Computer Vision and Image Understanding 144 (2016) 24–35

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Incremental learning of human activity models from videos☆

Mahmudul Hasan[a,*], Amit K. Roy-Chowdhury[b]

[a] Department of Computer Science and Engineering, University of California, Riverside, United States
[b] Department of Electrical and Computer Engineering, University of California, Riverside, United States

ARTICLE INFO

Article history:
Received 18 December 2014
Accepted 13 October 2015

Keywords:
Incremental learning
Activity recognition
Graphical model

ABSTRACT

Learning human activity models from streaming videos should be a continuous process as new activities arrive over time. However, recent approaches for human activity recognition are usually batch methods, which assume that all the training instances are labeled and present in advance. Among such methods, the exploitation of the inter-relationship between the various objects in the scene (termed as context) has proved extremely promising. Many state-of-the-art approaches learn human activity models continuously but do not exploit the contextual information. In this paper, we propose a novel framework that continuously learns both of the appearance and the context models of complex human activities from streaming videos. We automatically construct a conditional random field (CRF) graphical model to encode the mutual contextual information among the activities and the related object attributes. In order to reduce the amount of manual labeling of the incoming instances, we exploit active learning to select the most informative training instances with respect to both of the appearance and the context models to incrementally update these models. Rigorous experiments on four challenging datasets demonstrate that our framework outperforms state-of-the-art approaches with significantly less amount of manually labeled data.

1. Introduction

Human activity recognition is a challenging and widely studied problem in computer vision. It has many practical applications such as video surveillance, video annotation, video indexing, active gaming, human computer interaction, assisted living for elderly, etc. Even though enormous amount of research has been conducted in this area, it still remains a hard problem due to large intra-class variance among the activities, large variability in spatio-temporal scale, variability of human pose, periodicity of human action, low quality video, clutter, occlusion, etc.

With few exceptions, most of the state-of-the-art approaches [1] to human activity recognition in video are based on one or more of the following four assumptions: (a) It requires an intensive training phase, where every training example is assumed to be available; (b) Every training example is assumed to be labeled; (c) At least one example of every activity class is assumed to be seen beforehand, i.e., no new activity type will arrive after training; (d) A video clip contains only one activity, where the exact spatio-temporal extent of the activity is known. However, these assumptions are too strong and not realistic in many real world scenarios such as streaming and surveillance videos. In these cases, new unlabeled activities are coming continuously and the spatio-temporal extent of these activities are usually unknown in advance.

Recent successes in object and activity recognition take the advantages of the fact that, in nature, objects tend to co-exist with other objects in a particular environment. This is often termed as *context* and plays an important role in human visual system for object recognition [2]. Similarly, most of the human activities in the real world are inter-related and the surroundings of these activities can provide significant visual clue for their recognition. Several research works [3–8] considered the use of context from different perspectives to recognize complex human activities and showed significant performance improvement over the approaches that do not use context. However, these approaches are batch methods that require large amount of manually labeled data and are not able to continuously update their models in order to adapt to the dynamic environment. Even though few research works such as [9–11] learn human activity models incrementally from streaming videos, they do not utilize contextual information, which can lead to superior performance.

Motivated by the above, the main goal of this work is twofold: to classify new unknown activities in streaming videos, and also leverage upon them to continuously improve the existing activity recognition models. In order to achieve this goal, we develop an incremental activity learning framework that will use new activities identified in the incoming video to *incrementally improve* the existing models by

☆ This work was supported in part by ONR grant N00014-15-C-5113 and NSF grant IIS-1316934.

* Corresponding author.
E-mail addresses: mhasa004@ucr.edu, hasaninbuet@yahoo.com (M. Hasan), amitrc@ee.ucr.edu (A.K. Roy-Chowdhury).

http://dx.doi.org/10.1016/j.cviu.2015.10.018
1077-3142/

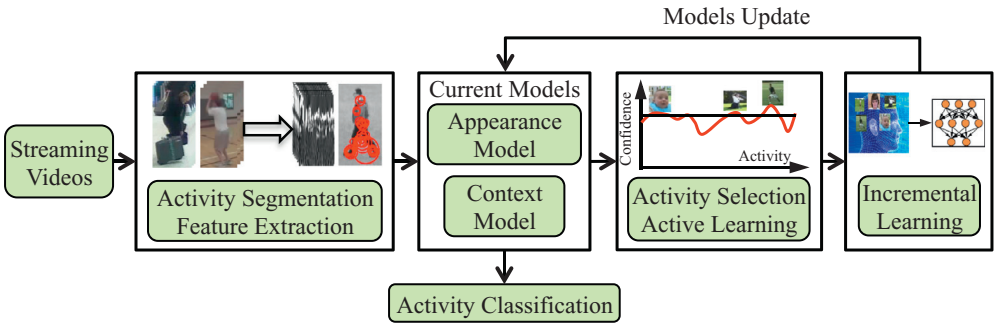

Fig. 1. This figure shows our proposed incremental activity modeling framework, which is comprised of following stages: activity segmentation, feature extraction, appearance and context model learning, activity classification, training set selection by active learning, and model updating by incremental learning with the help of the active learning system.

leveraging relevant machine learning techniques, most notably active learning. The proposed model not only utilizes the appearance features of the individual activity segments but also takes the advantages of interrelationships among the activities in a sequence and their interactions with the objects.

1.1. Overview of the proposed approach

The detailed framework of our proposed incremental activity recognition algorithm is shown in Fig. 1. Since, we do not have any prior information about the spatio-temporal extent of the activities in the continuous video, our approach begins with video segmentation and localization of the activities using a motion segmentation algorithm. Each of the atomic motion segments are considered as the activity segments from which we collect spatio-temporal local feature STIP [12]. These features are widely used in action recognition and achieve satisfactory performance in state-of-the-art challenging datasets. We construct a single feature vector using these local features by exploiting the method described in [13]. Then, we learn a prior model using few labeled training activities in hand. In this work, we propose to use an ensemble of linear Support Vector Machine (SVM) classifiers as the prior model. Note that *we do not assume that the prior model is exhaustive* in terms of covering all activity classes or in modeling the variations within the class. It is only used as a starting point for the incremental learning framework.

We start incremental learning with the above mentioned prior model and update it during each run of incremental training. When a newly segmented activity arrives, we apply the current model to get a tentative label with a confidence score. However, it is not practical and rational to use all of the newly segmented activities as the training examples for the next run of incremental training. This is because it is costly to get a label for all of them from a human annotator, and not all of them posses distinguishing properties for effective update of the current model. We only select a subset of them and rectify the tentative labels by our proposed active learning system. In order to learn the activity model incrementally, we employ an ensemble of linear SVMs. When we have sufficient new training examples labeled by the active learning system, we train a new set of SVM classifiers and consequently, update the current model by adding these new SVM classifiers to the ensemble with appropriate weights.

For the incremental learning with context features, we use a conditional random field (CRF) graphical model in order to represent the interrelationships among the activity segments and the associated object attributes segmented from a video sequence. The nodes of the CRF represent the activities and the object attributes and the edges represent the interrelationships among them. Confidence scores of the activities from the ensemble of SVM classifiers are used as the activity nodes potential, whereas scores obtained from the object detectors are used as the object nodes potentials. Various spatio-temporal relationships such as co-occurrence of activities and objects are used as the edge potentials. We run inference on the CRF in order to obtain the posterior activity labeling with confidence scores. These confidence scores are used in the active learning system consisting of strong and weak teachers to rectify the labels. Hence, these labels are used to update the edge potentials.

1.2. Main contributions

In this work we propose a novel framework to incrementally learn the activity models from streaming videos, which is achieved through an active learning system. The main contributions are as follows -

- We incrementally learn the human activity models with the newly arriving instances using an ensemble of SVM classifiers. It can retain the already learned information and does not require the storage of previously seen examples.
- We reduce the expensive manual labeling of the incoming instances from the video stream using active learning. We achieved similar performances comparing to the state-of-the-arts with less amount of manually labeled data.
- We propose a framework to incrementally learn the context model of the activities and the object attributes that we represent using a CRF.

2. Related works

Activity Recognition. We would like to refer to the paper [1] for a comprehensive review on the sate-of-the-art approaches to human activity recognition. Based on the level of abstraction used to represent an activity, state-of-the-art approaches can be classified into three general categories such as low-level [12], mid-level [10], and high-level [14] feature based methods. However, as discussed in Section 1, most of these state-of-the-art approaches suffer from the inability to model activities in continuous streaming video and unable to take advantages of unseen incoming activities.

Incremental Learning. Incrementally learning from streaming data is a well studied problem in machine learning and a lot of approaches have been proposed in the literature. Among these approaches, ensemble of classifiers [15,16] based methods are most commonly used, where new weak classifiers are trained as new data is available and added to the ensemble. Their outputs are combined using an appropriate combination rule, which is set according to the system's goal.

Context Modeling. Recently, context has been successfully used for human activity recognition. Based on the problem of interest, context may vary. For example, [3] used object and human pose as the context for the activity recognition from single images. Collective or group activities were recognized in [5] and [6] using the context in the group. Spatio-temporal and co-occurrence contexts among the activities and the surrounding objects were used in [7] and [8] for recognizing complex human activities. In [4], Markov random field

were used to predict sports moves and human activities. Apart from these, spatio-temporal graphs [17], AND-OR grammar [18], and HMM [19] have also been used for recognizing complex human activities.

