Activity Auto-Completion: Predicting Human Activities from Partial Videos

Zhen Xu[1,2], Laiyun Qing[1,2], Jun Miao[2]
[1] Key Laboratory of Big Data Mining and Knowledge Management, Chinese Academy of Sciences, University of Chinese Academy of Sciences, Beijing, China
[2] Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing, China
xuzhen13@mails.ucas.ac.cn, lyqing@ucas.ac.cn, jmiao@ict.ac.cn

Abstract

In this paper, we propose an activity auto-completion (AAC) model for human activity prediction by formulating activity prediction as a query auto-completion (QAC) problem in information retrieval. First, we extract discriminative patches in frames of videos. A video is represented based on these patches and divided into a collection of segments, each of which is regarded as a character typed in the search box. Then a partially observed video is considered as an activity prefix, consisting of one or more characters. Finally, the missing observation of an activity is predicted as the activity candidates provided by the auto-completion model. The candidates are matched against the activity prefix on-the-fly and ranked by a learning-to-rank algorithm. We validate our method on UT-Interaction Set #1 and Set #2 [19]. The experimental results show that the proposed activity auto-completion model achieves promising performance.

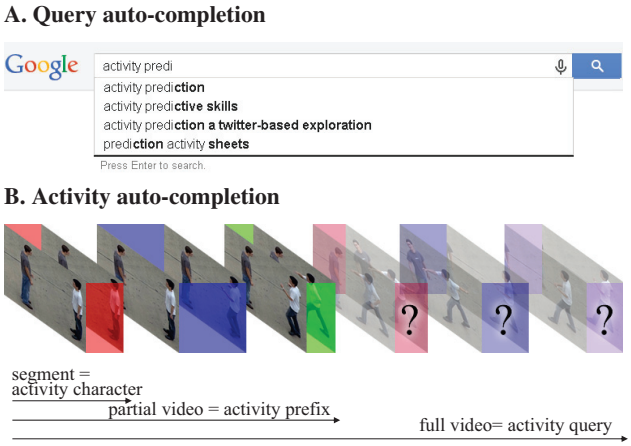


Figure 1. An analogy between activity prediction and query auto-completion. A) Query candidates for a query prefix: *activity predi*, according to google.com. B) We focus on activity prediction. Partially observed videos are considered as activity prefixes, consisting of one or more visual characters.

1. Introduction

The exponential growth of video devices has leaded to a tremendous demand for intelligent video analysis techniques. In the past decade, various methods have been proposed for recognizing after-the-fact activities [1]. Though great successes have been achieved, it's too luxury for an intelligent system to recognize human activities until the end of videos (*e.g.* video surveillance, health care, human-computer interaction). In recent years, predicting human activities from partially observed videos has been an active research topic [18, 8, 17, 13, 14].

Ryoo [18] defines human activity prediction as an inference of the ongoing activity given only partial observations (see Fig.1). Existing solutions based on multi-class sequential matching usually output a clear class label [18, 8]. However, the information contained in a partially observed video may be ambiguous. We claim that an intelligence system should not jump to a conclusion in the early stages of an activity. For instance, presented with first few frames of a surveillance video, we cannot judge whether it is shoplifting or just reaching for something. When more frames are observed, the judgement will be more confident. A good choice may be ranking all potential activities in real time.

Inspired by query auto-completion (QAC) [20], we propose a novel activity auto-completion (AAC) model for human activity prediction. A partially observed video and a full video, in our paper, are regarded as a *prefix* and a *query* respectively. Fig.1 illustrates the comparison between QAC and our proposed AAC. Just like QAC providing users with high quality candidates in modern search engines, our AAC recommends potential activities. We first explore mid-

1550-5499/15 $31.00 © 2015 IEEE
DOI 10.1109/ICCV.2015.365

IEEE computer society

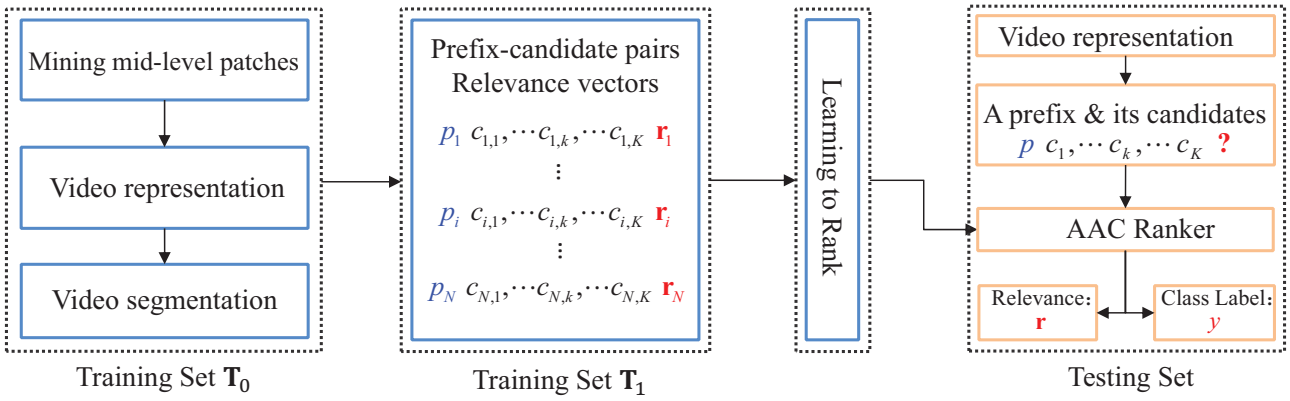


Figure 2. The framework of the proposed AAC model: First, mid-level discriminative patches are mined from training set $\mathbf{T}_0$, and each video is divided into some segments according to their patch-based representations. We then split videos in $\mathbf{T}_0$ into all activity prefixes and for each case obtain its activity candidates. Dataset $\mathbf{T}_1$ consists of such prefix-candidate pairs. Finally, we train our AAC ranker on the new dataset $\mathbf{T}_1$. In prediction phase, the ranking list of activity candidates is dynamically updated.

