FlowBoost – Appearance Learning from Sparsely Annotated Video

Karim Ali[1,2] David Hasler[2] François Fleuret[3]

[1] École Polytechnique Fédérale de Lausanne (EPFL), CVLAB, Switzerland

[2] Swiss Center for Electronic and Microtechnology (CSEM), Switzerland

[3] Idiap Research Institute and École Polytechnique Fédérale de Lausanne (EPFL), Switzerland

karim.ali@epfl.ch, david.hasler@csem.ch, francois.fleuret@idiap.ch

Abstract

We propose a new learning method which exploits temporal consistency to successfully learn a complex appearance model from a sparsely labeled training video. Our approach consists in iteratively improving an appearance-based model built with a Boosting procedure, and the reconstruction of trajectories corresponding to the motion of multiple targets. We demonstrate the efficiency of our procedure on pedestrian detection in videos and cell detection in microscopy image sequences. In both cases, our method is demonstrated to reduce the labeling requirement by one to two orders of magnitude. We show that in some instances, our method trained with sparse labels on a video sequence is able to outperform a standard learning procedure trained with the fully labeled sequence.

1. Introduction

Many classes of objects can now be successfully detected with machine learning techniques. Faces, cars, pedestrians and hands, have all been detected with low error rates by learning their appearance in a highly generic manner from extensive training sets. These recent advances have enabled the use of reliable object detection components in real systems, such as automatic face focusing functions on digital cameras. One key drawback of these methods, and the issue addressed here, is the prohibitive requirement that training sets contain thousands of manually annotated examples.

We propose to reduce the requirement for such an extensive labeling by exploiting the temporal consistency occurring in a training video. Our method, FlowBoost, starts from a sparse labeling of the video, and alternates the training of an appearance-based detector with and a convex, multi-target, time-based regularization. The latter relabels the full training video in a manner that is both consistent with the response of the current detector, and in accordance with physical constraints on target motions.

The performance of this approach is evaluated on pedestrian detection in a surveillance camera setting, and on cell detection in microscopy data. Our experiments show that our method allows for a reduction in the number of training frames by a factor of 15 to 60. This comes with virtually no loss in performance when compared to a standard learning procedure trained on a fully labeled sequence. In fact, in some cases, gains in performance are observed.

2. Related Work

The use of labeled and unlabeled data to learn a classification model falls under the broad category of semi-supervised learning, initially introduced in [12]. Given the prominence of machine learning techniques, the cost associated with producing annotated data and the abundance of unlabeled data, a growing number of works have addressed this problem.

Broadly speaking, one can view the semi-supervised learning problem as a constrained instance of unsupervised learning, where the additional constraints come in the form of labeled data. This approach was taken in [11] where a generative classification model for document classification is used along with Expectation-Maximization. The algorithm first trains a naive Bayes classifier using the labeled data and probabilistically labels the unlabeled data. It then retrains a classifier using the most confident labels and the procedure is iterated.

A number of other works rely on the assumption that the data lies on a low-dimensional manifold. As a consequence, the data is represented as a graph where vertices represent samples and edges-weights, pairwise similarity. Various methods which propagate labels to the entire graph

This work has been supported in part by the Swiss National Science Foundation under project 200021-117997, by the Fond Québécois de recherche sur la nature et les technologies, and by the European Community's Seventh Framework Programme FP7 - Challenge 2 - Cognitive Systems, Interaction, Robotics - under grant agreement No 247022 - MASH.

until convergence are proposed in [15, 13, 3, 7]. These algorithms are naturally transductive and hence of limited applicability to the object detection setting where an inductive classification rule is desired.

Still other works rely on the assumption that the decision boundary must lie in a low-density region. In [14], a method to maximize the margin of both labeled and unlabeled sample, termed Transductive Support Vector Machine (TSVM), is introduced. However, the corresponding problem is non-convex and therefore difficult to optimize. One approach presented in [6] starts with an initial SVM solution obtained from the labeled data alone. Next, the unlabeled data points are labeled by SVM predictions, their weights increased and the SVM retrained. An alternative approach derived from a bayesian discriminative framework and involving a so-called Null Category Noise Model with Gaussian Processes is presented in [10].

In this paper, we exploit temporal consistency in videos to assist in the labeling of unlabeled data and iteratively improve an appearance based classifier. Given a training video, we begin by annotating a small subset of frames while the remaining frames are not annotated. This limited initial training data is used to train an appearance based classifier which is subsequently evaluated on the entire video sequence. Admissible trajectories of targets are retained as positive samples while the remaining data is retained as negative samples and the process is iterated.