Active Learning. Active learning has been successfully used in speech recognition, information retrieval, and document classification [20]. Some recent works used two stage active learning framework in several computer vision applications such as image segmentation [21], image and object classification [22], unusual event detection [23], action recognition [24], etc. However, unlike most of these methods, our framework does not require the storage of already used training examples and takes the advantage of highly confident decision provided by the current classification model, which in turns reduces the amount of manual labeling.

Incremental Activity Modeling. A few methods have considered incremental activity modeling. A feature tree based incremental action recognition method was proposed in [9], where the feature-tree grows when additional training examples are available. It requires the storage of all training examples in the form of feature tree, which is not feasible for continuous streaming videos because the number of activities could be very large over time. Human track-based incremental activity learning framework was proposed in [10]. It requires annotation of the human body in the initial frame of an action clip, which restricts the variety of application domains possible.

This paper has significant differences with our previous work in [11]. In [11], we proposed a method that incrementally learned human activity models (only the appearance model) when new training examples become available. We did not utilize the interrelationships among the activities and the object attributes (context model) during activity modeling. In this work, we learn both of the appearance and the context model for human activity recognition. We update both of these models incrementally when new training examples become available. These changes improve the performance of our framework significantly for recognizing more complex human activities over time.

3. Incremental learning of individual activity classes

We now provide a detailed overview of our proposed incremental activity modeling framework for the appearance model. We assume that we have a set of activities segmented from a video sequence and we have extracted a set of features $\{\mathbf{x_i} : i = 1, 2, 3, \ldots, n\}$ from these activity segments. Details of activity segmentation and feature extraction are discussed in the experiment section. In this section we mainly focus on learning activity models without using the contextual information or the interactions with the object attributes.

3.1. Activity model

We use an ensemble of multi-class linear Support Vector Machines (SVM) for activity modeling, which can be defined as follows: $\mathcal{H}(\mathbf{x}) = \sum_t \log \frac{1}{\beta_t} h_t(\mathbf{x})$, where h_t is the t^{th} classifier in the ensemble, $\beta_t = \epsilon_t/(1-\epsilon_t)$ is the corresponding weight, and ϵ_t is the normalized error of h_t on the training set. We select this model because it has the ability to learn incrementally without storing the training instances. It can propagate information in terms of learned weak SVM classifiers in the ensemble. Moreover, it can learn new activity classes too. Above mentioned properties are absent in most of the classification frameworks. A detailed mathematical analysis of ensemble of SVM classifiers can be found in [25].

3.2. Incremental activity modeling

We present the detailed incremental activity modeling approach in Algorithm 1, while each of the steps is described in the following subsections.

Algorithm 1: Incremental Activity Modeling.

Data: $\mathcal{V}$: Continuous Streaming Video.
Result: $\mathcal{H}$: Activity Recognition Model, $[(\mathbf{x}_{t_0+i}, y_{t_0+i})|i = 1, \ldots]$: Labeled Activities.
Parameters: Number of SVMs to be trained for batch k, T_k. Active learning parameters δ and ϵ.
Step 0: Learn the prior model $\mathcal{H}_0$ using fewer training data available.
Step 1: Segment the video $\mathcal{V}$ at timestamp $(t_0 + i)$to get an unlabeled activity segment, $\mathbf{x}_i$ (Section 6.2).
Step 2: Apply the current model $\mathcal{H}_k$ on$\mathbf{x}_i$. Based on the condition met, get a label y_i for $\mathbf{x}_i$ (Section 5) and put $(\mathbf{x}_i, y_i)$ in the buffer, $\mathcal{B}_k$.
Step 3: If $\mathcal{B}_k$ contains m training examples, goto step 4 for next incremental learning, otherwise goto step 1.
Step 4: Initialize the distribution for selecting training examples: $\mathbf{w}_1(i) = D(i) = \frac{1}{m}, \forall i = \{1, \ldots, m\}$
Step 5:
for $t = 1$ *to* T_k **do**
  1. Normalize distribution: $\mathbf{D}_t = \mathbf{w}_t / \sum_{i=1}^{m} \mathbf{w}_t(i)$
  2. From $\mathcal{B}_k$, randomly choose 2/3 examples according to $\mathbf{D}_t$. Lets say them Tr_t.
  3. Error: $\epsilon_t = 1$
  4. while $\epsilon_t > 0.5$ **do**
    Train a linear SVM, $h_t : \mathbf{x} \rightarrow y$ using Tr_t.
    Error of h_t on $\mathcal{B}_k$, $\epsilon_t = \sum_i \mathbf{1}[h_t(\mathbf{x}_i) \neq y_i]\mathbf{D}_t(i)$.
  end
  5. Normalized error: $\beta_t = \epsilon_t/(1-\epsilon_t)$
  6. Obtain the composite hypothesis and error:
  $\mathcal{H}_t = \arg\max_{y \in Y} \sum_t \mathbf{1}[h_t(\mathbf{x}_i) = y] \log(1/\beta_t)$
  $E_t = \sum_{i:\mathcal{H}_t(\mathbf{x}_i) \neq y_i} \mathbf{D}_t(i) = \sum_i^m \mathbf{D}_t(i)\mathbf{1}[\mathcal{H}_t(\mathbf{x}_i) \neq y_i]$
  7. If $E_t > 0.5$, set $t = t - 1$, discard $\mathcal{H}_t$ and goto line 2.
  8. Normalized composite error, $B_t = E_t/(1-E_t)$
  9. Update the distribution: $\mathbf{w}_{t+1}(i) = \mathbf{w}_t(i) \times B_t^{1-|H_t(\mathbf{x}_i) \neq y_i|}$
end
Step 6: Final decision on an unlabeled activity,
$\mathcal{H}(\mathbf{x}) = \arg\max_{y \in Y} \sum_k \sum_t \mathbf{1}[\mathcal{H}_t(\mathbf{x}) = y] \log \frac{1}{B_t}$
Step 7: Empty the buffer. Goto step 1 for incremental learning with the next batch.

Learning Prior Model. At first, we learn a prior model $\mathcal{H}_0$ using very few labeled training examples. In this work, we use an ensemble of SVM classifiers as described in Section 3.1 as the prior model. Prior model learning stage is neither intensive like other state-of-the-art approaches, nor exhaustive in terms of covering all activity classes or in modeling the variation within the class. It is used as the starting point for the incremental learning.

Activity Segmentation and Active Learning. Let us consider that we have a video stream $\mathcal{V}$, starting at timestep t_0. As time progress, new activities are arriving from the streaming video. We segment an activity $\mathbf{x}_i$ at time $t_0 + i$ and collect features using the methods described in Section 6.2. We apply the current model $\mathcal{H}_k$ on the unlabeled activity $\mathbf{x}_i$ to get a label y_i using the active learning system described in Section 5. We store the labeled activity $(\mathbf{x}_i, y_i)$ temporarily in a buffer $\mathcal{B}_k$, where k stands for k^{th} incremental training step.

Incremental Learning. As in [15], our incremental learning approach is based on the following intuition: each new classifier added to the ensemble is trained using a set of examples drawn according to a distribution, which ensures that examples that are misclassified by the current ensemble have a high probability of being sampled in

the next round. Weight update mechanism of the individual SVMs remains same as in [15], which we describe below. Let us consider that inputs to the k^{th} incremental learning stage are a sequence of training examples, $\mathcal{B}_k = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$, where $\mathbf{x}_i$s are the training instances and y_is are the corresponding labels. Let us assume that a weak baseline SVM classifier model is known, and let T_k be the number of classifiers to be learned at the k^{th} stage.

- At first, we initialize the distribution $\mathcal{D}_1$ of the training instances in $\mathcal{B}_k$ by assigning equal probability to all of them so that each of the instances has equal chance to be selected to train a weak SVM. We obtain this by diving one by the number of training instances.
- We now train t^{th} weak SVM h_t. We call them weak because they are not required to be highly efficient. But they have to perform more than the average. We use parameterless linear SVM as the weak SVM classifier. In order to train a weak SVM, we select a subset of the training instances Tr_t from $\mathcal{B}_k$ according to the current distribution $\mathcal{D}_t$.
- We train h_t using these randomly selected subset of the training instances. If the error (ϵ_t) associated with this new classifier on the training set is higher than a threshold 0.5, we reject the classifier, otherwise we add it to the ensemble $\mathcal{H}_t$. In case of rejection, we start training h_t again with a newly selected random subset of the training instances.
- After the inclusion of h_t to the ensemble, we compute the error E_t of the ensemble $\mathcal{H}_t$ on the training data. If the error associated with this updated ensemble $\mathcal{H}_t$ is higher than a threshold 0.5, the new update will be rejected and training of h_t starts all over again with new Tr_t.
- If everything goes fine up to this point and we have a new weak SVM in the ensemble, we update the training data distribution so that in the next round examples for which errors occurred get higher priority to be selected as the training instance.