level patches [21] to represent videos by mining discriminative patches in each activity class. Then, we divide patch-represented videos into variable-length segments (characters). A full video (query) is decomposed into all activity prefixes that lead to it, and for each case other videos in training set are considered as its activity candidates. Activity candidates are ranked using learning-to-rank techniques.

An overview of our method is illustrated in Fig.2. Our method makes several contributions:

- We propose the idea of activity auto-completion for activity prediction.
- We show how to capture the essence of partial videos by mining discriminative patches.
- The proposed AAC model achieves impressive performances on UT-Interaction Set #1 and Set #2 [19].

The remainder of the paper is organized as follows. In Section 2, related work is covered. The details of the AAC model are introduced in Section 3. Experiments are discussed in Section 4. And we conclude in Section 5.

2. Related Work

Activity representation: Generally, features for representing videos include local features [15, 23] and global features [4, 3]. Global features are powerful since they encode the global templates of human activities. But they are more sensitive to viewpoints and deformations. On the other hand, local features [15, 23] encode only local information surrounding the interesting points, which sometimes are not discriminative enough. Recently, some works focus on extracting mid-level features [5, 21, 10, 24, 11, 2]. The work of poselets [5] learns both appearance and configuration space of human body parts, which requires non-trivial amounts of hand-labeled training data (*e.g.* key points). Spatio-temporal patches [24, 11] have achieved exciting performance in human action recognition, but they may not be suitable in activity prediction task with only partial observations available. Researches on mid-level patches [21, 10, 2] mine representative patches from a large number of randomly sampled patches. In our paper, we apply the method described in [21] to discover mid-level patches in each activity class.

Activity prediction: Ryoo [18] designs an integral bag-of-words (IBoW) and a dynamic bag-of-words (DBoW) to represent human activities. The activity model of each progress level is obtained by averaging features of a particular progress level in the same class, which may suffer from outliers. Cao *et al*. [8] apply sparse coding (SC) to derive activity bases at different observation ratios, then partial videos in testing are reconstructed by theses bases. Moreover, they extend SC to handle intra-class activity variations by including bases from different time periods. Raptis and Sigal [17] develop a model to recognize activity from streaming video based on poselets [5]. Their model localizes activity keyframes temporally and spatially in both partial and full videos. Kong *et al*. [13] propose a multiple temporal scale support vector machine (MTSSVM) to handle action prediction problem and their model takes advantage of a prior that the more frames are observed, the more information will be obtained. A temporal action evolution constraint and a label consistency constraint are considered in their structured SVM model. Lan *et al*. [14] introduce a novel three-layer hierarchical movemes representation for predicting actions from short video clips or still images. Each layer captures human actions at different semantic and temporal granularity. However, their model is builded on an assumption that human bounding boxes and human motion

