Latent Boosting for Action Recognition

Zhi Feng Huang[1]
zfh@sfu.ca
Weilong Yang[1]
wya16@sfu.ca
Yang Wang[2]
yangwang@uiuc.edu
Greg Mori[1]
mori@cs.sfu.ca

[1] School of Computing Science
Simon Fraser University,
British Columbia, Canada

[2] Department of Computer Science
University of Illinois at
Urbana-Champaign,
Illinois, USA

Abstract

In this paper we present LatentBoost, a novel learning algorithm for training models with latent variables in a boosting framework. This algorithm allows for training of structured latent variable models with boosting. The popular latent SVM framework allows for training of models with structured latent variables in a max-margin framework. LatentBoost provides an analogous capability for boosting algorithms. The effectiveness of this framework is highlighted by an application to human action recognition. We show that LatentBoost can be used to train an action recognition model in which the trajectory of a person is a latent variable. This model outperforms baselines on a variety of datasets.

1 Introduction

In this paper we describe a novel boosting algorithm with latent variables, applied to human action recognition. Consider the problem of recognizing an action such as *running* in a surveillance video. A typical approach involves first detecting and tracking people, followed by classification. However, accurate tracking is challenging. As illustrated in Fig. 1, trackers will suffer from jitter, especially when people are performing varied actions. Since accurate tracking is not a direct end-goal of action recognition, it is natural to consider tracking as a latent variable and train a model focused on action recognition.

Recently, the latent SVM [6] has proven effective for modelling latent variables in a variety of vision tasks. One view on its success is the argument that alignment or registration of data can be performed using latent variables. From the learning perspective, this is a version of the multiple instance learning framework. The latent SVM is a variant of the mulitple instance SVM (MI-SVM) [1]. The latent SVM explicitly models the structure of the latent variables that standard MI-SVM does not.

A similar vein can be followed in the boosting literature. Boosting algorithms have been widely used for visual recognition, e.g. the Viola-Jones face detector [16]. A multiple instance learning variant of boosting algorithm was developed in [17] to handle imperfections in labeling face locations on the training data. A similar model was later used for visual tracking [2]. The latent variables in these models are simply the location of a face in an

Figure 1: Typical tracklets from the TRECVID dataset. The first row consists two 5-frames tracklets of running people. The second row consists two 5-frames tracklets of not-running people.

image. In our model, however, the latent variables can have much more complex structures, e.g. the space-time locations of a person in a track.

Boosting algorithms have desirable properties, for instance the ability to use many non-linear weak learners and features, and avoiding the need to set SVM slack-tradeoff (C) parameters for large numbers of features. A boosting algorithm that allows for structured latent variables has potential to be useful for a variety of recognition problems. Hence, in this paper we develop a boosting algorithm for training models with structured latent variables. We show that this approach is competitive, and outperforms comparable boosting-based approaches for human action recognition.

Human action recognition is a very active area of research. Weinland et al. [19], Turaga et al. [15], and Poppe [12] provide recent surveys. We review closely related work below. Probabilistic models with latent variables, such as the hidden conditional random field (hCRF) [13], have been explored for action recognition. Wang and Mori [18] develop a similar model to ours, yet with max-margin and probabilistic learning criteria different from our boosting approach. Boosting algorithms are commonly used in action recognition. For example, Laptev and Pérez [10] learn a cascade of boosted action classifiers to detect actions in movies. Fathi and Mori [5] and Kim and Cipolla [9] use boosting to select a subset of features for discriminating between actions. Our focus is on an algorithm for incoporating latent variables, different from these pieces of work.

2 Boosting with Latent Structures

In this section, we first briefly review GradientBoost algorithm proposed by Friedman [7] in Sec. 2.1, which also serves as the baseline approach in our experiment. Then, we present our LatentBoost algorithm in Sec. 2.2.