4. Incremental learning of contextual relationships

We model the inter-relationships among the activity instances and the object attributes using a CRF graphical model. An illustrative example of the CRF with four activity nodes is shown in Fig. 2. It is an undirected graph $G = (V, E)$, with a set of nodes $V = \{A, C, X, Z\}$, and a set of edges $E = \{A-A, A-C, A-X, C-Z\}$, where As are the activity nodes, Cs are the object attributes, (i.e. context features), $\mathcal{P}$ and $\mathcal{D}$ are the activity classifier and the objects detectors respectively, and X and Z are the observed visual features. $\mathcal{P}$ is used for computing prior node potentials and $\mathcal{D}$ is used to construct context features. As are the hidden nodes, whereas Cs, Xs, and Zs are the observed nodes (shadowed). We are interested in computing the posterior of the A nodes. Red edges among the A and C nodes represent spatio-temporal relationship. The connections between As and Cs are fixed but we automatically determine the connectivity among the As online along with their potentials. The overall potential function (Φ) of the CRF is shown in Eqn. (1), where, ϕs and ψs are node and edge potentials.

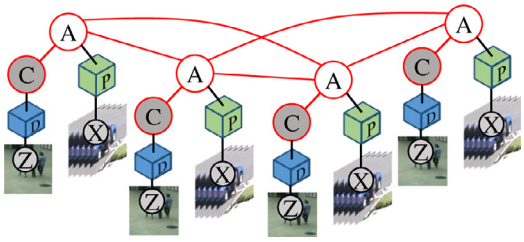

Fig. 2. Illustrative example of a CRF for encoding the contextual information. Please see the text in Section 4 for details.

We define the potential functions in the following subsections.

$$\Phi = \prod_{\substack{a_i \in A, c_i \in C \\ x_i \in X, z_i \in Z}} \phi(a_i, x_i)\phi(c_i, z_i) \prod_{\substack{a_i, a_j \in A \\ c_i \in C}} \psi(a_i, a_j)\psi(a_i, c_i) \tag{1}$$

$$\phi(a_i, x_i) = p(a_i | x_i, \mathcal{P}) \tag{2}$$

$$\phi(c_i, z_i) = \phi(c_i^1, z_i) \odot \phi(c_i^2, z_i) \tag{3}$$

$$\phi(c_i^1, z_i) = p(c_i^1 | z_i, \mathcal{D}) \tag{4}$$

$$\phi(c_i^2, z_i) = \text{bin}(c_i^2)\mathcal{N}(c_i^2, \mu_{c^2}, \sigma_{c^2}) \tag{5}$$

$$\psi(a_i, a_j) = F_a(a_i, a_j)\mathcal{N}(\|t_{a_i} - t_{a_j}\|^2, \mu_t, \sigma_t)N(\|s_{a_i} - s_{a_j}\|^2, \mu_s, \sigma_s) \tag{6}$$

$$\psi(a_i, c_i) = \psi(a_i, c_i^1) \otimes \psi(a_i, c_i^2) \tag{7}$$

$$\psi(a_i, c_i^1) = F_{c^1}(a_i, c_i^1)\mathcal{N}(\|s_{a_i} - s_{c_i^1}\|^2, \mu_{c^1}, \sigma_{c^1}) \tag{8}$$

$$\psi(a_i, c_i^2) = \sum_{a \in A} \text{bin}(c_i^2)\mathcal{I}(a = a_i)^T \mathcal{N}(c_i^2, \mu_{c^2}, \sigma_{c^2}) \tag{9}$$

4.1. Node potentials

Activity potentials, $\phi(a_i, x_i)$. These potentials correspond to the A nodes of the CRF in Fig. 2. They are related to the low level motion features that describe inherent characteristics of the activities. We extract low level features x_i from the activity segments a_i and train a baseline multi-class classifier $\mathcal{H}$ (Section 3.1). Classification scores of the candidate activity segments a_is generated by $\mathcal{H}$ are then used as the node potentials as defined in Eq. (2). It is a vector of length c (Number of activity classes).

Context potential, $\phi(c_i, z_i)$. These potentials correspond to the C nodes of the CRF in Fig. 2, which are scene level features and object attributes related to the activity of interest. They are not associated with motion features of the activities but may provide important and distinctive visual clues. For example, presence of a car may distinguish *unloading a vehicle* activity from *entering a facility* activity. These context features can be learned automatically or assembled in an ad hoc basis. In this work, we employ a semi-automatic technique. We use a number of detectors on the image observation Zs in the activity segment in order to construct the context feature. Number and nature of such features may vary for different application domains. We concatenate them as in Eq. (3), where $\odot$ is the concatenation operation and k is the number of context feature. We use two context features - objects ($\phi(c_i^1, z_i)$) and person ($\phi(c_i^2, z_i)$) attributes as defined in Eqs. (4) and (5), where o_i is the object class vector, $\|L_1 - L_2\|$ is the distance covered by a person in the activity region, bin$(\cdot)$ is a binning function as in [26], and μ_{c2} and σ_{c2} are the mean and variance of the covered distances.

4.2. Edge potentials

Activity-Activity potential, $\psi(a_i, a_j)$. This potential models the connectivity among the activities of a sequence. We assume that activities which are within a spatio-temporal distance are related to each other. This potential has three components - association, spatial, and temporal components. The association component is the co-occurrence frequencies of the activities. The spatial (temporal) component models the probability of an activity belonging to a particular category given its spatial (temporal) distance from its neighbors. $\psi(a_i, a_j)$ is defined in Eq. (6), where $a_i, a_j \in A$, $F_a(a_i, a_j)$ is the co-occurrence frequency between the activities a_i and a_j, s_{a_i}, s_{a_j}, t_{a_i}, and t_{a_j} are the spatial and temporal locations of the activities, and

μ_{c^1}, σ_{c^1}, μ_{c^2}, and σ_{c^2} are the parameters of the Gaussian distribution of relative spatial and temporal positions of the activities, given their categories.

Activity-Context potential, $\psi(a_i, c_j)$. This potential function models the relationship among the activities and the context features (Eq. (3)). It corresponds to $A-C$ link in Fig. 2. This potential is defined in Eqs. (7)–(9). $\psi(a_i, c_i^1)$ models the relationship between the activity and the object attribute and $\psi(a_i, c_i^2)$ models the relationship between the activity and the person attribute. Operator $\otimes$ performs horizontal concatenation of matrices.

More details on the context features can be found in the experiment section. We also illustrate a simplified CRF with node and edge potentials in Fig. 5 for the better understanding of the model.

4.3. Structure learning

A part of the structure of our proposed graphical model (Fig. 2) is predefined. For example, we assume links among the nodes A and C if the involved person or the objects are detected by the detector $\mathcal{D}$. However, we learn the connections among the nodes in A online because it is hard to predict the number of activities and all the activities might not be related to each other. Recent approaches for learning the structure are hill climbing structure search [3] and iterative approach [7]. However, these approaches are not designed for learning continuously from the streaming video. In this work, we utilize a max-margin learning framework to determine the links among the nodes in A. We start with all the nodes in A to be connected to each other. Then we apply two thresholds - spatial and temporal - on the links. We keep the links whose spatial and temporal distances are below these thresholds, otherwise we delete the links. We learn these two thresholds as follows.

Suppose, we have a set of training activities $\{(a_i, t_i, s_i) : i = 1 \dots m\}$ and we know the pairwise relatedness of these activities. The goal is to learn a function $f_r(d) = w^T d$, that satisfies following constraints,

$$\begin{aligned} f_r(d_{ij}) &= +1, && \forall \text{ related } a_i \text{ and } a_j, \\ f_r(d_{ij}) &= -1, && \text{Otherwise.} \end{aligned} \tag{10}$$

$d_{ij} = [\text{abs}(t_i - t_j), \|s_i - s_j\|]$. We can formulate this problem as a traditional max-margin learning problem [3]. Solution to this problem will provide us a function to determine the existence of link between two unknown activities.

4.4. Inference

Inference in a graphical model is the process of computing the marginal probabilities of the hidden variables given the observed variables. We choose belief propagation (BP) message passing algorithm for performing inference on CRF. BP does not provide guarantee to convergence to true marginals for a graph with loops but it has proven excellent empirical performance [27]. Its local message passing is consistent with the contextual relationship we model among the nodes.

At each iteration, belief of the nodes are updated based on the messages received from their neighbors. Consider a node $a_i \in V$ with a neighborhood $N(a_i)$. The message sent by a_i to its neighbors can be written as,

$$m_{a_i,a_j}(a_j) = \alpha \int_{a_i} \psi(a_i, a_j)\phi(a_i, x_i) \prod_{a_k \in N(a_i)} m_{a_k,a_i}(a_i) da_i.$$

The marginal distribution of each node a_i is estimated as,

$$p'(a_i) = \alpha\phi(a_i, x_i) \prod_{a_j \in N(a_i)} m_{a_j,a_i}(a_i).$$

The class label which has the highest marginal probability is the actual class label.

4.5. Updating context model

Updating the context model is actually recomputing the parameters of the Eqs. (4)–(5)–(6)–(8), and (9). The parameters are mainly co-occurrence frequencies and means and variances of the Gaussian distributions. The parameters of the Gaussians can be updated using the method in [28], wheres the co-occurance frequency matrices can be updated as follows, $F_{ij} = F_{ij} + \text{sum}([(L = i).(L = j)^T] .* Adj)$, where, $i, j = \{1, \dots, c\}$, L is the set of labels of the instances obtained from the active learning system, Adj is the adjacency matrix of the CRF G of size $|L| \times |L|$, sum(.) is the sum of the elements in the matrix, and $.*$ is the element wise matrix multiplication.