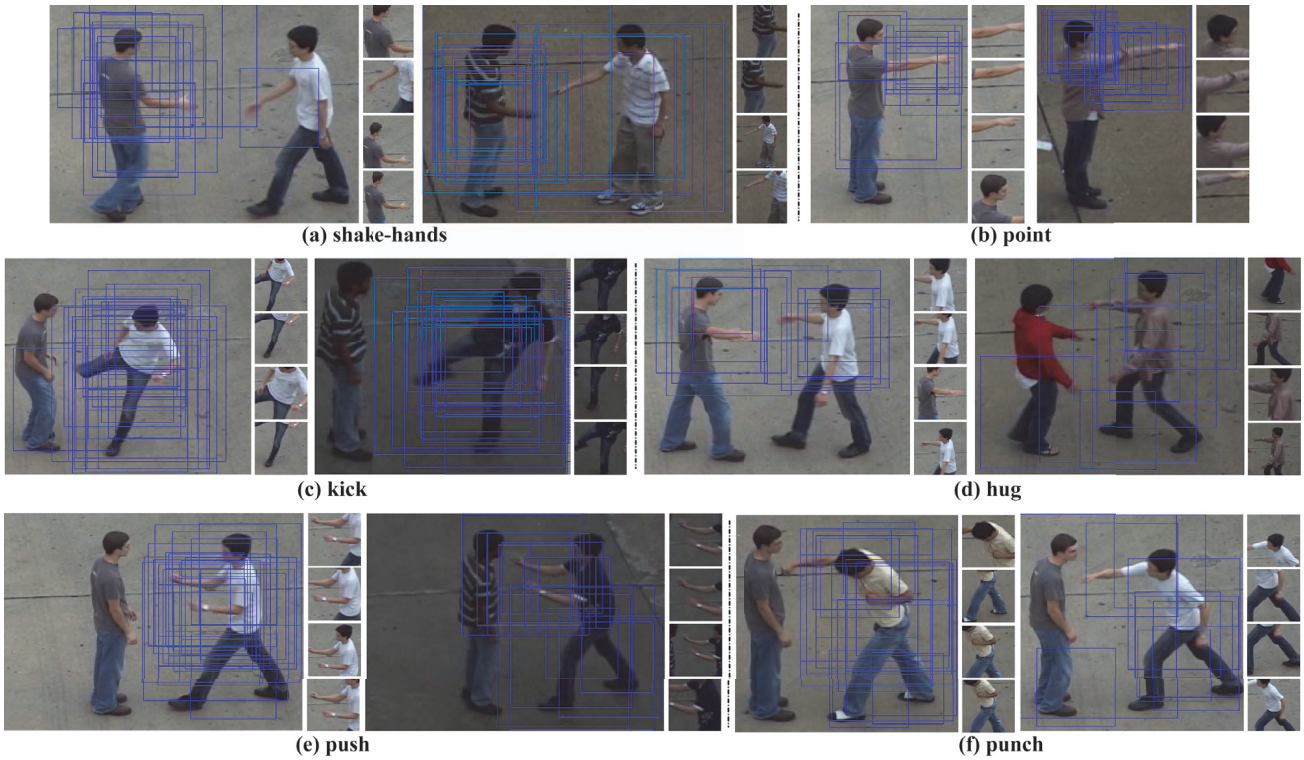


Figure 3. Examples of discriminative patches discovered by refined SVM detectors. Patches in a frame with top four detection scores are displayed individually.

tracks are available during training. In addition, Li and Fu [16] focus on solving long-duration complex activity prediction problem through sequential pattern mining, which is different from our goal.

In this paper, we propose a novel activity auto-completion (AAC) model for human activity prediction, avoiding jumping to a conclusion in the early stages of an activity.

3. Our Approach

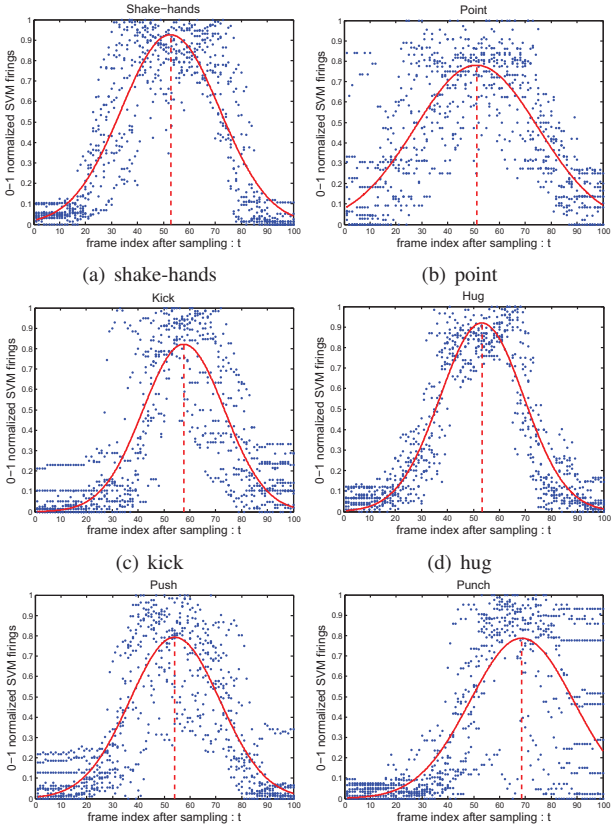
Given a fully observed activity video $x[1:T]$ of length T, a partial observation of it is $x[1:t]$, where $t = 1, \ldots, T$. Note, the lengths T of different videos may vary. We adapt query auto-completion [20] here to recognize activity class label y from the partial video $x[1:t]$. In the paper, a partial video and a full video are considered as a *prefix* and a *query*, respectively.

We first introduce how to represent videos by mining discriminative mid-level patches in Section 3.1. Video segmentation based on clustering are discussed in Section 3.2. Then, the details of the proposed AAC model are given in Section 3.3, 3.4 and 3.5.

3.1. Activity Representation

We build a compact video representation based on discriminative patches [21]. These patches have two characteristics: 1) **purity**: they should fire frequently enough in one activity class; 2) **discriminativeness**: they should occur rarely in other activity classes. Note, here "rarely" doesn't mean "never". We adopt a *one-against-rest* strategy to discover discriminative patches for each activity class. For example, there are M activity classes in training set. We select all video frames from the m-th type of activity as positive samples, and frames from the remaining $M-1$ activity classes as negative samples, where $m = 1, \ldots, M$. Following the procedure described in [21], we discover the discriminative patches for this type of activity class by alternating the steps of detecting similar patches and training new support vector machine (SVM) detectors. Patches with similar feature descriptors are equipped with a same linear SVM detector.

Let $\mathbf{S}_m = \{s_j^m\}_{j=1}^{n_m}$ be the SVM detectors mined for m-th activity class and n_m is the number of SVM detectors mined in m-th activity class. However, the numbers n_m are unbalanced. For example, in our experiments on UT-Interaction Set #1 [19], there are over 300 detectors mined in "push" class and only about 150 detectors discovered in



(a) shake-hands (b) point (c) kick (d) hug (e) push (f) punch

Figure 4. Patch distributions of six activities on UT-Interaction Set #1. The lengths of videos are regularized to 100. The numbers of patches in a video are frame-by-frame counted and 0-1 normalized. Data points from all videos belonging to a same activity class are plotted in a single figure and fitted with a Gaussian function.

"punch" class. Several factors may contribute to this unbalance. First, activities from different classes have their own characteristics. Secondly, patches mined in a class don't necessarily satisfy the aforementioned two characteristics, for some SVM detectors only focus on the backgrounds.