2.1 Multi-class GradientBoost [7]

Let us consider a classification problem with K classes. We denote a class label y as a length K binary vector of all zeros with a single one for the corresponding class. For example, for the k-th class, the class label y is written as $y = [y_1, y_2, ..., y_K]$, where $y_k = 1$ and $y_l = 0$ for all $l \neq k$. In GradientBoost, the goal is to learn a set of scoring functions $\{F_k(\mathbf{x})\}_{k=1}^{K}$, one for

each class. The probability $p_k(\mathbf{x})$ of an example $\mathbf{x}$ being class k can be defined as:

$$p_k(\mathbf{x}) = \frac{\exp(F_k(\mathbf{x}))}{\sum_{l=1}^{K}\exp(F_l(\mathbf{x}))} \tag{1}$$

Given a set of N training examples $\{\mathbf{x}^{(n)}, y^{(n)}\}_{n=1}^{N}$, GradientBoost learns the set of scoring functions $\{F_k(\mathbf{x})\}_{k=1}^{K}$ by minimizing the negative log-loss of the training data

$$loss_{GB} = \sum_{n=1}^{N}\Psi(\{y_k^{(n)}, F_k(\mathbf{x}^{(n)})\}_{k=1}^{K}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} y_k^{(n)}\log p_k(\mathbf{x}^{(n)}). \tag{2}$$

Under the boosting framework, the scoring function $F_k(\mathbf{x})$ for the k-th class is assumed to be a linear combination of so-called "weak learners":

$$F_k(\mathbf{x}) = \sum_{m=1}^{M}\rho_{k,m}h_{k,m}(\mathbf{x}) \tag{3}$$

where $h_{k,m}$ is the m-th weak learner for the k-th class, and $\rho_{k,m}$ is the weight associated with this weak learner. Let $F_{k,m}(\mathbf{x})$ be the scoring function for the k-th class after m iterations. At the m-th iteration of the gradient boosting algorithm, for the k-th class, a new weak learner $h_{k,m}$ and its weight $\beta_{k,m}$ are obtained by solving the following optimization problem:

$$(\beta_{k,m}, h_{k,m}) = \arg\min_{\beta,h}\sum_{n=1}^{N}\Psi(y_k^{(n)}, F_{k,m-1}(\mathbf{x}^{(n)}) + \beta h(\mathbf{x}^{(n)})). \tag{4}$$

Then the scoring function for the k-th class will be updated as: $F_{k,m}(\mathbf{x}) = F_{k,m-1}(\mathbf{x}) + \beta_{k,m}h_{k,m}(\mathbf{x})$. Let us define the functional gradient of the loss function with respect to the current estimated $F_{k,m-1}(\mathbf{x})$ $(k = 1,2,...,K)$ as

$$-g_{k,m}(\mathbf{x}^{(n)}) = -\left[\frac{\partial\Psi(\{y_j^{(n)}, F_j(\mathbf{x}^{(n)})\}_{j=1}^{K})}{\partial F_k(\mathbf{x}^{(n)})}\right]_{\{F_j(\mathbf{x})=F_{j,m-1}(\mathbf{x})\}_{j=1}^{K}} = y_k^{(n)} - p_k(\mathbf{x}^{(n)}),\ n = 1,2,...,N \tag{5}$$

As noted in [7], $h(\mathbf{x})$ is chosen from a finite size pool of weak learners, it usually can not have the exact form as the function $-g(\mathbf{x})$. One alernative approach is to select the $h(\mathbf{x})$ that is the most parallel in the N-dimensional data space with the negative gradient $\{-g(\mathbf{x}^{(n)})\}_1^N$ by the following least-squares minimization problem:

$$h_{k,m} = \arg\min_{h}\sum_{n=1}^{N}[-g_{k,m}(\mathbf{x}^{(n)}) - h(\mathbf{x}^{(n)})]^2 \tag{6}$$

After $h_{k,m}$ is chosen, its weight $\rho_{k,m}$ can be found by line search. Algorithm 1 illustrates the K-class GradientBoost approach.

2.2 LatentBoost

In discriminative latent structure learning frameworks (i.e. latent SVM [6] or HCRF [13]), each training sample is usually assumed to be associated with a set of latent variables. The latent variables can be structured in some way. For example, in Sec. 3, we consider person