5. Active learning and teacher selection

Previously, we described the appearance and the context models and the approach we use to update them incrementally with an assumption that we have the labels of the incoming instances. However, in a streaming video scenario incoming instances are unlabeled. Now, we describe our active learning system where we carefully select the most useful instances to be labeled by a human annotator. The main goal is to reduce the amount of expensive manual labeling while retaining the same level of performance similar to the state-of-the-arts.

According to [20], active learning can achieve greater learning accuracy with fewer training labels if the learner is allowed to choose the training data from which it learns. An active learner usually poses queries in the form of unlabeled training data instances to be labeled by an oracle. However, based on the type of teacher (oracle) available, the active learning system can be classified into two broad categories: strong teacher and weak teacher. Strong teachers are assumed to give correct and unambiguous class labels. Most, but not all, strong teachers are humans, which are assumed to have a significant cost. On the other hand, weak teachers generally provide more tentative labels. Most, but not all, weak teachers are assumed to be classification algorithms that make errors but perform above the accuracy of random guess [29]. Our proposed framework provides the opportunity to take advantages of both kind of teachers.

Active learning works within two common schemes: pool-based sampling and stream-based sampling [20]. In our proposed framework, we take the advantages of stream-based sampling, where unlabeled examples are presented one at a time and the learner must decide whether or not it is worth to invoke a teacher to label the example. Now, the following questions remain: When we should ask a teacher? Which teacher to invoke? And what action should we perform in response?

Teacher Selection: Details of the active learning mechanism are illustrated in Fig. 3 using a flowchart. Whenever an unlabeled activity is presented to the system, the current activity recognition model is applied on the activity, which generates a tentative decision $\mathcal{H}(\mathbf{x})$, with a confidence score $\mathcal{S}(\mathbf{x})$. Let the second highest confidence score be $\mathcal{G}(\mathbf{x})$. $\mathcal{S}(\mathbf{x})$ and $\mathcal{G}(\mathbf{x})$ are defined as follows,

$$\mathcal{S}(\mathbf{x}) = \max_{y \in Y} \sum_k \sum_{t \in T_k} \mathbf{1}[\mathcal{H}_t(\mathbf{x}) = y] \log \frac{1}{B_t} \tag{11}$$

$$\mathcal{G}(\mathbf{x}) = \max_{y \in (Y - \mathcal{H}(\mathbf{x}))} \sum_k \sum_{t \in T_k} \mathbf{1}[\mathcal{H}_t(\mathbf{x}) = y] \log \frac{1}{B_t}, \tag{12}$$

where, $\mathbf{1}[.]$ is the indicator function, $\mathbf{x} \in \mathbf{R}^n$ is the input activity, $y \in \{1, \dots, Y\}$ are class labels, $\mathcal{H}_t(\mathbf{x})$ is a classifier, and $\log(1/B_t)$ is the corresponding weight. We invoke the weak teacher when the tentative decision $\mathcal{H}(\mathbf{x})$ has sufficiently large confidence score. That means, if $\mathcal{S}(\mathbf{x})$ is greater than a threshold δ, the unlabeled activity is labeled using the label $\mathcal{H}(\mathbf{x})$ from the current model. Else if, $|\mathcal{S}(\mathbf{x}) - \mathcal{G}(\mathbf{x})|$ is less than a threshold ϵ, the current model is not confident enough to decide about the label. This example lies near the decision boundary and possesses valuable information. In this case, the system invokes

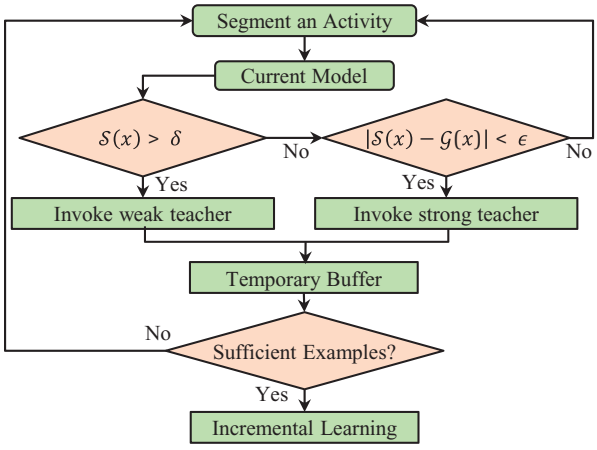

Fig. 3. Flowchart of selecting examples for incremental learning and how to get the correct labels of these examples through active learning. $\mathcal{S}(\mathbf{x})$ and $\mathcal{G}(\mathbf{x})$ are defined in Section 5.

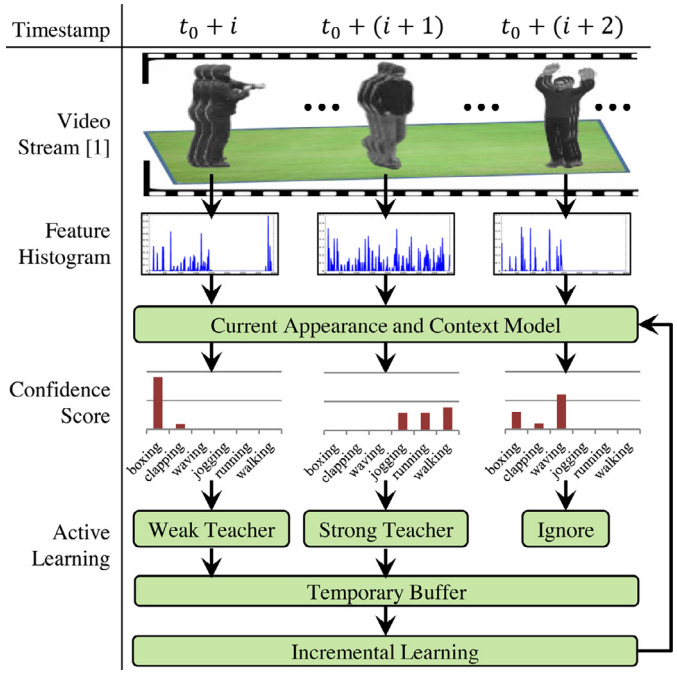

Fig. 4. A sample run of our proposed incremental activity learning framework. After segmenting an activity from the video stream, we generate features and obtain a tentative label with a confidence score from the current model. Our active learning system analyzes the score and obtains the correct label for the activity by invoking a teacher. We temporarily store this new activity with the label for the next incremental learning step.

the strong teacher and obtains the label. Otherwise, the unlabeled activity is not used for incremental learning. When the system has accomplished the task of labeling the unlabeled activity, new activity $\mathbf{x}$ with label y is stored in a buffer temporarily. Choice of the parameters δ and ϵ are domain dependent and can be updated regularly based on system's performance. If the current model performs better on the unseen validation data, these parameters can be set such that the costly strong teacher is invoked rarely during training. Sensitivity analysis of these two parameters are provide in Section 6.

A sample run of our incremental learning framework on KTH dataset using STIP feature is illustrated in Fig. 4. An activity is segmented at timestamp $t_0 + i$, which is followed by feature generation. New activity is labeled as "boxing" by the current model with a very high confidence score that leads the system to invoke the weak teacher. At timestamp $t_0 + (i+1)$, current model labeled another new activity as "walking" with a lower confidence score, which leads the system to invoke the strong teacher. At timestamp $t_0 + (i+2)$, the segmented activity is labeled as "waving", which is eventually ignored by the active learning system because the score is neither confident enough nor it is close to the decision boundary. Activities in the first two cases are stored temporarily in a buffer to be used as the training examples for the next incremental learning step.

6. Experiments

We perform experiments on four challenging datasets to evaluate and compare the performances of our framework. These datasets are KTH [30], UCF11 [31], VIRAT [32], and UCLA-Office [33]. In the first two datasets - KTH and UCF11, activities are temporally segmented, it means that each video segment contains only one activity, whereas in VIRAT and UCLA-Office datasets video sequences are long and contain more than one activities. That's why, for the last two datasets, we use a video segmentation algorithm as described below to segment the activities from the long video sequences. Since the activities in KTH and UCF11 are already temporally segmented, they do not naturally posses any activity–activity or activity-object contextual information. We improvise context features for these datasets in order to evaluate our context model. On the other hand, activities in VIRAT and UCLA-Office naturally posses contextual information. We apply a set of detectors in the activity regions in order to construct the context features. We describe them in details later in this section. We now briefly describe the four datasets - KTH, UCF11, VIRAT, and UCLA-Office as follows.

6.1. Human activity datasets

KTH Human Action Dataset: There are six action classes in KTH [30] dataset such as boxing, handclapping, handwaving, jogging, running, and walking. These actions are performed by 25 individual subjects in four different scenarios - outdoors, outdoors with scale variation, outdoors with different clothes, and indoors with lighting variation. There are totally 599 video clips with the resolution of 160 × 120 pixels. As mentioned above activities are temporally segmented.