Refining Detectors: To maintain the balance of detectors, we refine J representative SVM detectors from $\mathbf{S}_m$ for each class based on patch temporal distribution. Given V_m training videos belonging to the m-th activity class, we run detectors in $\mathbf{S}_m$ on these training videos at multiple resolutions (4) in a sliding window fashion. If the output score of a SVM detector is greater than a threshold γ (-1), a patch is discovered. We count the number of discovered patches in each video frame. The frame-by-frame counts in a video are normalized to the range of $[0, 1]$. We also regularize the lengths of videos to a canonical length (100) by upsampling or downsampling. Note, sampling is not applied in other sections of this paper. Fig.4 gives some examples on the numbers of patches on UT-Interaction Set #1 [19]. Data points are fitted with Gaussian functions. Clearly, there

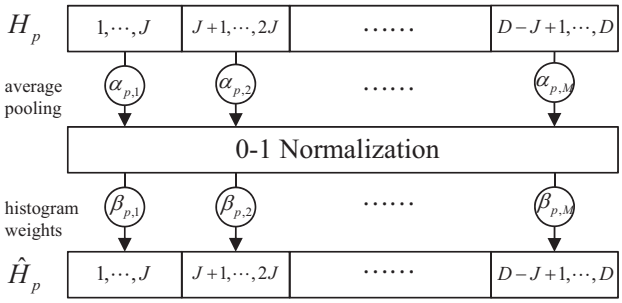


Figure 5. The weighting procedure of the patch-based histogram

are more patches around the center of each Gaussian distribution. This suggests that there are little discriminative information in the early stages of an activity.

To select really representative patch detectors for m-th activity class, we sum up the SVM detection scores around the center of each Gaussian distribution as

$$\Theta(s_j^m) = \frac{1}{V_m} \sum_{v=1}^{V_m} \sum_{l=t_1}^{t_2} \varphi_l^v(s_j^m), \tag{1}$$

where $\varphi_l^v(s_j^m)$ is the maximum detection score of detector s_j^m in the l-th frame of video v, t_1 is the 25-th frame on the left of the center and t_2 is the 25-th frame on the right of the center.

All detectors in $\mathbf{S}_m$ are ranked according to their cumulations $\Theta(\cdot)$. We select the top-ranked J (50) SVM detectors as representatives for this activity class. Fig.3 shows some examples of the detected patches. It can be seen from Fig.3 that patches for every activity are distinct and representative.

Patch-based Histogram: Let $\mathbf{S}'_m = \{s_j^m\}_{j=1}^J$ denote the survived SVM detectors. Following the classical bag-of-words (BoW) scheme, we consider all detectors together as the codebook of the BoW. The size of codebook is $D = M \times J$. Given a prefix $p = x[1:t]$, let H_p denote the patch-based histogram of p, so that $H_p(d)$ is the average number of firings in this prefix with respect to a detector,

$$H_p(d) = \frac{1}{t} \sum_{l=1}^{t} \mathbb{1}\big(\varphi_l^p(s_j^m) > \gamma\big), \tag{2}$$

where $d = (m-1) \times J + j$ is the bin index of H_p, and $\mathbb{1}$ is indicator function.

Weighted Histogram: Each bin in histogram H_p corresponds to a SVM detector from a specific activity class. Ideally, class-specific detectors only fire in their own class, but they maybe ambiguous and activate aimlessly. We compute a weighted version $\hat{H}_p$, aiming to give more weights to detectors from the dominant activity class. In other words,

if an activity class holds the largest number of discriminative patches in this prefix p, it is dominant and we boost the weights of the corresponding detectors. As shown in Fig.5, $\hat{H}_p$ is obtained in three steps:

1. Average pooling computes the average number $\alpha_{p,m}$ of SVM detections per activity class.
2. 0-1 normalization is used to obtain a weight value $\beta_{p,m}$ between 0 and 1.
3. SVM detectors from the m-th activity class share a same weight: $\hat{H}_p(d) = \beta_{p,m} H_p(d)$.

3.2. Video Segmentation

Many works [27, 6] have shown evidence that human motion is locally linear (see Fig.1), so rather than setting the video granularity to frames, we divide a full video $x[1:T]$ into a collection of segments, which we call *activity characters*. This operation also reduces the number of prefixes.

Given a video with T frames $p = x[1:T]$, each frame $f = x[t:t]$ in p is firstly represented by patch-based histogram H_f. Then affinity propagation (AP) [7] is used to assign each frame f to a cluster with histogram intersection similarity. And we get a cluster label vector $\mathbf{g}$ of length T. However, the initial AP assignment of cluster labels is rough and labels are not consistent along the timeline, so we further apply a simple voting scheme to reassign labels. This relabeling procedure is like median filter, but each entry is replaced with the dominant cluster label of neighboring entries in a one-dimensional window of size W (9). This procedure is repeated until the labels of all frames don't change. After relabeling, adjacent frames with same cluster label in vector $\mathbf{g}$ are considered as an activity character in this video (see Fig.1). The algorithm for video temporal segmentation is summarized in Algorithm 1.