Algorithm 1 K-class GradientBoost

$F_{k,0} = 0, k = 1, \ldots, K$
for $m = 1$ to M **do**
  $p_k(\mathbf{x}) = \exp(F_k(\mathbf{x}))/\sum_{l=1}^{K}\exp(F_l(\mathbf{x})),\ k = 1, \ldots, K$
  for $k = 1$ to K **do**
    $-g_{k,m}(\mathbf{x}^{(n)}) = y_k^{(n)} - p_k(\mathbf{x}^{(n)}),\ n = 1, \ldots, N$
    $h_{k,m} = \arg\min_h \sum_{n=1}^{N}[-g_{k,m}(\mathbf{x}^{(n)}) - h(\mathbf{x}^{(n)})]^2$
    $\rho_{k,m} = \arg\min_\rho \sum_{n=1}^{N}\Psi(y_k^{(n)}, F_{k,m-1}(\mathbf{x}^{(n)}) + \rho h_{k,m}(\mathbf{x}^{(n)}))$
    $F_{k,m} = F_{k,m}(\mathbf{x}) + \rho_{k,m}h_{k,m}(\mathbf{x})$
  end for
end for

tracks as latent variables, which are constrained by a chain structure. We emphasize that our LatentBoost algorithm is not limited to this chain structure but can be easily generalized to other types of latent structures.

We assume that an example $(\mathbf{x}, y)$ is associated with a set of latent variables $\mathbf{L} = \{l_1, l_2, \ldots, l_T\}$, where each latent variable takes its value from a discrete set, i.e. $l_t \in \mathscr{L}$[1]. We assume these latent variables are constrained by an undirected graph structure $\mathscr{G} = (\mathscr{V}, \mathscr{E})$, where $\mathscr{V}$ and $\mathscr{E}$ denote vertices and edges in the graph $\mathscr{G}$, respectively. For a fixed $\mathbf{L}$, the scoring function of the $(\mathbf{x}, \mathbf{L})$ pair for the k-th class can be written as the sum of a set of unary and pairwise potential functions:

$$F_k(\mathbf{x}, \mathbf{L}) = \sum_{t \in \mathscr{V}} H_k^t(\mathbf{x}, l_t) + \sum_{(t,s) \in \mathscr{E}} H_k^{t,s}(\mathbf{x}, l_t, l_s) \tag{7}$$

Under the boosting framework, we define $H_k^t(\mathbf{x}, l_t)$ and $H_k^{t,s}(\mathbf{x}, l_t, l_s)$ as linear combinations of weak learners:

$$H_k^t(\mathbf{x}, l_t) = \sum_{m=1}^{M} \rho_{k,m}^t h_{k,m}^t(\mathbf{x}, l_t) \tag{8}$$

$$H_k^{t,s}(\mathbf{x}, l_t, l_s) = \sum_{m=1}^{M} \rho_{k,m}^{t,s} h_{k,m}^{t,s}(\mathbf{x}, l_t, l_s) \tag{9}$$

Similar to GradientBoost, we define the probability of an example $\mathbf{x}$ being class k as:

$$\hat{p}_k(\mathbf{x}) = \frac{\sum_L \exp(F_k(\mathbf{x}, \mathbf{L}))}{\sum_L \sum_{l=1}^{K} \exp(F_l(\mathbf{x}, \mathbf{L}))} \tag{10}$$

The difference (comparing Eq. 1 and Eq. 10) from GradientBoost is that now we need to sum over $\mathbf{L}$ since they are latent variables. Similarly, we can define the loss function for LatentBoost as the negative log-likelihood of the training data:

$$loss_{LB} = \sum_{n=1}^{N} \Psi(\{y_k^{(n)}, F_k(\mathbf{x}^{(n)}, \mathbf{L})\}_{k=1}^{K}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} y_k \log \hat{p}_k^{(n)}(\mathbf{x}^{(n)}) \tag{11}$$

[1]To simplify notation, we assume the same set $\mathscr{L}$. But our formulation can be generalized so that each latent variable is associated with a different label set.

Similar to GradientBoost, we learn the weak learners (both unary and pairwise) and their associated weights in an iterative fashion. Let us first focus on the unary potential $H_k^t(\mathbf{x}, l_t)$. At the m-th iteration, the gradient of the loss function $loss_{LB}$ with respect to current strong learners can be calculated as:

$$\begin{aligned} -g_{k,m}^t(\mathbf{x}^{(n)}, l_t) &= -\left[\frac{\partial \Psi(\{y_j^{(n)}, F_j(\mathbf{x}^{(n)}, \mathbf{L})\}_{j=1}^K)}{\partial H_k^t(\mathbf{x}^{(n)}, l_t)}\right]_{\{F_j(\mathbf{x},\mathbf{L})=F_{j,m-1}(\mathbf{x},\mathbf{L})\}_{j=1}^K} \\ &= y_k^{(n)} \frac{\sum_{\mathbf{L}:l_t} \exp(F_{k,m-1}(\mathbf{x}^{(n)}, \mathbf{L}))}{\sum_{\mathbf{L}} \exp(F_{k,m-1}(\mathbf{x}^{(n)}, \mathbf{L}))} - \frac{\sum_{\mathbf{L}:l_t} \exp(F_{k,m-1}(\mathbf{x}^{(n)}, \mathbf{L}))}{\sum_{j=1}^K \sum_{\mathbf{L}} \exp(F_{j,m-1}(\mathbf{x}^{(n)}, \mathbf{L}))} \\ &= y_k^{(n)} \Pr(l_t|\mathbf{x}^{(n)}) - \Pr(y_k^{(n)} = 1, l_t|\mathbf{x}^{(n)}) \end{aligned} \tag{12}$$

Then, we can obtain the optimal weak learner $h_{k,m}^t(\mathbf{x}, l_t)$ by solving the following least-squares minimization problem:

$$h_{k,m}^t = \arg\min_{h^t} \sum_{n=1}^N \sum_{l_t \in \mathscr{L}} [-g_{k,m}^t(\mathbf{x}^{(n)}, l_t) - h^t(\mathbf{x}^{(n)}, l_t)]^2 \tag{13}$$

Similarly, the negative gradient for the pairwise term is as follows:

$$\begin{aligned} -g_{k,m}^{t,s}(\mathbf{x}^{(n)}, l_t, l_q) &= -\left[\frac{\partial \Psi(\{y_j^{(n)}, F_j(\mathbf{x}^{(n)}, \mathbf{L})\}_{j=1}^K)}{\partial H_k^{t,s}(\mathbf{x}^{(n)}, l_t, l_s)}\right]_{\{F_j(\mathbf{x},\mathbf{L})=F_{j,m-1}(\mathbf{x},\mathbf{L})\}_{j=1}^K} \\ &= y_k^{(n)} \Pr(l_t, l_s|\mathbf{x}^{(n)}) - \Pr(y_k^{(n)} = 1, l_t, l_s|\mathbf{x}^{(n)}) \end{aligned} \tag{14}$$

where the marginal distributions $\Pr(l_t|\mathbf{x}^{(n)})$, $\Pr(y_k^{(n)} = 1, l_t|\mathbf{x}^{(n)})$, $\Pr(l_t, l_s|\mathbf{x}^{(n)})$ and $\Pr(y_k^{(n)} = 1, l_t, l_s|\mathbf{x}^{(n)})$ can be computed efficiently by using Belief Propagation (or be approximated by using Loopy Belief Propagation if the latent structure has loops).

The weights $\rho_{k,m}^t$ and $\rho_{k,m}^{t,s}$ can be simply computed by a line search algorithm. Putting everything together, we have the LatentBoost algorithm illustrated in Algorithm 2.

3 LatentBoost for Human Action Recognition

We test LatentBoost on the task of human action recognition. In Sec. 3.1, we describe the LatentBoost model we implement. In Sec. 3.2, we describe the features we use and implementation details of LatentBoost.

3.1 Model

Our method for human action recognition operates on a "figure-centric" representation of the human figure extracted from an input video. The figure-centric representation is obtained by running a human detection/tracking algorithm over the input video. We use the term tracklet to denote short 5-frame long human trajectories returned by our tracker. Our method will operate on these tracklets, and classify them into one of K actions.

A tracklet is denoted by 5 tuples $x_t = (I_t, u_t, v_t)$, $t = 1, \ldots, 5$ where I_t is the image feature and (u_t, v_t) are the position of person in the t-th frame of the tracklet. Since the tracker is