Thus, our unlabeled data is in fact highly structured. In particular, once all trajectories are correctly identified, all remaining samples certainly belong to the negative class. From this perspective, our work is related to the tracking approach of [8, 9] even though the goals differ significantly. In [8, 9], a system is built with three components: a tracker, a detector and a model. The tracker and detector are always run in parallel and the hypothesis from either that minimizes the distance to the model is kept. The model itself is updated by growing and pruning events which generate positive and negative samples to update the model. The system is online, real-time and shows promising results.

One important difference with our work is that [8, 9] relies on the very stringent assumption that no more than one target is present in any given frame. Thus, once the hypothesis with maximal confidence is identified in a frame, all remaining samples are considered negative. By contrast, we rely on a global offline optimization, that is integrated into the AdaBoost algorithm while allowing for an unknown and unconstrained number of targets as is next explained.

3. Methodology

We are interested in detection by classification. That is, given an image and a feature vector $x_s \in \mathbb{R}^D$ for every location s in the image, we want to build a classifier

$$\varphi : \mathbb{R}^D \to \mathbb{R}$$

such that the set of locations

$$\{s : \varphi(x_s) \geq 0\}$$

forms a good prediction of the set of locations where the targets (faces, cars, etc.) are actually visible. The standard technique consists in building a hand-labeled training set

$$(x_n, y_n) \in \mathbb{R}^D \times \{-1, 1\}$$

and training a classifier φ with a low empirical error rate.

As described in the introduction, we want to avoid the requirement for a large training set by using a training video, and by going back-and-forth between the optimization of an appearance-based classifier φ, and the optimization of the labels of non-labeled frames. The latter can be reformulated as the optimization of a Boolean flow f in a graph under physically plausible motion constraints.

Let $\mathcal{T} = \{1, \ldots, T\}$ be the set of time steps, and $\mathcal{S} = \{1, \ldots, W\} \times \{1, \ldots, H\}$ the set of locations, where W and H are respectively the width and heights of the video frames[1]. Let

$$\mathbf{x} \in \left(\mathbb{R}^D\right)^{\mathcal{T} \times \mathcal{S}},$$

be the feature vectors of dimension D computed in every time frame t and at every location s in the training sequence (see § 4.1 for details), and let

$$\mathbf{y} \in \{-1, 0, 1\}^{\mathcal{T} \times \mathcal{S}},$$

be the available ground-truth for every time frame and every location, where the value 0 stands for "not available".

Table 1. Notation

$\mathcal{T} = \{1, \ldots, T\}$	time steps
$\mathcal{S} = \{1, \ldots, W\} \times \{1, \ldots, H\}$	spatial locations
$x_{t,s} \in \mathbb{R}^D$	feature vector at location s in time frame t
$y_{t,s} \in \{-1, 0, 1\}$	ground-truth label
$h_m : \mathbb{R}^D \to \{-1, 1\}$	weak learners
$\mathcal{H} = span\{h_1, \ldots, h_M\}$	the combinations of weak learners
$\varphi(x) \in \mathcal{H}$	a strong classifier
$\mathcal{V} = \mathcal{T} \times \mathcal{S}$	vertices of the motion graph
$\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$	edges of the motion graph
$\mathcal{F} \subset \{0, 1\}^{\mathcal{V}}$	Boolean labelings of the graph consistent with multi-target motions
$L_{\mathcal{V}}(\varphi; \mathbf{y})$	exponential loss summed on vertices
$L_{\mathcal{E}}(\varphi; f)$	exponential loss summed on edges
$\Theta(f)$	signed labels on vertices corresponding to the Boolean labeling f of the edges.

[1] We consider in this article only locations in the image plane, but additional parameters such as scale or rotation could be handled in a similar manner, albeit with an increase of the computational cost.

3.1. Boosting

We quickly summarize here some characteristics of AdaBoost relevant to the understanding of our approach. Let

$$h_m : \mathbb{R}^D \to \{-1, 1\},\ m = 1, \dots, M$$

be a family of "weak learners", and

$$\mathcal{H} = span\{h_1, \dots, h_M\}$$

the set of linear combinations of weak-learners. Given a labeling $\mathbf{y}$ and a mapping

$$\varphi : \mathbb{R}^D \to \mathbb{R}$$

we can define the exponential loss

$$L_{\mathcal{V}}(\varphi; \mathbf{y}) = \sum_{(t,s) \in \mathcal{V}} \exp\left(-y_{t,s}\varphi(x_{t,s})\right) \tag{1}$$

which is low when φ takes values consistent with the labeling $\mathbf{y}$. This loss ignores samples whose labels are equal to 0. Boosting consists in approximating the mapping

$$\varphi^* = \operatorname*{argmin}_{\varphi \in \mathcal{H}} L_{\mathcal{V}}(\varphi; \mathbf{y})$$

by successively picking Q weak learners h_{m_q} and weights $\omega_q \in \mathbb{R}$ such that $L_{\mathcal{V}}$ is reduced in a greedy fashion. In our experiments, we used stumps, parameterized by a feature index $d \in \{1, \dots, D\}$ and a threshold $\rho \in \mathbb{R}$, as weak learners of the form:

$$\forall x \in \mathbb{R}^D, \quad h(x) = 2 \cdot \mathbf{1}_{\{x^d \geq \rho\}} - 1.$$

3.2. Time-based regularization

The core idea of our approach is to automatically compute labels for non-labeled samples, which are as consistent as possible with an existing classifier and with a constraint of continuous motion for the targets.

