlabeled Examples

Due to the strong asymmetry in the distribution of positive and negative examples, a different procedure is used to sample from the different classes.

The main challenge for negative examples is the sheer number of candidates. There are a huge number of confidently labeled negative patches available after scanning 22,000 images, more than can be accommodated by the learning algorithm. As a result, a justifiable technique of selecting a good subset out of them is required.

Recall that in each iteration of the LogAdaBoost algorithm each example x_i is assigned a weight w_i. A new feature h_t is then add to the classifier, such that the weighted sum over the training examples $\sum_i y_i w_i h_t(x_i)$, is maximized.

- **Initial setup**
  - Train two detectors
    - Detector One: operates on grey level images
    - Detector Two: operates on difference images
    - Each detector has 5 classifier stages each containing 4 features.
    - Detection rate on training set is 100%; false positive is 0.2%.
  - Final stage is retrained to contain 30 features, which produces more reliable classification scores). Note: the scores assigned by this final stage are used to select additional examples for co-training.
  - Two thresholds are computed $\theta_p$and θ_n.
    - θ_p= max score achieved by a negative patch from the validation set
    - θ_n= min score achieved by a positive patch from the validation set
- **Co-training process (run for 12 rounds)**
  - 22,000 unlabeled images are scanned.
  - In each range of co-training patches are sampled from the 22,000 unlabeled images
    - Image patches labeled positive by the 5 stage cascade are examined
    - Positive patches are extracted if score is greater than θ_p
      - Pick local maxima in the score function (to improve alignment)
    - Negative patches have score less than θ_n
      - Since there are so many negative examples, examples are selected at random based on AdaBoost weight
  - The final stage of the classifier is augmented with 3 additional features using this new training data.

Figure 3: A concrete description of the co-training process. Note there are very few parameters in this process. the structure of the cascade is determined automatically during training (based on a target false positive equal to the true positive rate). Thresholds are set automatically as well.

Though one could sample uniformly from the set of confident negatives, this would ignore one critical piece of information, the margin of the example. If co-training is to work well it relies on the assumption that some examples that are confidently classified as negative by one classifier are not confidently labeled by the other. These examples are highly informative for the learning process.

A more principled sub sampling procedure is to use the *importance sampling* approach, and randomly select negative data using a probability distribution related to the example weight w_i, and then assign the sampled examples a constant weight. That is, if a subset of training examples $\{x_k\}_{k=1}^{N}$ was sampled based on the LogAdaBost weight, then the right approximation to the feature score is $\sum_i y_i h_t(x_i)$. According to the importance sampling principal, the achieved approximation will be significantly better then the one achieved by uniformly sampling, and weighting the sampled examples.

The naive way to sample N examples out of the training set will require scanning the confidently labeled negatives N times. Since in each stage we are to sample few thousands out of approximately billion examples, the above approach will result in an extremely inefficient algorithm. Tentatively, a reasonable approximation can be achieved with only one pass over the data. To see this consider scanning the data once, and for each example flipping N coins with head probability αw_i, $\alpha = 1/\sum_i w_i$. The expected number of selected examples resulting from such a process is N. Moreover, the expected number of times each example comes out in the sample is $N\alpha w_i$. This leads us to the following algorithm: scan the data once. For each example in the data- if $N\alpha w_i < 1$ select the example with probability $N\alpha w_i$ and assign it a weight 1. If $N\alpha w_i \geq 1$, select the example with probability 1, and assign it the weight $N\alpha w_i$.

Note, however, that since the weights are exponentially related to the margin, some examples have very high weights and the number of *different* example achieved by the above procedure is likely to be significantly lower then N. Since the accuracy of the selected feature is strongly related to the number of different examples used for the evaluation, we wouldn't like the resulting number of samples to be too small. In order to achieve N different examples there is a need to scale the examples weights w_i by a larger α. The correct solution is to solve $\sum_i min(\alpha N w_i, 1) = N$, which is difficult when there are very many examples. This equation can be approximately solved using a histogram of the weights, which records the number of examples within different ranges of weights.