Algorithm 2 LatentBoost

$F_{k,0} = 0, k = 1, \ldots, K$
for $m = 1$ to M **do**
  Compute $\Pr(l_t|\mathbf{x}^{(n)})$, $\Pr(y_k^{(n)} = 1, l_t|\mathbf{x}^{(n)})$, $\Pr(l_t, l_s|\mathbf{x}^{(n)})$ and $\Pr(y_k^{(n)} = 1, l_t, l_s|\mathbf{x}^{(n)})$ $\forall n, \forall t \in \mathscr{V}, \forall (t,s) \in \mathscr{E}$
  for $k = 1$ to K **do**
    //update unary potentials
    for $t \in \mathscr{V}$ **do**
      $h_{k,m}^t = \arg\min_{h^t} \sum_{n=1}^N \sum_{l_t} [-g_{k,m}^t(\mathbf{x}_n, l_t) - h^t(\mathbf{x}^{(n)}, l_t)]^2$
      $\rho_{k,m}^t = \arg\min_{\rho^t} \sum_{n=1}^N \Psi(y_k^{(n)}, h_{k,m-1}^t(\mathbf{x}^{(n)}, l_t) + \rho^t h_{k,m}^t(\mathbf{x}^{(n)}, l_t))$
      $F_{k,m}^t(\mathbf{x}, l_t) = F_{k,m-1}^t(\mathbf{x}, l_t) + \rho_{k,m}^t h_{k,m}^t(\mathbf{x}, l_t)$
    end for
    //update pairwise potentials
    for $(t,s) \in \mathscr{E}$ **do**
      $h_{k,m}^{t,s} = \arg\min_{h^{t,s}} \sum_{n=1}^N \sum_{l_t, l_s} [-g_{k,m}^{t,s}(\mathbf{x}^{(n)}, l_t, l_s) - h^{t,s}(\mathbf{x}^{(n)}, l_t, l_s)]^2$
      $\rho_{k,m}^{t,s} = \arg\min_{\rho^{t,s}} \sum_{n=1}^N \Psi(y_k^{(n)}, F_{k,m-1}^{t,s}(\mathbf{x}^{(n)}, l_t, l_s) + \rho^{t,s} h_{k,m}^{t,s}(\mathbf{x}^{(n)}, l_t, l_s))$
      $F_{k,m}^{t,s}(\mathbf{x}, l_t, l_s) = F_{k,m-1}^{t,s}(\mathbf{x}, l_t, l_s) + \rho_{k,m}^{t,s} h_{k,m}^{s,t}(\mathbf{x}, l_t, l_s)$
    end for
  end for
end for

not completely reliable, we allow each frame of a tracklet to move around in a fixed range with respect to its initial position from the tracker. This will provide some robustness against imperfections of the tracker. We denote the offset of the frame t in a tracklet using a latent variable $l_t \in \mathscr{L}$. This results in five latent variables per tracklet. We define an offset as $l_t = (dx, dy)$ within a fixed range W, so that $\mathscr{L} = \{(dx, dy) | -W \leq dx, dy \leq W\}$. After applying an offset in a latent variable l_t, we can refine the position of the person in frame t of a tracklet in a video.

We assume these five latent variables to form a chain structure, as showing in Fig. 2. For each frame of a tracklet, we will define unary features that can be used to classify the action of a person. As described below, in our implementation these will be based on optical flow values in the frame. These features will depend on the latent variable for a frame, which provides the offset to refine the position of the person.

In addition, we will define a set of pairwise features in our model that relate variables in adjacent frames of a tracklet. These features will be used to enforce appearance consistency, a tracking constraint, over the values of adjacent offset latent variables l_t and l_{t+1}. These features will be based on colour similarity in our implementation. Since the model forms a chain structure, the marginal distributions in Eqs. 12 and 14 can be computed efficiently via belief propagation.

3.2 Features and Implementation Details

Our model is built upon the optical flow features in [4] and colour histogram features. The optical flow features are for the unary potentials in the model to describe motions, and they have been shown to perform reliably with noisy image sequences for action classification.

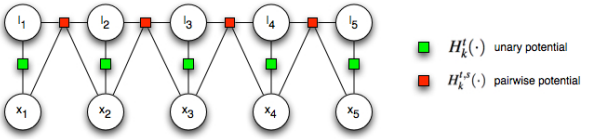


Figure 2: Illustration of the LatentBoost model. Each circle corresponds to a variable, and each square corresponds to a factor in the model. x_1 to x_5 are the variables representing frame 1 to frame 5 in a tracklet, and l_1 to l_5 are the latent variables representing the offset of each frame in a tracklet.

The colour histogram features are used by the pairwise potentials. They impose tracking constraints between two consecutive frames.