Our estimation of a labeling consistent with the physically possible motions is strongly inspired from the convex multi-target tracking of [2]. Prior knowledge regarding target motions is represented by a directed "motion graph", which possesses one vertex for each (t, s) pair, where t is a time step and s a spatial location. Let

$$\mathcal{V} = \mathcal{T} \times \mathcal{S}$$

be this set of vertices. The edges of that graph correspond to admissible motions:

$$\mathcal{E} = \left\{ \big((t, s), (t + 1, s')\big), 1 \leq t < T, s \in \mathcal{S},\ s' \in \mathcal{N}(s) \right\},$$

where $\mathcal{N}(s) \subset \mathcal{S}$ denotes the locations a target located in s at time t can reach at time $t + 1$. Given such a graph, we optimize a flow

$$f : \mathcal{E} \to \{0, 1\}$$

which associates to every edge of the motion graph the number of targets moving along it. If the flow f is equal to 0 on the edge from (t, s) to $(t + 1, s')$, it means that no target moves from location s at time t to locations s' at time $t + 1$, and conversely if f is equal to 1 on that edge, it means that a target makes that motion at that moment. Let $\mathcal{F}$ stand for the set of Boolean labeling of the edges which are physically plausible and therefore obey the following constraints: (1) the flow on each edge is smaller than one and greater than 0, and (2) the sum of the flow on the edges arriving at a certain vertex is equal to the sum of the flows on the edges leaving that vertex.

For clarity, for any edge $e = \{(t, s), (t + 1, s')\} \in \mathcal{E}$, we will use $\varphi(e)$ as a short-cut for $\varphi(x_{t,s})$, that is the value of the classifier at the time and location corresponding to the originating vertex of the edge. To select a flow consistent with the responses of the classifier, we minimize the exponential loss of Equation (1), as during Boosting. This means that the flow f should minimize

$$L_{\mathcal{E}}(\varphi; f) = \sum_{e \in \mathcal{E}} \exp(-(2f(e) - 1)\varphi(e)). \tag{2}$$

Since f takes its values in $\{0, 1\}$, we have

$$\begin{aligned}
f^* &= \operatorname*{argmin}_{f \in \mathcal{F}} L_{\mathcal{E}}(\varphi; f) \\
&= \operatorname*{argmin}_{f \in \mathcal{F}} \sum_{e \in \mathcal{E}} \exp(-(2f(e) - 1)\varphi(e)) \\
&= \operatorname*{argmin}_{f \in \mathcal{F}} \sum_{e \in \mathcal{E}} \exp(-\varphi(e))f(e) + \exp(\varphi(e))(1 - f(e)) \\
&= \operatorname*{argmin}_{f \in \mathcal{F}} \sum_{e \in \mathcal{E}} \left(\exp(-\varphi(e)) - \exp(\varphi(e))\right) f(e).
\end{aligned}$$

As in [2], we have to minimize a linear cost, under the linear equalities of conservation of flow at every vertex, and the linear inequalities corresponding to an upper-bound of one target moving per edge, and a lower bound of zero. If we relax the binary flow constraint and let f take continuous values in $[0, 1]$, this results into a convex linear programming system, which can be solved optimally[2].

Additionally, entrance into the graph is made possible by relaxing the constraint of conservation of flow for vertices corresponding to "virtual locations", connected to the border of the images, where targets can appear or disappear. Finally, the flow is forced to pass through vertices for which we have explicit positive labels and conversely it is prevented from passing through vertices with explicit negative labels by setting the scores appropriately.

[2] As shown in [2], this linear programming system can equivalently be solved with the more efficient K-Shortest Paths (KSP) algorithm. Our implementation relies on the KSP algorithm and in the remainder of this paper the terms KSP and linear programming system are used interchangeably.