5.3 Sampling Positive Data

Positive data must also be sampled carefully, but since positive data is very rare, it is not necessary to reduce the total number of examples using sub-sampling. The key challenge is alignment. In many detection tasks significant effort goes into establishing a good alignment between the labeled positive examples: the car images are scaled to the same size, and translated so that visible features appear in a consistent location relative to the detection window.

The alignment of examples sampled based on a high score is problematic. It is frequently the case that for any given positive example many overlapping subwindows are assigned high score. This is a fundamental property of all scanning detectors, and it has been observed for many types of detection tasks. Typically during detection, these set of overlapping detected sub-windows are merged into a single detection. During co-training great care must be taken, since injecting positive examples at different scales and translations can confuse the training process, and reduce performance.

The solution is to select only examples which are at the peaks of the scoring function. The weight assigned to each peak is the sum of the weights above the selection threshold and are nearby the peak. See Figure 2 for automatically selected positive examples.

5.4 Evaluation

Testing was performed on a separate set of hand labeled examples that weren't used anywhere in the training process. This set included 90 images containing 980 positive examples. Since we are measuring the incremental improvement of the cascaded detectors due to co-training, the final co-trained stage is only tested on those positives and negatives which make it through the first 4 layers of the cascade. In this case 7,000 negative examples remain for the background subtracted cascade, and 10,000 for the grey image classifier.

Figure 4 presents the ROC curves that were computed using this testing data. Note that evaluating the ratio between the error rates of the original classifier and the co-trained classifier, shows an improvement by a factor of 2 to 3 for the standard gray levels classifier, and a factor of 2 to 11 for the background subtracted classifier.

Figure 5 shows detection results, evaluated on some unlabeled images.

6 Conclusions

We have shown that co-training can be used to significantly improve detectors using unlabeled data. These detectors are provided with less than 10% of the labeled data used

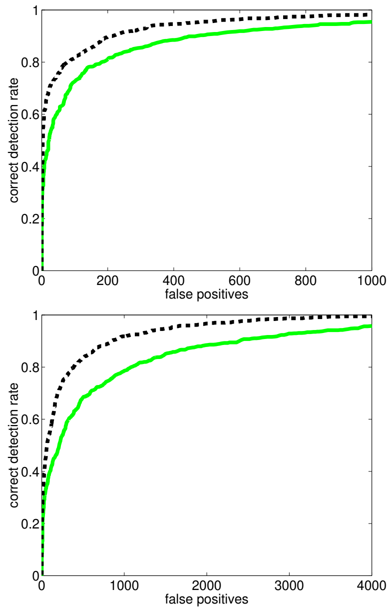


Figure 4: ROC curves. Green/Grey line: the original classifier. Black dashed: the co-trained classifier. TOP the GREY classifier. BOTTOM: the BackSub classifier.

to train other published visual detectors. After co-training the detection rates are quite good and the system is highly functional.

One application of co-training is to reduce the cost of constructing visual detectors. Another application may be to produce detectors which are finely tuned to the specifics of a particular problem.

As demonstrated in this paper, co-training automatically improves a pair of weak detectors with no additional labeled data. In principle one could deploy a large number of generic and therefore weak detection systems. Each system could then use co-training to automatically fine tune performance to achieve much higher detection rates. To achieve this improvement each system would leverage the unique characteristics of the deployed environment.