Optical flow features: To compute the optical flow features, we use entire frames from the input video instead of just the stabilized human figure in the tracklets in order to obtain absolute human figure motion. The Lucas and Kanade [11] algorithm is employed to compute the optical flow of each frame. Following [4], the optical flow vector field is then split into 4 scalar fields channels corresponding to different flow directions, and blurred. We add another channel corresponding to the motion magnitude which is obtained by computing the L_2 norm of an optical flow vector.

One optical flow feature $f_j(x_t)$ is defined as the motion in a channel c at a location (u,v). Let $\mathscr{U} = \{f_j(x_t) : j = (c,u,v)\}$ be the pool of possible optical flow features. A weak learner h^t used in a unary potential $H_k^t(\cdot)$ picks one of the features $f_j \in \mathscr{U}$:

$$h_j^t(x_t, l_t) = \begin{cases} 1 & \text{if } p_j f_j(x_t + l_t) < p_j \theta_j \\ -1 & \text{otherwise} \end{cases} \tag{15}$$

where the operation $+$ denotes applying the tracker offset in l_t to the position of the person in x_t.

Colour histogram features: The pairwise features in our model are built from comparing colour histograms. We compare the colour histograms in sub-windows of the tracklets to enforce a tracking constraint between adjacent person locations.

One colour histogram feature $f_j(x_t, x_{t+1})$ is defined as the difference between colour histograms in rectangular sub-windows taken from adjacent frames. Let $\mathscr{P} = \{f_j(x_t, x_{t+1})\}$ denote the pool of possible colour histogram features, with j enumerating over a set of image sub-windows. In our experiments we use image sub-windows of fixed size (i.e. 15×15).

A weak learner $h_j^{t,t+1}$ on pairwise potential $H_k^{t,t+1}(\cdot)$ picks one of the features $f_j \in \mathscr{P}$:

$$h_j^{t,t+1}(x_t, x_{t+1}, l_t, l_{t+1}) = \begin{cases} 1 & \text{if } p_j f_j(x_t + l_t, x_{t+1} + l_{t+1}) < p_j \theta_j \\ -1 & \text{otherwise} \end{cases} \tag{16}$$

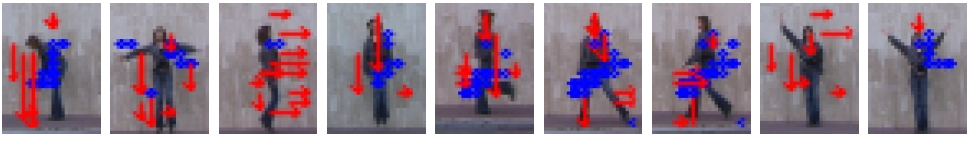
again using the same $+$ notation as above.

The weak learners in Eq. (15) and (16) are decision stumps. Each weak learner consists of a feature f_j, a threshold θ_j, and a parity p_j indicating the direction of the inequality sign.

Line search: The optimal ρ_k^t and $\rho_k^{t,s}$, which are the weights of h_k^t and $h_k^{t,s}$ respectively, are computed by a line search algorithm. However, there is no closed form solution to the optimization problem. Our approach is to approximate them with a single Netwon-Raphson step similar to the one in [7].

	pre-frame	pre-video
LatentBoost	**0.95**	**1.0**
GradientBoost	**0.93**	**0.97**
MMHCRF [13]	0.93	1.0
HCRF [13]	0.90	0.97

Table 1: Comparison of classification accuracy with similar baselines on the Weizmann dataset.



(a) (b) (c) (d) (e) (f) (g) (h) (i)

Figure 3: The representation of the positive optical flow features from the final LatentBoost classifier for the Weizmann dataset. The red arrows are from the first 4 directional motion channels, and the blue points are from the motion magnitude channel. (a) bend, (b) jack, (c) jump, (d) pjump, (e) run, (f) side, (g) walk, (h) wave1, (i) wave2.