3.3. Iterative optimization

Given the Boosting procedure, and the time-based regularization described above, we depict here our iterative learning process. Let

$$\mathbf{y}^{(0)} \in \{-1, 0, 1\}^{\mathcal{T} \times \mathcal{S}}$$

be the initial labeling provided by experts. In practice, only one frame in every M will be labeled, leading to labels equal to ± 1 at all the locations in these frames, and 0 everywhere in other frames. Given that initial labeling, we define the following algorithm:

for $k = 1, \ldots, K$ **do**
$\quad \varphi^{(k)} \leftarrow \operatorname*{argmin}_{\varphi \in \mathcal{H}} L_{\mathcal{V}} \left(\varphi; \mathbf{y}^{(k-1)}\right)$
$\quad f^{(k)} \leftarrow \operatorname*{argmin}_{f \in \mathcal{F}} L_{\mathcal{E}} \left(\varphi^{(k)}; f\right)$
$\quad \mathbf{y}^{(k)} \leftarrow \Theta \left(f^{(k)}\right)$
end for

To summarize, we train with Boosting a first classifier $\varphi^{(1)}$ from the scarce labeling provided by experts, then compute an optimal Boolean labeling of edges $f^{(1)}$, consistent with physical motions of targets, and from it a new signed labeling $\mathbf{y}^{(1)}$ over all frames and locations. From that new labeling, we train with Boosting a second classifier $\varphi^{(2)}$, etc. The Θ operator computes the ± 1 labels on vertices, from the Boolean labels on edges. Precisely:

$$\Theta(f)_{t,s} = 2 \sum_{s' \in \mathcal{N}(s)} f\left((t, s), (t + 1, s')\right) \; - 1$$

Hence, both the Boosting and the KSP procedure are minimizing a common exponential loss function. This loss favors consistency between the response maps generated by the classifier and the Boolean flows while penalizing discrepancy.

3.4. Implementation details

In this section, we outline specific implementation details used in our our algorithm.

3.4.1 Numerical stability

The cost of Equation (2) grows quickly with a high classifier response $\varphi(e)$. This naturally causes numerical stability issues for the convex linear programming system. For this reason, given that, as was shown in [5],

$$\varphi(x_{t,s}) \simeq \frac{1}{2} \log \frac{\hat{P}(y_{s,t} = 1 | x_{s,t})}{\hat{P}(y_{s,t} = -1 | x_{s,t})}, \tag{3}$$

we clip the classifier response $\varphi(x_{t,s})$ at ± 8.0 which corresponds to allowing a maximum classifier confidence of $P(y_{s,t} = \pm 1 | x_{s,t}) = 1 - 10^{-7}$.

3.4.2 Regularization

Non-Maxima Suppression. Ideally, at every iteration, the classifier would feed the linear programming system a dense response map. Doing so, however, would result in a cluster of trajectories around each target being output by the linear programming system. This behavior is not desirable from both a computational perspective, in that the linear optimization would deploy vast resources to resolve numerous trajectories centered on each target, and from a learning perspective in that multiple shifted copies of a target would be fed back to the Boosting stage at subsequent iterations. For this reason, the linear programming system is fed a sparse response map obtained by applying standard non-maxima suppression to the dense response map *without thresholding*.

Trajectory Filtering. The linear programming system allows for an unconstrained number of targets to appear and disappear from any location along the boundary of the video frame. This behavior is naturally desired to maintain the generality of the approach. A drawback of this approach however is that as soon as the classifier outputs a positive response on a boundary point, the linear programming system will admit a trajectory at that location even if its duration is only one frame. To mitigate this effect, we introduced a simple heuristic that rejects any trajectory which does not venture deep enough (10 pixels) into the middle of the scene.

Soft Consensus. We introduced a final heuristic in order to include a hard confidence estimation on the output of the KSP stage and only retain the most confident samples for the subsequent iteration. To this end, the samples along the trajectories output by the linear programming system are not fed as a whole to the Boosting stage of the next iteration. Instead, they are sorted according the response of the classifier of the current stage and the bottom 25% are pruned. This choice was *ad hoc* and was not optimized on our test sets.

4. Experimental Results

To evaluate the performance of our proposed learning strategy, experiments were performed on two different data sets: video sequences of pedestrians and time-lapse microscopy data containing migrating neurons. In what follows, the specifics of our experimental setup are given and the results of our experiments provided.

4.1. Image features

We describe here the image features that we used for the learning component. A scene x is preprocessed by computing and thresholding the derivatives of the image intensity to obtain an edge image. The orientation of these edges are further quantized into q bins, resulting in q edge maps. Let ϕ

denote the possible orientations of an edge on $\Phi = [-\pi, \pi[$, and let $\hat{\Phi} = \{0, \frac{2\pi}{q}, \frac{4\pi}{q}, \dots, (q-1) * \frac{2\pi}{q}\}$ denote the possible orientations of a *quantized* edge.

Now $\forall e \in \hat{\Phi},\ x \in \mathcal{I},\ s \in \{1, \dots, W\} \times \{1, \dots, H\}$, let

$$\xi_e(x, s) \in \{0,\ 1\}, \tag{4}$$

denote the presence of an edge with quantized orientation e at pixel s in image x. We assume $\xi_e(x, s)$ is equal to 0 if the location s is not in the image plane. Thus, each $\xi_e(x, s)$ is simply a map of edges with quantized orientation e.