Algorithm 1 Video Temporal Segmentation

Input:
A fully observed video: $p = x[1:T]$;
Feature vector of each frame f: H_f;
Window size of voting: W.
Output:
A cluster label vector of length T: $\mathbf{g}$
Initialize $\mathbf{g}$ by APCLUSTER(p, H_f)
repeat
  $\mathbf{g} \leftarrow$ VOTING$(\mathbf{g}, W)$
until $\mathbf{g}$ doesn't change
return $\mathbf{g}$

3.3. Prefix-Candidate Pairs

In our work, we formulate activity prediction problem as an auto-completion problem rather than traditional multi-class matching problem. Query auto-completion (QAC) is one of the requisite features in modern search engines, such as Google, Bing, *et al*. Its goal is to predict users intent and suggest possible other queries matching the first few words typed. Usually, learning to auto-completion [20] is a kind of supervised learning, and the learning is a two-step process. Step one is to obtain and filter query suggestions based on the matching against the prefix on-the-fly. Step two is to learn to rank the filtered candidates based on the likelihood of those suggestions being the correct one.

Following these steps in QAC, we create a new training set $\mathbf{T}_1$ for activity auto-completion task from the original training set $\mathbf{T}_0$, which consists of [1]:

- prefixes $\{p_i\}_{i=1}^{N}$, where $p_i = x[1:t]$.
- candidates $\{c_{i,k}\}_{i=1,k=1}^{N,K}$, where $c_{i,k} = x[1:T]$.
- relevance vectors $\{\mathbf{r}_i\}_{i=1}^{N}$,where $\mathbf{r}_i \in \mathbb{R}^K$.

where N is number of all activity prefixes, K is the number of activity candidates associated with a prefix.

First we decompose each full video (query) in original training set $\mathbf{T}_0$ into all prefixes that lead to it. Each prefix is composed of one or more activity characters. Secondly, for each prefix we obtain its activity candidates by using prefix matching. Typical exact prefix matching techniques in information retrieval are on the basis of special data structures, such as hash tables, prefix-trees (tries), *etc*. These data structures are efficient in candidates lookup and filtering. However, in our prediction task, activity characters are not exactly quantified, therefore we resort to a fuzzy prefix matching technique to select and filter activity candidates from $\mathbf{T}_0$. Our fuzzy matching method calculates the similarities between a prefix and its candidates based on histogram intersection similarity, and ranks the candidates in descending order. Then we restrict candidates of an activity prefix to top K videos. Thirdly, there are only two relevance levels in our activity auto-completion problem. For each prefix-candidate pair, we assign a positive label to the candidate if the prefix and candidate have a common class label in $\mathbf{T}_0$, otherwise a zero label.

3.4. Learning to Rank

Once the new training set $\mathbf{T}_1$ is created, many existing learning-to-rank algorithms can fit into our activity auto-completion framework, such as Lambda-MART [25], RankSVM [12], *etc*. In our work, we choose RankSVM as our learning model. Particularly, only one ranking model is created for all activity classes.

Joachims [12] has showed that the final goal of learning-to-rank is to learn a linear ranking function (3), such that

[1] To avoid ambiguity, we use $\mathbf{T}_0$ and $\mathbf{T}_1$ to denote the training set for mining discriminative patches and the training set for learning the auto-completion ranker, respectively.

the maximum number of the inequalities (4) between relevant prefix-candidate pairs $(p_i, c^+_{i,k})$ and irrelevant pairs $(p_i, c^-_{i,k'})$ are satisfied. Here $c^+_{i,k}$ and $c^-_{i,k'}$ represent relevant and irrelevant candidates of a prefix p_i respectively, and their numbers are K^+_i and K^-_i ($K^+_i + K^-_i = K$).

$$\mathcal{L}\big(\mathbf{w}, (p_i, c_{i,k})\big) = \mathbf{w}^{\mathrm{T}}\Phi(p_i, c_{i,k}) \tag{3}$$

$$\mathbf{w}^{\mathrm{T}}\Phi(p_i, c^+_{i,k}) > \mathbf{w}^{\mathrm{T}}\Phi(p_i, c^-_{i,k'}) \tag{4}$$

where $i = 1, \ldots, N$, $k = 1, \ldots, K^+_i$, $k' = 1, \ldots, K^-_i$ and $\Phi(p_i, c_{i,k})$ is a joint feature mapping that reflects the compatibility between an activity prefix and its activity candidates. Details of the joint feature mapping will be discussed in Section 3.5. We solve the ranking function with the publicly available PRSVM software [9] which integrates primal Newton method to speed up RankSVM training and shows excellent performance.

Prediction Phase: When ranking function (3) is learned, our AAC model can automatically complete any new activity prefix (see the rightmost flow chart in Fig.2): 1) Given a new prefix, we obtain its activity candidates from training set $\mathbf{T}_0$. 2) We calculate the joint feature mapping of the prefix-candidate pairs. 3) The pairs are feeded into the learned AAC ranker, and we will get a ranking of these activity candidates according to their relevance outputs. 4) Further, we can consider the first-ranked candidate's class label as the final predicted label.

3.5. Joint Feature Mapping

To operationalize our AAC ranker, we need to design a suitable joint feature mapping $\Phi(p_i, c_{i,k})$ which can depict the common ground between an activity prefix p_i and its candidates $c_{i,k}$. In the paper, a prefix and its candidates are represented by patch-based histogram H_{p_i} and $H_{c_{i,k}}$. The joint feature mappings are then designed as follow.