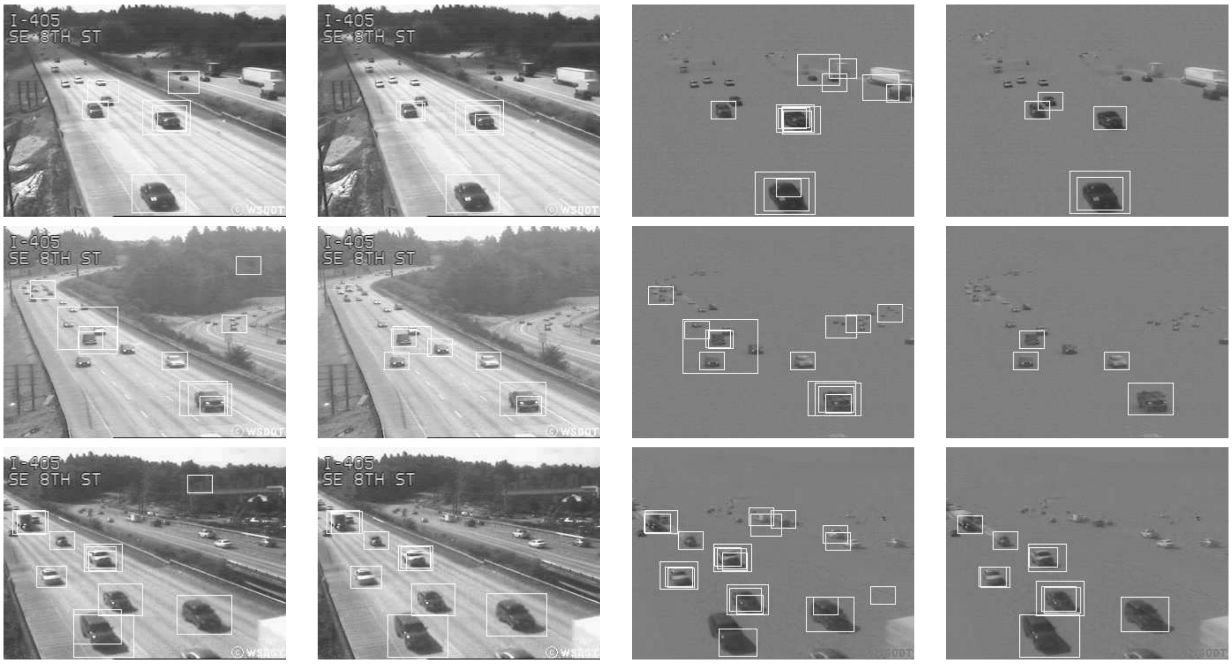
References

[1] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT: Proceedings of the Workshop on Computational Learning Theory, Morgan Kaufmann Publishers*, 1998.

[2] M. Collins and Y. Singer. Unsupervised models for named entity classification, 1999.

[3] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, September 1995.

[4] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting.



(a) (b) (c) (d)

Figure 5: Detection results. (a)- The gray level classifier before co-training. (b)- The gray level classifier after co-training. (c)- The background subtracted classifier before co-training. (d)- The background subtracted classifier after co-training.

Journal of Computer and System Sciences, 55(1):119–139, Aug. 1997.

[5] Y. Freund and R. E. Schapire. A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780, Sept. 1999. Appearing in Japanese, translation by Naoki Abe.

[6] T. Hofmann and J. Puzicha. Unsupervised learning from dyadic data. Technical Report TR-98-042, Berkeley, CA, 1998.

[7] T. Joachims. Transductive inference for text classification using support vector machines. In I. Bratko and S. Dzeroski, editors, *Proceedings of ICML-99, 16th International Conference on Machine Learning*, pages 200–209, Bled, SL, 1999. Morgan Kaufmann Publishers, San Francisco, US.

[8] K. Nigam, A. K. McCallum, S. Thrun, and T. M. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134, 2000.

[9] C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *International Conference on Computer Vision*, 1998.

[10] S.-B. Park and B.-T. Zhang. Document filtering boosted by unlabeled data. In *In Proceedings of IEEE International Symposium on Industrial Electronics (ISIE 2001)*, 2001.

[11] H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. In *IEEE Patt. Anal. Mach. Intell.*, volume 20, pages 22–38, 1998.

[12] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651–1686, October 1998.

[13] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. In *Computer Vision and Pattern Recognition*, 2000.

[14] K. Sung and T. Poggio. Example-based learning for view-based face detection. In *IEEE Patt. Anal. Mach. Intell.*, volume 20, pages 39–51, 1998.

[15] M. Szummer and T. Jaakkola. Kernel expansions with unlabeled examples. In *Advances in Neural Information Processing Systems*, volume 13, pages 626–632, 2001.

[16] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[17] P. Viola and M. J. Jones. Robust real-time object detection. In *Proc. of IEEE Workshop on Statistical and Computational Theories of Vision*, 2001.