Our features, as those in [1], compute the ratio of edges of a particular orientation within a sub-window of the detector's $r \times r$ square of interest, with respect to the total number of edges within the same sub-window. Let R denote such a sub-window of random size and location contained in $\{1, \dots, r\} \times \{1, \dots, r\}$ plane. Our features are entirely parameterized by the sub-window R and the edge type e and are defined as:

$$h_{R,e}(x) = \sum_{m \in R} \xi_e(x, m) \;/ \sum_{d \in \hat{\Phi},\, m \in R} \xi_d(x, m). \tag{5}$$

These features give the classifier the ability to check for the presence of outlines and textures and can be computed in constant time using q integral images, one for each edge map. In all our experiments, we used $q = 8$.

4.2. AdaBoost

The standard AdaBoost learning procedure is used. The selection of the stump at every iteration of AdaBoost results from examining 1000 of our features. The threshold ρ_i of the selected stumps is optimized through an exhaustive search. Finally, feature parameters R and e are chosen uniformly at random. In all our experiments, a single AdaBoost stage is trained with the bootstrapping procedure described in [4]: this allows us to avoid the difficulties associated with training and tuning a cascade.

4.3. Data sets

Pedestrian data Our pedestrian data set contains two sequences: a training sequence and a testing sequence. This data was obtained by mounting a standard DV camera in a slightly elevated location over a central and relatively busy campus site. The training and testing sequences are non overlapping in time and both sequences contain multiple pedestrians entering and exiting the scene from all borders of the images. Some positive patches sampled uniformly at random from the test sequence are shown in Figure 1.

Both sequences have a resolution of 568×328, a frame rate of 25*fps*, contain approximately 1000 frames and approximately 3500 positive samples. The sequences are relatively challenging in that pedestrians often pass through the scene walking closely to each other and thereby partially occluding each other. In addition, a number of pedestrians walk through a relatively unlit area and a large range in pose variation is observed. The detector's window size for these sequences is 68×68.

Neuron data Our neuron data set is made up of 14 sequences of resolution 168×128 each containing exactly 97 frames. The sequences show developing neurons in a cell culture imaged with bright-field microscopy and using Green Fluorescent Protein staining. Each sequence is imaged over 16 hours with an image taken every 10 minutes. The detector's window size for these sequences is 34×34.

Ordinarily, these developing neurons would move in a guided fashion while extending and retracting neurites seeking protein signals. However given that they are in a cell culture, they are moving randomly. These sequences are highly challenging as the neurons vary greatly in appearance and can move almost their entire length from one frame to the next. In addition, the microscope periodically loses focus causing a drastic change in appearance in the neurons.

Out of these 14 short 97 frame sequences, we build a training sequence by randomly selecting 8 of the sequences, while reserving the remaining 6 for testing. The training and test sequences contain approximately 3000 neurons in motion each. Some positive patches selected uniformly at random from the test sequence are shown in Figure 3.

4.4. Error rates

Error rates were computed in a conservative fashion. A detection is a true alarm if its location is within a certain distance from the target and a false alarms otherwise. The considered distance is 0.25 times the length of the detector's square window of interest for the pedestrian data and 0.4 times for the neuron data set. The choice of these numbers reflect how closely two targets may appear to each other in each data set. Two targets may still lie within the above mentioned distance. In this scenario, if only one alarm is raised, a miss is counted.

4.5. Baselines and FlowBoost

In all our experiments we compare the performance of our algorithm against two baselines. The first is an AdaBoost classifier as described above trained with 200 stumps and using the full training data. The second is an AdaBoost classifier, as described above, trained with 200 stumps but using the sparse labeling. We varied the labeling rate with each experiments by annotating 1 frame in 16, 32, 64 and 128.

The results are shown for the first three iterations of FlowBoost. The first iteration trains an AdaBoost stage with only 50 stumps, while the remaining two iterations train an AdaBoost stage with 200 stumps.

4.6. Results

Experiments were run on the pedestrian sequences for the cases where 1 frame is labeled every 32, 64 and 128 frames. This corresponds to initial training sets containing 114, 61 and 29 positive examples out of a population of 3500. Results are shown in Figure 2. For the neuron data set, experiments were run for the cases where 1 frame is labeled every 16, 32 and 64 frames. This corresponds to initial training sets containing 228, 141 and 102 positive examples out of a population of 3000. Results are shown in Figure 4.

Results on both data sets are very good, confirming the soundness of this approach. With the exception of the experiment where 1 frame in 128 is labeled for the pedestrian data set (see Figure 2 (c)), FlowBoost's third iteration is able to *outperform* a standard Boosting procedure trained with the entire data set. This is a truly surprising result which can be explained by the ability of the system to register data in translation more precisely, or at the very least in a more learning-friendly manner, than a human annotator.