1) **histogram_intersection_similarity**:

$$\Phi_1(p_i, c_{i,k}) = \frac{\sum_{d=1}^{D} \min\big(H_{p_i}(d), H_{c_{i,k}}(d)\big)}{\sum_{d=1}^{D} H_{p_i}(d)} \tag{5}$$

Φ_1 calculates the similarity between histograms $H(p_i)$ and $H(c_{i,k})$ and reflects the overall quality of matching between a prefix and its candidates. (dimension is 1)

2) **histogram_intersection**:

$$\Phi_2(p_i, c_{i,k})(d) = \min\big(H_{p_i}(d), H_{c_{i,k}}(d)\big), \tag{6}$$

where $d = 1, \ldots, D$. Φ_2 is calculated by just intersecting two histograms. Each element of it reflects how many common discriminative patches are discovered in a prefix and its candidates by a same SVM detector. (dimension is D)

In our final model, we concatenate Φ_1 and Φ_2 together, so the dimension of the joint feature mapping $\Phi(p_i, c_{i,k})$ is $D + 1$.

Figure 6. Example snapshots of six different activities (shake-hands, point, kick, hug, push and punch) from UT Set #1 (top) and UT Set #2 (bottom). Set #2 is more complicated than UT Set #1, since there are more tree moves, camera jitters, *etc*.

4. Experiments

We test the activity auto-completion (AAC) model on UT-Interaction Set #1 and UT-Interaction Set #2 [19]. These two datasets are created for high-level activity analysis, and both of them consist of six different types of human-human interaction activities: shake-hands, hug, kick, point, punch and push, with 10 videos per activity class. Fig.6 shows example snapshots of six different activities from two datasets. Backgrounds in Set #2 are more complex than backgrounds in Set #1 (*e.g*. tree moves, camera jitters).

4.1. Details

Following the experiment settings in [18], 10-fold leave-one-sequence-out cross validation setting is used to measure model performances on both Set #1 and Set #2. For each round, six segmented videos from a same unsegmented long video are used for testing, and other 54 videos are used for training. Since there are only 60 videos on UT-Interaction Set #1 or Set #2, so in experiments, we don't filter candidates. That is to say, in training, the parameter K for fuzzy matching is set to 53, because one out of 54 videos is used to generate activity prefixes. In testing, the parameter K is set to the number of all training videos ($K = 54$). The trade-off parameter C of PRSVM, in all experiments, is set to 100.

4.2. Results

Our ACC model is compared with DBoW and IBoW in [18], SC and MSSC in [8], MTSSVM in [13], and some other baseline methods. To compare with them, we uniformly divide a full testing video into 10 segments and test our model at 10 different observation ratios from 0.1 to 1. Note, all the experimental results are achieved by taking the first-ranked activity candidate's class label as the final predicted label y.

UT-Interaction Set #1: Fig.7(a) illustrates the performances on the UT Set #1. The horizontal axis of the figure corresponds to the observed ratio, while the vertical axis represents the average prediction accuracy. The dimension of histograms ranges from 1416 to 1644 (10-fold) without

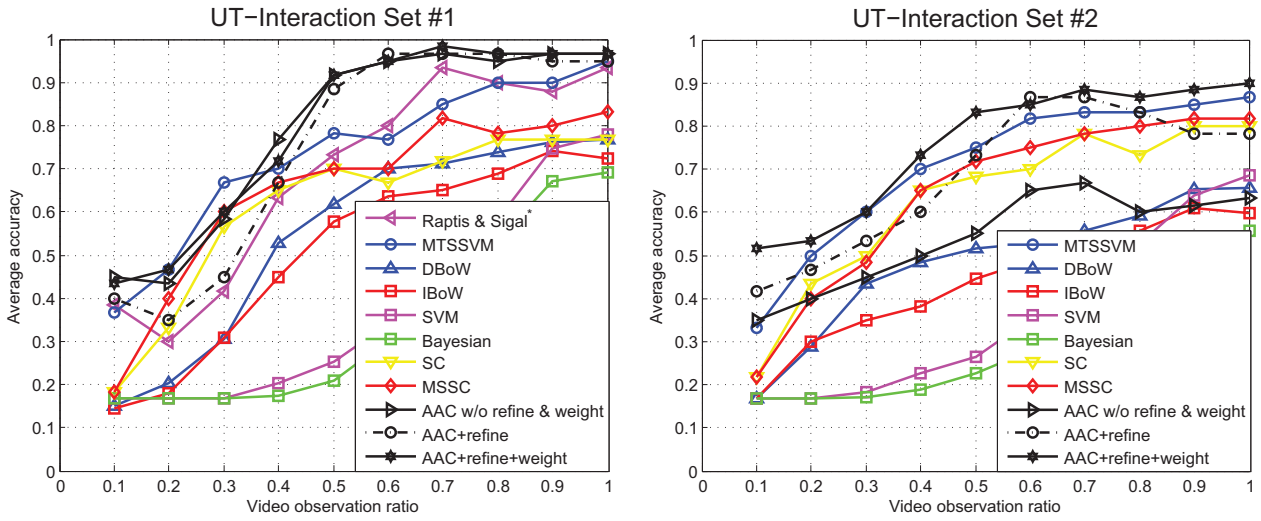
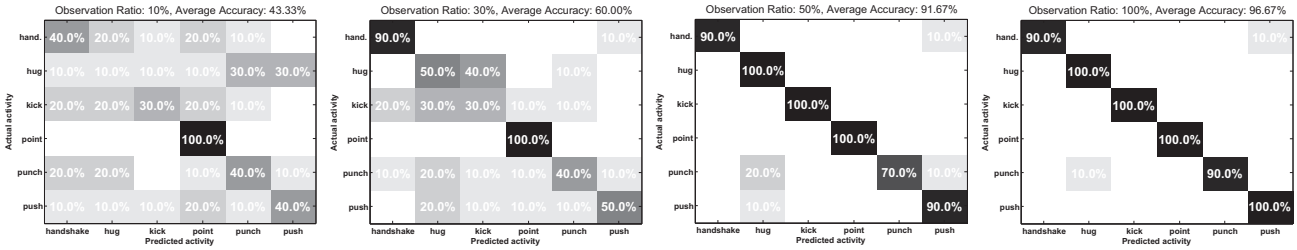


Figure 7. Prediction results on Set #1 (left) and Set #2 (right). ∗Raptis and Sigal [17] only report results on Set #1.