4 Experiments

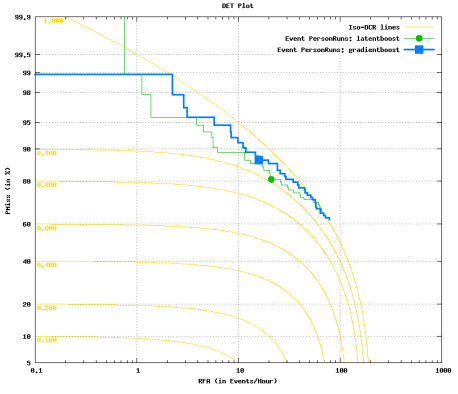
We present experimental results on two publicly available datasets: Weizmann human action dataset [3] and TRECVID surveillance event detection [14]. We show that LatentBoost outperforms a baseline using GradientBoost (Sec. 2.1) in these two datasets. Both LatentBoost and GradientBoost are trained with the same set of features including both optical flow features and colour histogram features. For GradientBoost, we fix the latent variable to be zeros (i.e. $(dx = 0, dy = 0)$), which corresponds to the case of no latent variables.

Weizmann dataset: We first present results on the Weizmann dataset, which is a standard benchmark dataset for action recognition. It consists of 83 videos showing nine different people, each performing nine different actions: *running*, *walking*, *jumping jack*, *jumping forward on two legs*, *jumping in place on two legs*, *galloping sideways*, *waving two hands*, *waving one hand*, and *bending*.

We track and stabilize the figures using the background subtraction masks that come with the dataset. We randomly choose videos of five subjects as the training set, and the videos of the remaining four subjects as the test set. We allow each frame in a tracklet to have an offset of at most 25 pixel locations centred around its initial position in the tracklet.

We compare LatentBoost with GradientBoost and the work in [13]. The comparative results are shown in Table 4. LatentBoost achieves the typical near-perfect results, and slightly outperforms GradientBoost, max-margin hCRF (MMHCRF) and probabilistic hCRF in [13]. A visualization of the positive optical flow features in the final LatentBoost model is shown in Fig. 3. These experiments provide evidence for the effectiveness of our LatentBoost-based action recognition model, albeit on a very simple dataset. Therefore, we focus next on the much more challenging TRECVID dataset.

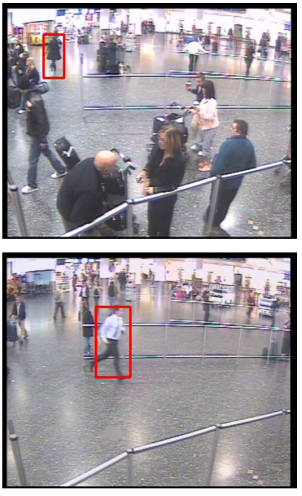
TRECVID dataset: The Weizmann dataset is relatively simple and the peformance on this dataset has already saturated. To further demonstrate our model, we have applied it on a much more challenging dataset from the TRECVID surveillance event detection challenge. The dataset consists of surveillance camera footage acquired at London Gatwick airport. The



(a)

	Minimum DCR
LatentBoost	**0.91**
GradientBoost	0.95

(b)



(c)

Figure 4: Comparison of GradientBoost with our LatentBoost on the TRECVID dataset. (a) The DET curves of the two methods. (b) The minimum DCR scores of the two methods. (c) Examples of running detection in the TRECVID dataset. Note that the DET curve is the lower the better and the minimum DCR score is also the lower the better. Please zoom in for a clear view of the plot.

dataset is large in scale. There are 5 fixed-view surveillance cameras, and each camera has 10 videos, each 2 hours long. The TRECVID chalenge aims to locating the starting/ending frame during which one of the following 7 events occurs: *PersonRuns*, *CellToEar*, *ObjectPut*, *PeopleMeet*, *PeopleSplitUp*, *Embrace*, and *Pointing*.

To test our algorithm, we focus on the *PersonRuns* event and use the videos from camera 5. We follow the same evaluation criteria of TRECVID event detection challenge. Note that TRECVID is very challenging. The *PersonRuns* event is the only event for which state-of-the-art algorithms can achieve reasonable performance (i.e. better than random guessing). Most of the *PersonRuns* events from camera 5 are captured from a common, canonical side view.

We randomly chose 5 videos of camera 5 as the training set, and the remaining videos of camera 5 as the test set. To generate the positive training set, we first obtain the ground truth video clips from the 5 training videos. Each ground truth video clip contains at least one running subject. For each ground truth video clip, we run a tracker to generate all possible tracklets, and then we manually select the tracklets containing a running person (the ground truth provided in TRECVID dataset does not contain spatial localization). For the negative training set, we randomly select not-running tracklets from the rest of the training videos. The final training set consists of 800 running tracklets and 1800 not-running tracklets. Notice that a person who is far from the camera is much smaller than a person who is close to the camera in camera 5. We resize the frames of every tracklet to the same size (i.e. 29×60 pixels), and use the resized tracklets to train both LatentBoost and GradientBoost. Similarly, we also need to normalize the magnitude of optical flow features across all tracklets to the

same size.