Performance at very high true positive rates is still often slightly worse when compared to the fully labeled baseline. It is indeed extremely difficult to handle unlabeled positive outliers: trajectories passing through truly positive, yet highly unusual, samples may in fact score as bad as trajectories passing through the background. This, unfortunately, cannot be handled in a totally unsupervised manner without additional knowledge.

5. Concluding Remarks

We presented a novel approach that propagates a sparse labeling of a training video to every frame in a manner consistent with the known physical constraints on target motions. Experiments demonstrate that, except at very high conservative regimes, our algorithm trained with less than 3% of the labels is able to outperform an equivalent Boosting algorithm trained with the fully labeled set.

This work admits several natural extensions. The first is to cope with geometrical poses of greater complexity. In both of our applications, we only considered location in the image plane, without variations in scale or orientation. While this extension is straightforward formally, the computational costs could be prohibitory. Hence, it would require additional development in hierarchical representation or adaptive evaluation.

The second is the explicit inclusion of the non-maximum suppression in the iterative framework: the flow should be optimized so that, *through the non-maximum suppression*, it is consistent with the classifier response. In the current version, this is ignored, which leads to suboptimal performance.

The third, finally, is the development of an adaptive pruning of the samples to use after the flow-relabeling. The current choice of the top 75% is ad-hoc, and probably a poor man's version of a confidence estimation based on the distance to the separation boundary.

Aknowledgment

The authors thank Kevin Smith, Engin Türetken and Jérôme Berclaz for their invaluable help.

References

[1] K. Ali, F. Fleuret, D. Hasler, and P. Fua. Joint Pose Estimator and Feature Learning for Object Detection. In *International Conference on Computer Vision*, 2009.

[2] J. Berclaz, F. Fleuret, E. Türetken, and P. Fua. Multiple Object Tracking Using K-Shortest Paths Optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

[3] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *International Conference on Machine Learning*, pages 19–26, 2001.

[4] F. Fleuret and D. Geman. Stationary Features and Cat Detection. *Journal of Machine Learning Research*, 9:2549–2578, 2008.

[5] J. Friedman, T. Hastie, and Tibshirani. Additive Logistic Regression: a Statistical View of Boosting. *Annals of Statistics*, 28:2000, 1998.

[6] T. Joachims. Transductive inference for text classification using support vector machines. In *International Conference on Machine Learning*, pages 200–209, 1999.

[7] T. Joachims. Transductive learning via spectral graph partitioning. In *International Conference on Machine Learning*, pages 290–297, 2003.

[8] Z. Kalal, J. Matas, and K. Mikolajczyk. Online Learning of Robust Object Detectors During Unstable Tracking. In *International Conference on Computer Vision*, 2009.

[9] Z. Kalal, J. Matas, and K. Mikolajczyk. P-N Learning: Bootstrapping Binary Classifiers from Unlabeled Data by Structural Constraints. In *Conference on Computer Vision and Pattern Recognition*, 2010.

[10] N. D. Lawrence and M. I. Jordan. Semi-supervised learning via gaussian processes. In *Advances in Neural Information Processing Systems*, pages 753–760, 2005.

[11] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Text Classification from Labeled and Unlabeled Documents Using Em. In *Machine Learning*, pages 103–134, 1999.

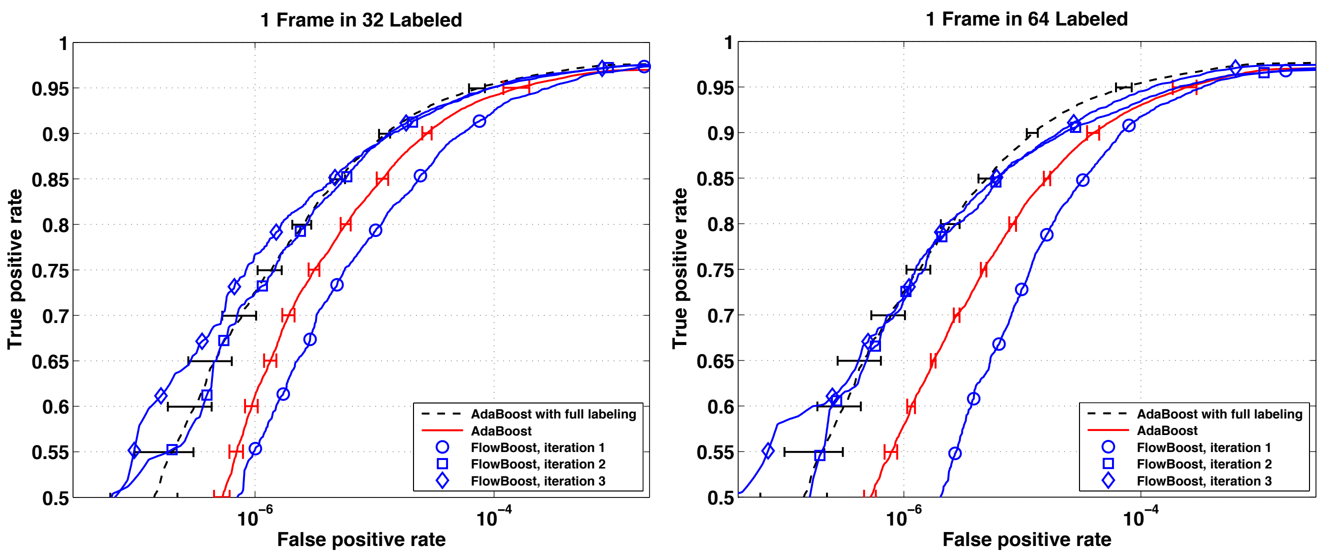
[12] H. Scudder. Probability of Error of Some Adaptive Pattern-Recognition Machines. *Information Theory, IEEE Transactions on*, 11(3):363–371, July 1965.