(a) Observation ratio: 10% (b) Observation ratio: 30% (c) Observation ratio: 50% (d) Observation ratio: 100%

Figure 8. Confusion matrices for UT-Interaction Set #1 at 0.1, 0.3, 0.5 and 1.0 observation ratios.

refining. After refining, the dimension is reduced to 300 at cost of some accuracy. When we combine refining with reweighting strategy, weighted histogram gives us a significant boost in prediction performance. Remarkably, compared to the existing activity prediction methods, our proposed AAC model with weighted histogram exhibits great improvement. When half video is observed, our AAC model with weighted histogram performs superior to all previous methods and achieve 91.67% accuracy. At 0.6 observation ratio, average prediction accuracy of 95.00% is achieved, which is as good as the current best result [13] with full observation.

Table 1 compares the results of our method with other leading approaches on Set #1 in a more direct way. With half observation, the accuracy obtained by our weighted model is 13% higher than MTSSVM [13] and is 8% higher than Lan *et al*. [14] which is the current state-of-art at 0.5 observation ratio. With full observation, our weighted model achieves state-of-the-art accuracy of 96.67%.

The confusion matrices obtained by weighted model at different observation ratios on UT Set #1 are illustrated in Fig.8. It is obvious that "point" activities are easy to be recognized at any observation ratio. Whereas "punch" activities are more difficult to be identified in early stages (e.g. observation ratio is less than 0.3). Such results are consistent with the cues in Fig.4. The numbers of detected patch-

Table 1. Activity prediction performances on UT Set #1

Methods	Accuracy (half video)	Accuracy (full video)
AAC+refine+weight	**91.67%**	**96.67%**
AAC+refine	**88.33%**	**95.00%**
MTSSVM [13]	78.33%	**95.00%**
Lan *et al*. [14]	83.1%	88.4%
DBoW [18]	70.0%	85.0%
IBoW [18]	65.0%	81.7%
SVM [18]	25.3%	78.0%
Bayesian [18]	20.9%	69.2%
SC [8]	70.0%	76.67%
MSSC [8]	70.0%	83.33%
Raptis and Sigal [17]	73.3%	93.3%
Zhang *et al*. [26]	-	95%
Vahdat *et al*. [22]	-	93%

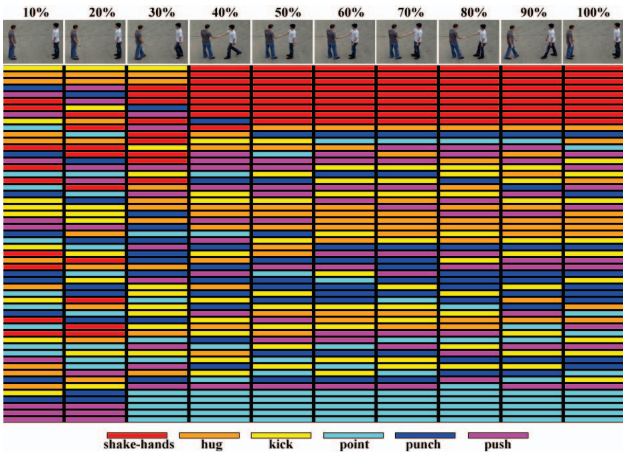


Figure 9. We visualize the rankings of a "shake-hands" activity at different observation ratios. In each column, the top row shows the current activity prefix. The middle rows show 54 activity candidates of this prefix. The last row shows that activity candidates from different activity classes are represented by distinct colors.

es of "point" class (see Fig.4(b)) distribute more uniformly along time axis, which means that some discriminative patches are discovered at the early stages, leading to good recognition performance. On the other hand, The numbers of detected patches of "punch" (see Fig.4(f)) surge at later stages (*e.g.* observation ratio is 0.7), which indicate that few discriminative patches can be founded at early stages, resulting in poor identification.

UT-Interaction Set #2: The experimental results on Set #2 are shown in Fig.7(b). Without refining, the dimension of histograms ranges from 1193 to 1262 (10-fold). After refining, the dimension is reduced to 300. With half observation, our weighted model achieves 83.33% accuracy, and our unweighted model achieves 73.33% accuracy. With full observation, the results are 90.00% and 78.33% respectively. In general, when compared with previous methods, the weighted AAC model consistently outperforms them.

Table 2. Mean Reciprocal Rank on UT Set #1

Obser. ratio	0.1	0.2	0.3	0.4	0.5
MRR	0.5542	0.5629	0.6434	0.7638	0.9285
Obser. ratio	0.6	0.7	0.8	0.9	1
MRR	0.9616	0.9867	0.9742	0.9718	0.9713

Table 3. Mean Reciprocal Rank on UT Set #2

Obser. ratio	0.1	0.2	0.3	0.4	0.5
MRR	0.5992	0.6166	0.6773	0.7861	0.8704
Obser. ratio	0.6	0.7	0.8	0.9	1
MRR	0.8937	0.9183	0.9191	0.9179	0.9226

Ranking Quality: Since we formulate activity prediction as a retrieval problem, we also use a retrieval metric to measure the performance of our model. Examples of the ranked candidates with "shake-hands" prefixes are depicted in Fig.9. In the cases of only 10% observations, the "shake-hands" prefix is predicted mistakenly as "kick" by the top one candidate, following by "hug", the top two and the top three candidates. With more observations are available, the true activities "shake-hands" pop up. That is to say, the ranking performance becomes better.