UCF11 Human Action Dataset: The second experiment is performed on more challenging UCF11 dataset [31]. There are eleven different action classes in this dataset such as basketball, biking, diving, golf swing, horse riding, soccer juggling, swing, tennis swing, trampoline jumping, volleyball spiking, walking, etc. Each action is performed by 25 different subjects under challenging scenarios and illumination conditions. Videos were collected from Youtube where subjects perform various sports activities in the wild. There are about 1600 video clips with the resolution of 320 × 240 pixels.

VIRAT dataset: It is a challenging wide area human activity dataset with 11 types of activities such as person entering vehicle, person exiting vehicle, person opening trunk, person closing trunk, person loading vehicle, person un-loading vehicle, person carrying an object, person gesturing, person running, person entering, and person exiting a facility. The types of the objects associated with the activities are bag, object, vehicle, bike, and person. These activities are subjected to high amount of occlusion and clutter. This dataset has 11 surveillance video scenes, which are fragmented into 329 sequences. We use first 170 sequences as the training examples and rest of them as the testing.

UCLA Office dataset: The UCLA office dataset consists of indoor and outdoor activities involving one or two persons interacting with each other or objects. We conduct experiment on three office scenes of around 35 minutes that contains around 136 activities. We use half of the activity instances as the training example to learn the models continuously and the rest of them are used for testing. There are ten activity classes such as enter room, exit room, sit down, stand up,

work on laptop, work on paper, throw trash, pour drink, pick phone, and place phone down. The object classes are laptop, water dispenser, phone, and paper.

6.2. Dataset preprocessing

Activity Segmentation: We use an adaptive background subtraction based algorithm [34] to locate motion regions in the continuous video. Inside these motion regions, moving persons are detected by [35]. These detections are used to initialize the tracking method developed in [36] to obtain trajectories of the moving persons. Spatio-temporal interest points (STIP) [12] are collected only for these motion regions. Each motion region is segmented into activity segments using the motion segmentation based on the method in [37] with STIP histograms as the model observation. We apply this method to segment activities in VIRAT and UCLA-Office datasets, which consist of long video sequences. In KTH and UCF11 activities are already temporally segmented. Moreover, we do not perform any spatial segmentation in these datasets.

Appearance Feature Extraction: We use STIP [12] as the basic local features in our experiments. STIP is a spatio-temporal local feature and widely used for representing human actions in video. Any other local features can also be used. After collecting STIP features, we use a spatio-temporal pyramid based pooling technique [13] in order to obtain a fixed length feature representation for each activity segment. This representation can effectively collect local information and also preserve the global structure of the activities.

Context Feature Extraction: As mentioned in Sections 4.1 and 4.2, the CRF has two types of nodes - activity nodes and context nodes. We assign the confidence scores from the ensemble of SVMs as the node potential. On the other hand, we construct the context node potentials as described by Eqs. (4) and (5) by analyzing the activity region. We train a set of object detectors based on HOG features and SVM classifier to detect objects of interest in the region. Detection results of the classifiers are used to construct the context features. We trained five object detectors for VIRAT dataset and four object detectors UCLA-Office dataset. For KTH and UCF11 datasets, we only use activity-activity context. Here, we assume that similar types of activity can influence each other. For example, boxing, handclapping, and handwaving activities of KTH dataset are similar and they can influence each other. By influence we mean correct recognition of one of these activities can help to obtain better recognition of another similar activity. We do not use activity-object context for KTH and UCF11 datasets, because KTH activities do not have interaction with objects. In UCF11, each activity is associated with an unique object class. Use of object context in UCF11 would overfit the model.

We provide an illustrative example of a simplified version of the CRF for two activities in Fig. 5. It also shows the corresponding node and edge potentials. Suppose, we have six activity classes. So the node potential is a vector of size 6×1 and the activity–activity edge potentials is a 2d matrix of size 6×6. Suppose four types of objects are associated with the activities and we have a very simple context model.

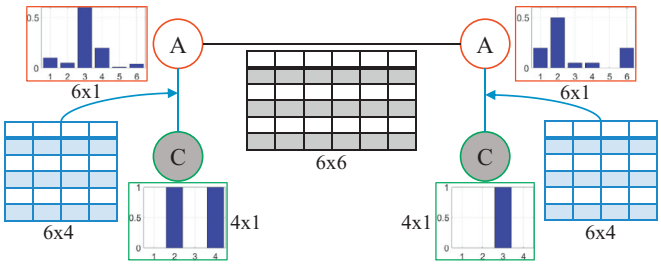

Fig. 5. An illustrative example of a simplified version of the CRF for two activities with potentials.

So the context feature is a vector of size 4×1, where each position indicates the presence of an object. Activity-object edge potential is a 2d matrix of size 6×4 that represent the co-occurrence frequencies of the activities and the objects. Now, if we run belief propagation algorithm on this CRF, it will give us the marginal probabilities of the activities. In this work, we incrementally update the node and edge potentials so that they can perform better with time on the unknown instances.

6.3. Objective and experiment setup

The main objective of the experiments is to analyze how well our framework incrementally learns the activity model with unlabeled data and to compare the performances against state-of-the-art batch and incremental methods. We abide by the following rules during all experiments -

- Due to the random selection of examples during training of SVM classifiers, each run of incremental learning on same dataset and features shows significant variance in accuracy. In order to get rid of this randomness, we average our results over multiple runs containing different random orders in which the data is presented.
- For splitting training and test data, we perform five fold cross validation and then report the mean results over these folds.

We report our results in similar looking plots later in this section. Y-axis of the plots represents the accuracy. It is normalized between zero to one. This accuracy is computed over the test set by dividing the number of correct classification by the total number of test instances. X-axis represents the percentage of training instances the framework has seen so far. For example, 0.6 means that the framework has seen sixty percent of the total training instances. The reported accuracy for 0.6 is the accuracy of the models trained with sixty percent of the total training instances.

6.4. Comparison with other methods

We compare performances of the three variants of our framework against the state-of-the-art methods as illustrated in Fig. 6(a), (b), (c), and (d) for VIRAT, UCLA-Office, KTH, and UCF11 datasets respectively. These three variants are No-Context, Context-A, and Context-AO. As the name implied, No-Context does not use any context information. The accuracies are solely based on the appearance model without any influence of other activities. Context-A uses only activity-activity contextual information, where nearby activities can influence each other for better recognition. Context-AO does not only use the activity-activity context but also take the advantages of activity-object interactions. Here, associated objects can influence for the better recognition of the corresponding activities. We compare our results against following state-of-the-art methods - structural SVM (SSVM) [8], sum product network (SPN) [38], spatio-temporal scene structure (STSS) [7], incremental activity modeling (IAM) [11], continuous learning with deep nets (CLDN) [13], bag of word (BOW) [8], incremental feature tree (IFT) [9], snippets (SNP) [10], incremental gaussian process (IGP) [24], recognizing realistic action (RRA) [31], and semantic visual vocabulary (SVV) [39]. All the variants of our framework utilize both of the weak and the strong teachers. Among all of the instance we manually label about fifty percent of them to achieve these results. Table 1 shows the numeric comparison with other methods. We analyze the characteristics of the plots in Fig. 6 as follows -

- Performances on all of the datasets increase asymptotically as the framework see more and more training instances. When the framework is finished seeing all of the incoming instances, performances are already better than state-of-the-art approaches. But it uses only about fifty percent of the labeled training instances.