[13] M. Szummer and T. Jaakkola. Partially labeled classification with markov random walks. In *Advances in Neural Information Processing Systems*, pages 945–952, 2002.

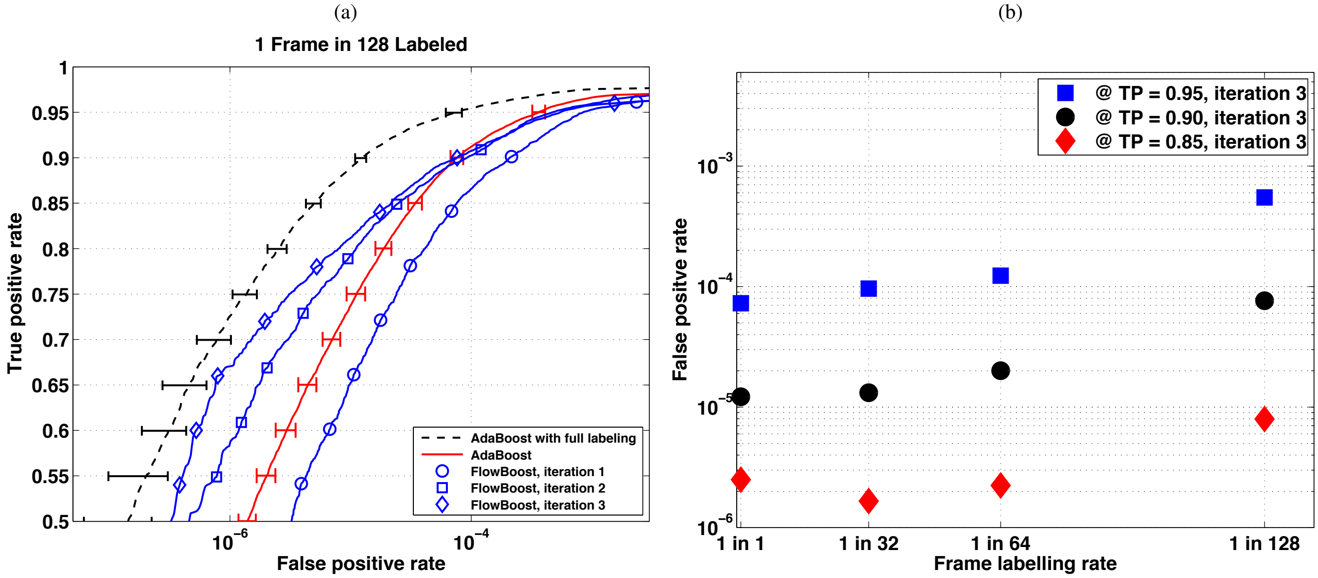
[14] V. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, New York, 1998.

[15] X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Carnegie Mellon University, 2002.

Figure 1. Some examples of people from our pedestrian test video taken uniformly at random across the entire set.



(a) (b)



(c) (d)

Figure 2. Performance of our learning framework compared with a standard labeling for our pedestrian data set. Figures (a,b,c) display true-positive rate as a function of the false alarms rate on a log scale for the case (a) where one frame in 32 is annotated, (b) where one frame in 64 is annotated and (c) where 1 frame in 128 is annotated. Figure (d) displays the false alarm rate at various true positive rate as a function of the number of labeled frames.

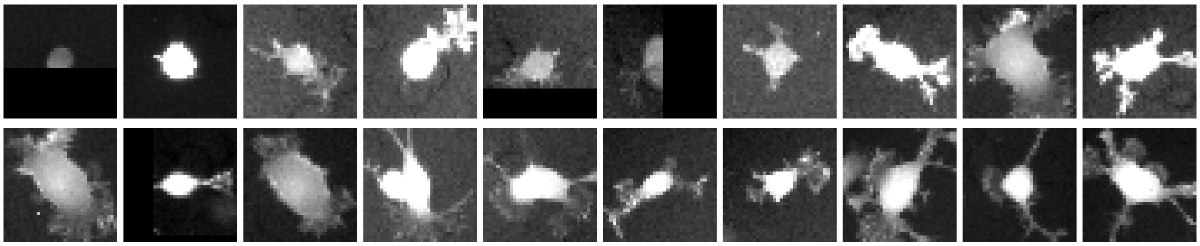


Figure 3. Some examples of migrating neurons taken uniformly at random across our microscopy test data.