Mean reciprocal rank (MRR) is adopted here to measure ranking performance. MRR at a given observation ratio is calculated as the average of the reciprocal rank (RR):

$$\mathrm{MRR} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{\mathrm{rank}_i}, \tag{7}$$

where n is the number of the total prefixes at a given observation ratio. rank_i is the rank of the first correct activity candidate of a prefix.

Table 2 presents the results of our weighted model in terms of MRR on Set #1. When at least half frames are observed, MRR value is 0.9285 and gradually close to 1. Similar results of Set #2 are presented in Table 3. These quantitative results conclusively demonstrate that our AAC model can provide valuable activity recommendations.

5. Conclusions

We present a novel activity auto-completion (AAC) model for activity prediction in this paper. We explore discriminative patches for video representation and construct prefix-candidate pairs for auto-completion based on such representation. Then the missing observation of an activity is automatically completed by the auto-completion model. Encouraging experimental results have been obtained on the UT-Interaction dataset. We would like to do further researches about the temporal evolution of mid-level features, expecting better performance in the early stages of ongoing activities.

6. Acknowledgments

This research is partially sponsored by Natural Science Foundation of China (Nos. 61272320, 61175115 and 61472387), Beijing Natural Science Foundation (No. 4152005), the President Project of University of Chinese Academy of Sciences (No. Y35101CY00), and the Open Project of Key Laboratory of Big Data Mining and Knowledge Management, Chinese Academy of Sciences.

References

[1] J. K. Aggarwal and M. S. Ryoo. Human activity analysis: A review. *ACM Computing Surveys (CSUR)*, 2011.

[2] B. Antic and B. Ommer. Learning latent constituents for recognition of group activities in video. In *ECCV*. 2014.

[3] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. In *ICCV*, 2005.

[4] A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *TPAMI*, 2001.

[5] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *ICCV*, 2009.

[6] R. Bowden. Learning statistical models of human motion. In *IEEE Workshop on Human Modeling, Analysis and Synthesis, CVPR*, 2000.

[7] J. F. Brendan and D. Delbert. Clustering by passing messages between data points. *Science*, 2007.

[8] Y. Cao, D. Barrett, A. Barbuand, S. Narayanaswamy, H. Yu, A. Michaux, Y. Lin, S. Dickinson, J. M. Siskind, and S. Wang. Recognize human activities from partially observed videos. In *CVPR*, 2013.

[9] O. Chapelle and S. S. Keerthi. Efficient algorithms for ranking with svms. *Information Retrieval*, 2010.

[10] A. Eigenstetter, M. Takami, and B. Ommer. Randomized max-margin compositions for visual recognition. In *CVPR*, 2014.

[11] A. Jain, A. Gupta, M. Rodriguez, and L. S. Davis. Representing videos using mid-level discriminative patches. In *CVPR*, 2013.

[12] T. Joachims. Optimizing search engines using clickthrough data. In *ACM SIGKDD*, 2002.

[13] Y. Kong, D. Kit, and Y. Fu. A discriminative model with multiple temporal scales for action prediction. In *ECCV*. 2014.

[14] T. Lan, T. C. Chen, and S. Savarese. A hierarchical representation for future action prediction. In *ECCV*. 2014.

[15] I. Laptev and T. Lindeberg. Space-time interest points. In *ICCV*, 2003.

[16] K. Li and Y. Fu. Prediction of human activity by discovering temporal sequence patterns. *TPAMI*, 2014.

[17] M. Raptis and L. Sigal. Poselet key-framing: A model for human activity recognition. In *CVPR*, 2013.

[18] M. S. Ryoo. Human activity prediction: Early recognition of ongoing activities from streaming videos. In *ICCV*, 2011.

[19] M. S. Ryoo and J. k. Aggarwal. Ut-interaction dataset, icpr contest on semantic description of human activities (sdha). In *ICPR Workshops*, 2010.

[20] M. Shokouhi. Learning to personalize query autocompletion. In *ACM SIGIR*, 2013.

[21] S. Singh, A. Gupta, and A. A. Efros. Unsupervised discovery of mid-level discriminative patches. In *ECCV*. 2012.

[22] A. Vahdat, B. Gao, M. Ranjbar, and G. Mori. A discriminative key pose sequence model for recognizing human interactions. In *ICCV Workshops*, 2011.

[23] H. Wang and C. Schmid. Action recognition with improved trajectories. In *ICCV*, 2013.

[24] L. Wang, Y. Qiao, and X. Tang. Motionlets: Mid-level 3d parts for human motion recognition. In *CVPR*, 2013.

[25] Q. Wu, C. J. C. Burges, K. M. Svore, and J. Gao. Adapting boosting for information retrieval measures. *Information Retrieval*, 2010.

[26] Y. Zhang, X. Liu, M. C. Chang, W. Ge, and T. Chen. Spatio-temporal phrases for activity recognition. In *ECCV*. 2012.

[27] F. Zhou, F. De la Torre, and J. K. Hodgins. Hierarchical aligned cluster analysis for temporal clustering of human motion. *TPAMI*, 2013.