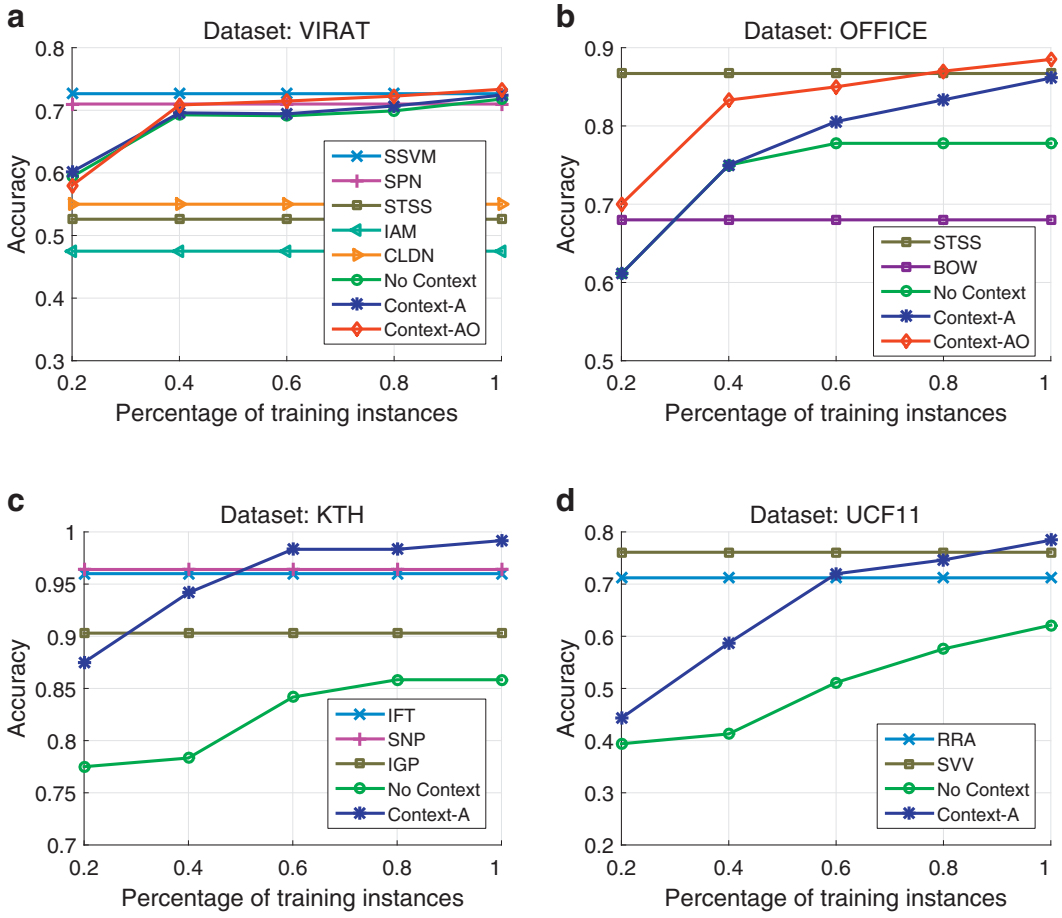

Fig. 6. Performances of the three variants of our framework are compared against the state-of-the-art methods. Please see the texts in Section 6.4 for more details. Plots are best viewable in color.

Table 1
Comparison of our results against state-of-the-art batch and incremental methods.

Datasets	Our methods	State-of-the-art methods
KTH	99.5%(Context-A)	92.1% (HoF) [40], 96.3% [24], 93.9% (ICA) [41], 91.0% [11], 96.4% [13], 96.0% [9], 90.3% [10]
UCF11	78.2% (Context-A)	71.2%[31], 76.1% [39]
VIRAT	73.6% (Context-AO)	47.0% [11], 54.2% [13] 73.5% [8] 72.0% [38]
UCLA-Office	87.7% (Context-AO)	86.7% [7]

- One of the major advantages of this framework is that it does not require to store previous training instances. After incremental learning with a new batch of training instances, it discards them. The framework retain information in terms of ensemble of SVM classifiers.
- Performances of Context-AO is much better than No-Context and Context-A, whereas performances of Context-A is better than No-Context as expected. Because they use more contextual information.

6.5. Effect of teacher selection

As discussed in Section 5, our framework takes the advantages of two kinds of teachers - strong and weak teachers. We select an instance based on its informativeness and send to the strong teacher for labeling, which is basically an expensive human annotator. Weak teacher obtains the labels from the existing classifier, which does not require any cost. Selection of these teachers has different impact on the performances of the framework as illustrated in Fig. 7(a), (b), (c), and (d) for VIRAT, UCLA-Office, KTH, and UCF11 datasets respectively. Our analysis of these results are as follows -

- All-Manual is the plot where we do not have any active learning system and hence, all of the instances are manually labeled. It is the most expensive way of learning. Strong+Weak-Teacher is the plot where we use the active learning system with both of the strong and the weak teachers. Here we only manually label the instances which are informative for the incremental learning. As name implied, Strong-Teacher and Weak-Teacher plots only use the strong or the weak teachers respectively. Weak-Teacher plot does not use any manually labeled instances.
- The plots show that All-Manual performs better than other methods. Because it uses all of the incoming instances for incremental learning and labels all of them manually. The performances of Strong+Weak-Teacher is very similar to the All-Manual. This proves the robustness of the framework in learning incrementally because it manually labels only fifty percent of the incoming data.
- Strong-Teacher also performs almost similarly to the Strong+Weak-Teacher. It demonstrates that the weak labels have little effect on incremental learning.
- The performances of Weak-Teacher is the worst comparing to other methods because it only uses the labels produced by the classifier. Performance diverges with time because of the noisy labels.

6.6. Performance evaluation on individual activities

Fig. 8 shows how incremental learning affects the classification probability scores of some test instances as time goes on. In

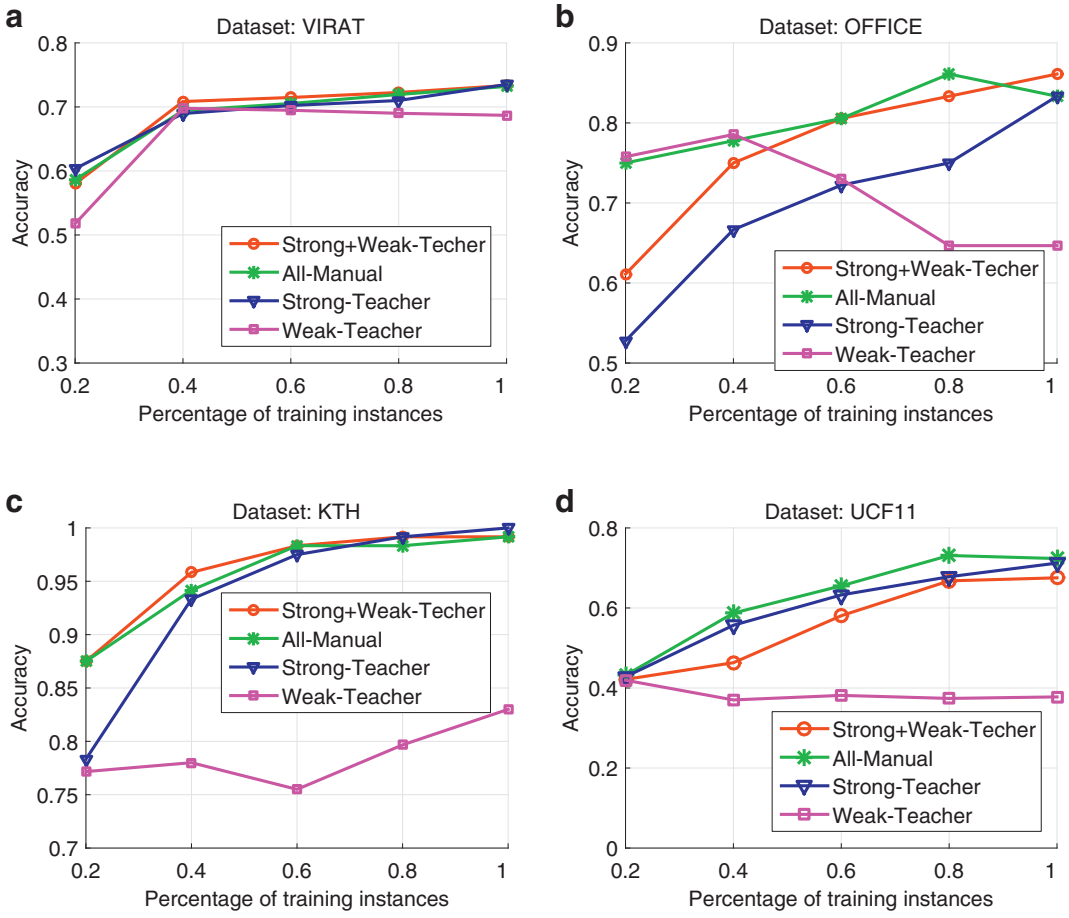

Fig. 7. Effect of different teachers in incremental active learning. Please see the text in Section 6.5 for more details. Plots are best viewable in color.

general with some exceptions, an activity may be misclassified at the beginning but when the models see more and more training instances it becomes more distinctive and correct the labels with higher probability scores. Fig. 8 shows some example activities of four datasets and corresponding probability distribution in different time points (the blue line). The red shade in a plot shows the time points where the corresponding instance was misclassified. GT is the ground truth label and PD is the predicted label of an instance.

6.7. Comparison with other active learning techniques

We perform some experiments in order to compare our active learning system with one other related existing active learning technique (Entropy) [20] and random sampling. Entropy selects an unlabeled instance to be labeled by a human annotator based on the uncertainty measure known as the entropy of the distribution. It does not take the advantage of weak teacher. Random selects an instance randomly without taking care of any informativeness measure. These plots are shown in Fig. 9(a) and (b) for VIRAT and UCLA-Office datasets respectively. Our active learning system consists of strong and weak teacher performs better than other two methods. The margin of difference is smaller for VIRAT but larger for UCLA-Office dataset.

6.8. Manual labeling comparison

Table 2 shows the amount of manual labeling used by different methods to obtain the results in Fig. 7. All-Manual labels all the training instances. Strong-Teacher manually labels forty percents of the training instances to generate results in Fig. 7. Weak-Teacher does not use any manually labeled data during incremental training but it need some labels during initial model learning. The most efficient method is the Strong+Weak teacher.