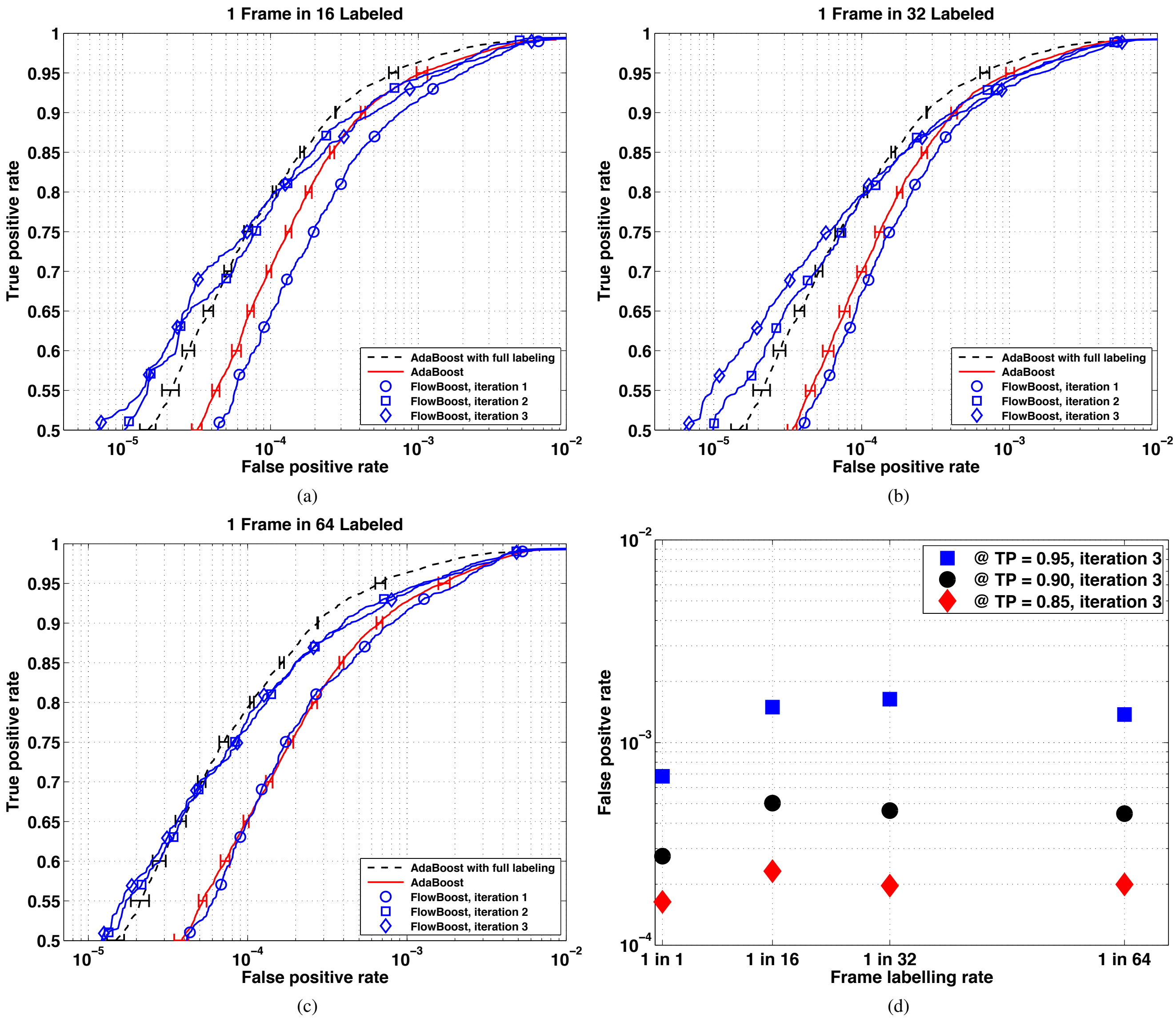


Figure 4. Performance of our learning framework compared with a standard labeling for our neuron data set. Figures (a,b,c) display true-positive rate as a function of the false alarms rate on a log scale for the case (a) where one frame in 16 is annotated, (b) where one frame in 32 is annotated and (c) where 1 frame in 64 is annotated. Figure (d) displays the false alarm rate at various true positive rate as a function of the number of labeled frames.