For this dataset, we again allow each frame in a tracklet to have an offset of at most 25 pixel locations centred around its initial position given by the tracker. After training the classifiers, we apply them to the test videos. There are 10 hours of test videos in total. The event detection system we use is a standard tracking-and-classification detection system from the TRECVID Workshop 2010 [8]. It has three steps: a pre-processing step to detect and track humans, classification, and a post-processing step for non-maximum suppression. More details on the system can be found in the supplementary material.

The results are shown in Fig. 4 (a) and (b). Detection-Error Tradeoff (DET) curves and minimum Detection Cost Ratio (DCR), a summary statistic, are shown. These are the standard TRECVID evaluation criteria. Again, LatentBoost outperforms GradientBoost, this time on a much challenging and realistic dataset. Examples of running detection are shown in Fig. 4 (c).

5 Conclusion

In this paper, we have presented a novel boosting algorithm with latent variables, called LatentBoost. The algorithm trains a K-class classifier and allows each data point to have a structure of latent variables. Both unary and pairwise potentials in the latent structure are learned by using gradient-descent in function space.

We have demonstrated that the latent boosting algorithm can be used for the task of human action recognition. On the simplistic Weizmann dataset [3], LatentBoost outperforms similar baseline methods: GradientBoost [7], MMHCRF [13], and probabilistic hCRF [13]. On the challenging TRECVID [14] dataset, LatentBoost outperforms GradientBoost [7] in a complex event detection system. LatentBoost opens a new way to solve problems with a structure of latent variables, which can be applied in a variety of applications.

Acknowledgement

This work was supported by grants from the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

[1] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. *NIPS*, 2002.

[2] B. Babenko, Ming-Hsuan Yang, and S. Belongie. Visual Tracking with Online Multiple Instance Learning. *CVPR*, 2009.

[3] Moshe Blank, Lena Gorelick, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. *ICCV*, 2005.

[4] A.A. Efros, A.C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. *ICCV*, 2003.

[5] Alireza Fathi and Greg Mori. Action recognition by learning mid-level motion features. *CVPR*, 2008.

[6] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 2010.

[7] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Technical report, Stanford University*, 1999.

[8] Zhi Feng Huang and Greg Mori. SFU at TRECVid 2010: Surveillance Event Detection. *TRECVid Workshop*, 2010.

[9] Tae-Kyun Kim and Roberto Cipolla. Canonical correlation analysis of video volume tensors for action categorization and detection. *PAMI*, 2009.

[10] Ivan Laptev and Patrick Pérez. Retrieving actions in movies. *ICCV*, 2007.

[11] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. *DARPA Image Understanding Workshop*, 1981.

[12] Ronald Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28:976–990, 2010.

[13] Ariadna Quattoni, Michael Collins, and Trevor Darrell. Conditional random fields for object recognition. *NIPS*, 2004.

[14] Alan F. Smeaton, Paul Over, and Wessel Kraaij. Evaluation campaigns and trecvid. In *MIR '06: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, pages 321–330, New York, NY, USA, 2006. ACM Press. ISBN 1-59593-495-2. doi: http://doi.acm.org/10.1145/1178677.1178722.

[15] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea. Machine recognition of human activities: A survey. *IEEE Transactions on Circuits and Systems for Video Technology*, October 2008.

[16] Paul Viola and Michael Jones. Robust real-time object detection. *2nd Intl. Workshop on Statistical and Computational Theories of Vision*, 2001.

[17] Paul A. Viola, John C. Platt, and Cha Zhang. Multiple instance boosting for object detection. *NIPS*, 2005.

[18] Yang Wang and Greg Mori. Hidden part models for human action recognition: Probabilistic vs. max-margin. *PAMI*, 2010.

[19] Daniel Weinland, Remi Ronfard, and Edmond Boyer. A survey of vision-based methods for action representation, segmentation and recognition. *Computer Vision and Image Understanding*, 2011. to appear.