6.9. Parameter sensitivity

Fig. 10 (a) and (b) shows the sensitivity analysis of the parameter T_k on VIRAT and KTH datasets respectively. Initially, the higher values of T_k perform better than the lower values of T_k, but this performance gap reduces significantly when the framework sees more and more data. Fig. 11(a) and (b) shows the effect of the different amount of manual labeling on the performance of our framework for VIRAT and KTH datasets respectively. For both of the datasets, our framework begins to achieve the best performance with about fifty percent of the manually labeled data. The plot All-Manual-Labeling is a flat line. It represents accuracy of the method that manually labels all of the training instances. Each of the accuracy plotted here is the final accuracy of the Strong+Weak teacher method with respective amount of manual labeling in the x-axis. Fig. 12(a) and (b) shows the sensitivity analysis of the weak teacher selection parameter δ for VIRAT and KTH datasets respectively. For VIRAT dataset, accuracy of the framework reduces for the lower values of δ, because it tends to select noisy labels for the training instances. However, for KTH dataset margin of the performance reduction is not significant because activities are more distinctive and most of them are classified with very high probability score. As a result, selecting instances with $\delta = 0.5$ returns almost same instances with $\delta = 0.9$.

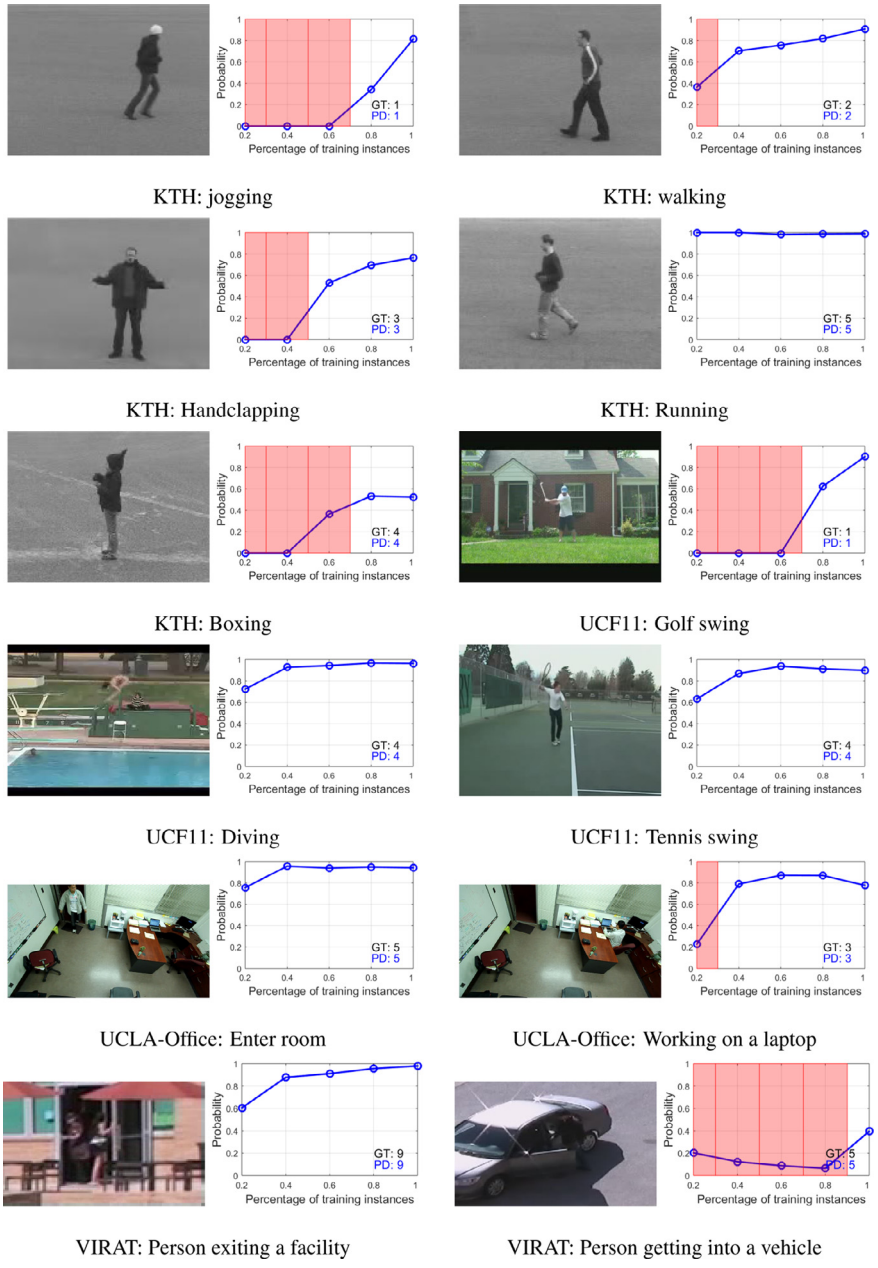

Fig. 8. This figure shows the performance of the proposed incremental activity modeling framework on individual test action clips of different datasets. Plots are best viewable in color.

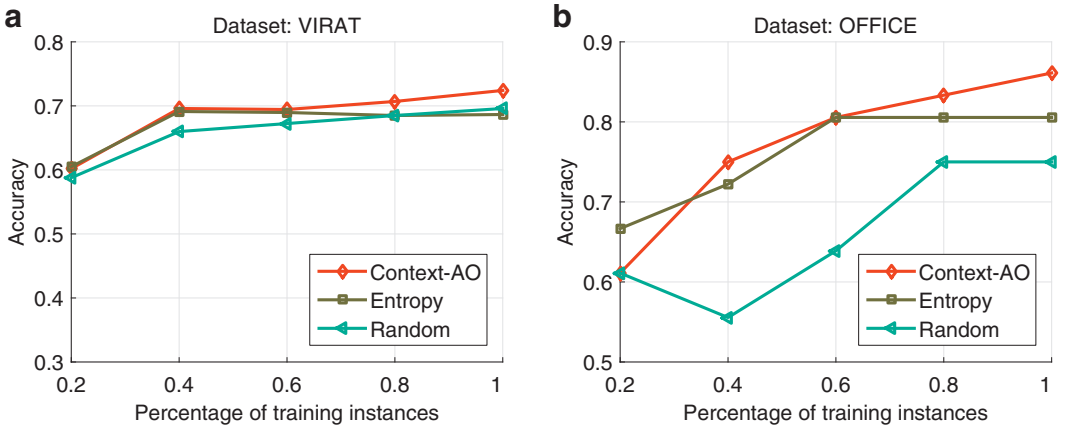

Fig. 9. Performance comparisons with other active learning techniques.

Table 2
Number of manually labeled instances used by different methods to obtain the results in Fig. 7.

Datasets	Total	Training	All-Manual	Strong+Weak	Strong	Weak
KTH	599	479	479	156 (32.6%)	249	96
UCF11	1523	1218	1218	597 (49%)	634	244
VIRAT	925	740	740	351 (47.5%)	385	148
UCLA-Office	136	109	109	48 (44.4%)	57	22

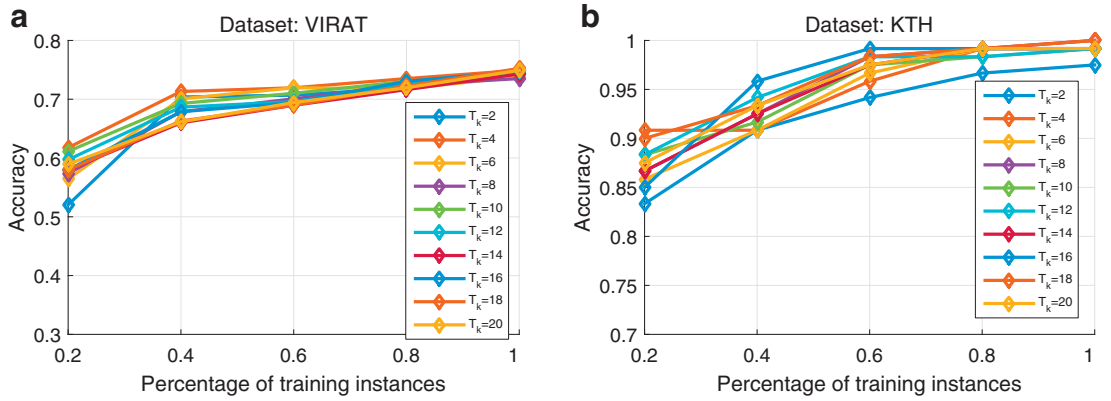

Fig. 10. Sensitivity of the parameter T_k. Please see the texts in Section 6.9 for more details.

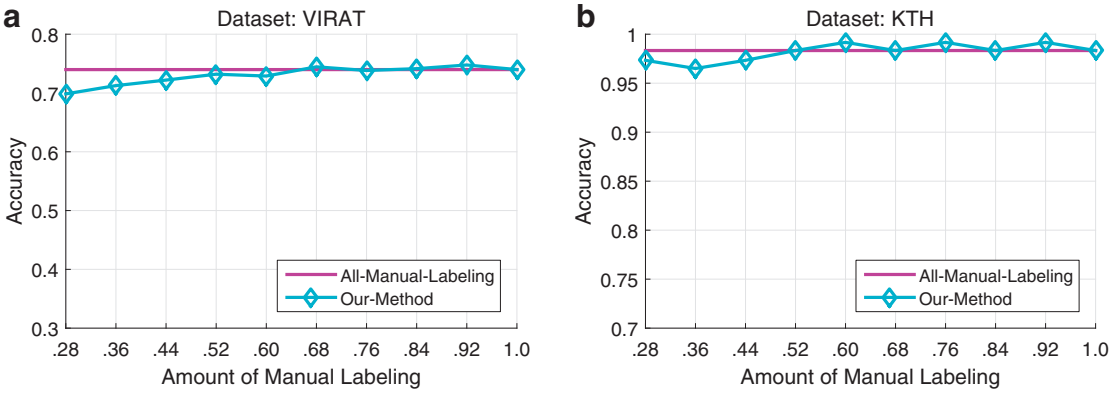

Fig. 11. Effect of the amount of manual labeling on the performance. Please see the texts in Section 6.9 for more details.

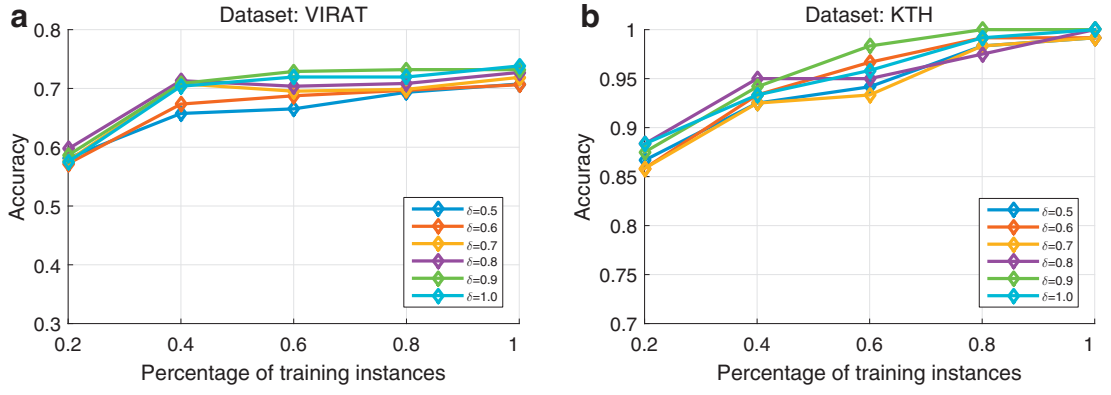

Fig. 12. Sensitivity analysis of the parameter δ. Please see the texts in Section 6.9 for more details.

7. Conclusion and future works

In this work, we proposed a framework for incremental activity modeling. Our framework took advantage of state-of-the-art machine learning tools and active learning to learn activity models incrementally over time with reduced amount of manually labeled data. We also exploit the contextual information and learn it incrementally so that it helps to recognize activities more efficiently over time. We performed rigorous experiments on four challenging datasets. Results show the robustness of our approach as accuracy asymptotically increases in all of the cases. One future direction of this work could be the use of interrelationships among the activities in a video sequence

during active learning, whereas this work only exploits the decision ambiguity of the classifier on an unknown instance independent of other instances to measure the informativeness. Some preliminary work in this direction is presented in [42].

References

[1] R. Poppe, A survey on vision-based human action recognition, Image Vis. Comput. 28 (6) (2010) 976–990.
[2] A. Oliva, A. Torralba, The role of context in object recognition, Trends Cogn. Sci. 11 (12) (2007) 520–527.
[3] B. Yao, L. Fei-Fei, Modeling mutual context of object and human pose in human-object interaction activities, in: Proceedings of the CVPR, 2010.
[4] Z. Wang, Q. Shi, C. Shen, Bilinear programming for human activity recognition with unknown mrf graphs, in: Proceedings of the CVPR, 2013.
[5] T. Lan, W. Yang, Y. Wang, G. Mori, Beyond actions: Discriminative models for contextual group activities, in: Proceedings of the NIPS, 2010.
[6] W. Choi, K. Shahid, S. Savarese, Learning context for collective activity recognition, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, 2011.
[7] N. Nayak, Y. Zhu, A. Roy-Chowdhury, Exploiting spatio-temporal scene structure for wide-area activity analysis in unconstrained environments, IEEE Trans. Inf. Forens. Sec. 8 (10) (2013).
[8] Y. Zhu, N.M. Nayak, A.K. Roy-Chowdhury, Context-aware modeling and recognition of activities in video, in: Proceedings of the CVPR, 2013.
[9] K. Reddy, J. Liu, M. Shah, Incremental action recognition using feature-tree, in: Proceedings of the ICCV, 2009.
[10] R. Minhas, A. Mohammed, Q. Wu, Incremental learning in human action recognition based on snippets, IEEETrans. Circ. Syst. Video Technol. 22 (11) (2012) 1529–1541.
[11] M. Hasan, A. Roy-Chowdhury, Incremental activity modeling and recognition in streaming videos, in: Proceedings of the CVPR, 2014.
[12] I. Laptev, On space-time interest points, Int. J. Comput. Vis. 64 (2-3) (2005) 107–123.
[13] M. Hasan, A. Roy-Chowdhury, Continuous learning of human activity models using deep nets, in: Proceedings of the ECCV, 2014.
[14] S. Sadanand, J. Corso, Action bank: a high-level representation of activity in video, in: Proceedings of the CVPR, 2012.
[15] R. Polikar, L. Upda, S. Upda, V. Honavar, Learn++: an incremental learning algorithm for supervised neural networks, IEEE TSMC Part:C 31 (4) (2001).
[16] H. He, S. Chen, K. Li, X. Xu, Incremental learning from stream data, IEEE TNN 22 (12) (2011) 1901–1914.
[17] W. Brendel, S. Todorovic, Learning spatiotemporal graphs of human activities, in: Proceedings of the ICCV, 2011.
[18] Z. Si, M. Pei, B. Yao, S.-C. Zhu, Unsupervised learning of event and-or grammar and semantics from video, in: Proceedings of the ICCV, 2011.
[19] A. Quattoni, S. Wang, L.-P. Morency, M. Collins, T. Darrell, Hidden-state conditional random fields, IEEE TPAMI (2007).
[20] B. Settles, Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan & Claypool Publishers 6 (1) (2012) 1–114.
[21] J.M. Buhmann, A. Vezhnevets, V. Ferrari, Active learning for semantic segmentation with expected change, in: Proceedings of the CVPR, 2012.
[22] A. Kapoor, K. Grauman, R. Urtasun, T. Darrell, Active learning with gaussian processes for object categorization, in: Proceedings of the ICCV, 2007.
[23] C. Loy, T. Xiang, S. Gong, Stream-based active unusual event detection, in: Proceedings of the ACCV, 2011.
[24] X. Liu, J. Zhang, Active learning for human action recognition with gaussian processes, in: Proceedings of the ICIP, 2011.
[25] R. Schapire, The strength of weak learning, Mach. Learn. 5 (2) (2005) 197–227.
[26] D. Ramanan, Learning to parse images of articulated objects, in: Proceedings of theNIPS, 2006.
[27] Y. Li, R. Nevatia, Key object driven multi-category object recognition, localization and tracking using spatio-temporal context., in: Proceedings of the ECCV, 2008.
[28] D.A. Ross, J. Lim, R.-S. Lin, M.-H. Yang, Incremental learning for robust visual tracking, Int. J. Comput. Vis. 77 (1-3) (2008) 125–141.
[29] Y. Hao, Y. Chen, J. Zakaria, B. Hu, T. Rakthanmanon, E. Keogh, Towards never-ending learning from time series streams, in: Proceedings of the SIGKDD, 2013.
[30] C. Schuldt, I. Laptev, B. Caputo, Recognizing human actions: a local svm approach, in: Proceedings of the ICPR, 2004.
[31] J. Liu, J. Luo, M. Shah, Recognizing realistic actions from videos “in the wild”, in: Proceedings of the CVPR, 2009.
[32] S. Oh, A. Hoogs, et. al., A large-scale benchmark dataset for event recognition in surveillance video, in: Proceedings of the CVPR, 2011.
[33] M. Pei, Y. Jia, S.-C. Zhu, Parsing video events with goal inference and intent prediction, in: Proceedings of the ICCV, 2011.
[34] Z. Zivkovic, Improved adaptive gaussian mixture model for background subtraction, in: Proceedings of the ICPR, 2004.
[35] P.F. Felzenszwalb, R.B. Girshic, D. McAllester, Discriminatively trained deformable part models, release 4,
[36] B. Song, T. Jeng, E. Staudt, A. Roy-Chowdury, A stochastic graph evolution framework for robust multi-target tracking, in: Proceedings of the ECCV, 2010.
[37] R. Chaudhry, A. Ravichandran, G. Hager, R. Vidal., Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions, in: Proceedings of the CVPR, 2009.
[38] M. Amer, S. Todorovic, Sum-product networks for modeling activities with stochastic structure, in: Proceedings of the CVPR, 2012.
[39] J. Liu, Y. Yang, M. Shah, Learning semantic visual vocabularies using diffusion distance, in: Proceedings of the CVPR, 2009.
[40] H. Wang, M.M. Ullah, A. Klaser, I. Laptev, C. Schmid, Evaluation of local spatio-temporal features for action recognition, in: Proceedings of the BMVC, 2009.
[41] Q. Le, W. Zou, S. Yeung, A. Ng, Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis, in: Proceedings of the CVPR, 2011.
[42] M. Hasan, A. Roy-Chowdhury, Context Aware Active Learning of Activity Recognition Models, in: ICCV, 2015.