

Action Detection in Crowd

Parthipan Siva
psiva@dcs.qmul.ac.uk
Tao Xiang
txiang@dcs.qmul.ac.uk

School of EECS,
Queen Mary University of London,
London E1 4NS, UK

Abstract

For automated surveillance, it is useful to detect specific actions performed by people in busy natural environments. This differs from and thus more challenging than the intensively studied action recognition problem in that for action detection in crowd an action of interest is often overwhelmed by large number of background activities of other objects in the scene. Motivated by the success of sliding-window based 2D object detection approaches, in this paper, we propose to tackle the problem by learning a discriminative classifier from annotated 3D action cuboids to capture the intra-class variation, and sliding 3D search windows for detection. The key innovation of our method is a novel greedy k nearest neighbour algorithm for automated annotation of positive training data, by which an action detector can be learned with only a single training sequence being annotated thus greatly alleviating the tedious and unreliable 3D manual annotation. Extensive experiments on real-world action detection datasets demonstrate that our detector trained with minimal annotation can achieve comparable results to that learned with full annotation, and outperforms existing methods.

1 Introduction

The use of digital video has been increasing rapidly in recent years. As the amount of video content in use increases, methods for searching and monitoring video feeds become very important. The ability to automatically detect actions performed by people in busy natural environments is of particular interest for surveillance applications where cameras monitor large areas of interest looking for specific actions like fighting or falling (see Fig. 1(b)).

Action detection in crowd is closely related to the action recognition problem which has been studied intensively recently [1, 2, 3]. However, there are key differences between the two problems which make action detection a much harder problem. Specifically an action recognition method assumes that each video clip has been pre-segmented to contain a single action with little or no background activities (see Fig. 1(a)). For real-world applications such as detecting fighting or falling in crowd (Fig. 1(b)) we will not have segmented clips nor can we assume the action occurs with little or no background activity. In order to recognize actions that are dwarfed by other background activities (Fig. 1(b-d)), we need to spatially and temporally localize as well as recognize the action. The existing action recognition methods are thus unsuitable for solving our problem. There have been a few attempts in the past three years on action detection [4, 5, 6, 7]. Nevertheless most of them are based on matching templates constructed from a single sample thus unable to cope with the large

intra-class variations caused by actions being executed by different people at different speed and captured at different view angle. Action detection remains a largely unsolved problem.

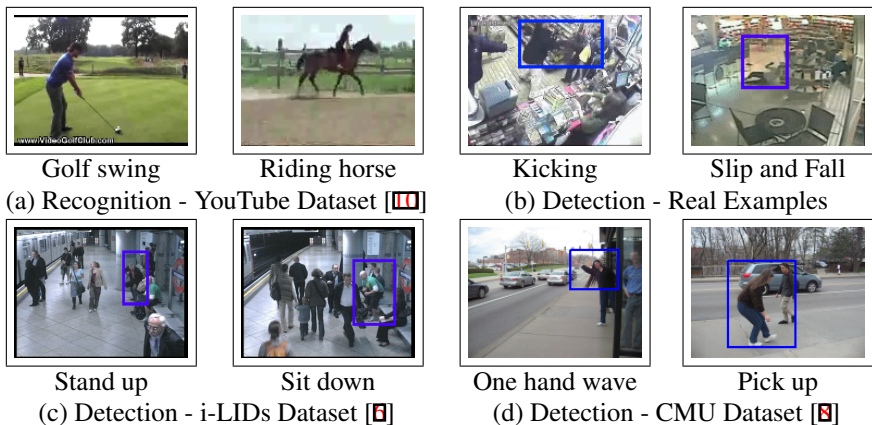


Figure 1: Action recognition vs. detection. (a) Recognition - each video only contain a single action. (b-d) Detection - action is amid other background activities.

We propose to tackle the problem by learning a discriminative classifier based on Support Vector Machine (SVM) from annotated 3D action cuboids to capture the intra-class variation, and sliding 3D search windows for detection. Our method is motivated by the success of sliding-window based 2D object detection approaches [5] that localize and recognize objects against background clutter and other objects in a static image. Similar to learning a 2D object detector, to learn a discriminative action detector, a large number of 3D action cuboids have to be annotated, which is even more tedious and unreliable compared to 2D object annotation. To overcome this problem, we propose a novel greedy k nearest neighbour algorithm for automated annotation of positive training data, by which an action detector can be learned with only a single manually annotated training sequence, thus greatly reducing the amount of manual annotation required. Our method can be seen as a solution to the multi-instance learning (MIL) problem [6], for weakly supervised learning. However, compared to conventional MIL methods it is much better at coping with large amounts of background activities that are visually similar to the action of interest. Our method is validated on the CMU dataset [8] and a new dataset created from the i-LIDS database [9] featuring unstaged actions performed under natural settings with large number of moving people co-existing in the scene. Our results demonstrate that the proposed action detector trained with minimal annotation can achieve comparable results to that learned with full annotation, and outperforms existing methods.

1.1 Related Work

Very few existing works on action analysis are concerned with action detection in crowd. Among them, most are focused on template matching using a single training clip [8, 19]. Ke *et al.* [8] use a semi-supervised part based template for action detection. While only a single training sample per action class is required, user interaction including manual segmentation of body parts is needed for developing a part based shape-flow model. The method proposed by Yang *et al.* [19] also requires only a single manually annotated clip for constructing an

action template. To compensate for the lack of training data, an auxiliary annotated action dataset that does not contain the action of interest is needed. These two template matching methods, while requiring light manual annotation, cannot handle large intra-class variations due to the limited training data and lack of statistical modelling of data distribution. In contrast, our method also requires only one clip being manually annotated, but can capture the variation in an action class via learning from a large set of training samples.

The most closely related work is Hu *et al.* [17] which proposes a simulated annealing multiple instance learning support vector machine (SMILE-SVM) for learning a discriminative action classifier. Similar to our method, their method is also based on a sliding window strategy. It limits manual annotation by only requiring the approximate head location of the person performing the action and not the region (window) in which the action occurs. They argue that head detection algorithms can be used to automate the annotation of the training set. However, head detection in a crowded environments is far from being solved. Furthermore, they rely on foreground appearance features obtained by background subtraction for action representation. Again background subtraction is not reliable in busy environments and particularly so when it is outdoor or there are camera movements.

Another sliding-window based action detection method is presented in [18] which focus on formulating a branch and bound method for reducing the complexity of sliding window detection. Their method again relies on a large training set with manually annotated data. In contrast, the focus of this paper is on effective training of an action detector with minimal manual annotation.

2 Action Representation

Action can be represented using global features such as motion history or local features such as interest point descriptors [19]. We use local features because they are better at handling occlusions and appearance changes [19] thus more suitable for crowded environments. The different categories of local features includes volumetric descriptors computed after 3D interest point detection [20, 24, 16] which encode information in local spatio-temporal blocks, trajectory descriptors [21, 25, 22] which tracks spatial (2D) interest points over time, and flow descriptors [2, 9] which use dense optical flow information. In this paper we use the trajectory based descriptor similar to that of Sun *et al.* [25], which has been shown to perform well in complex action recognition datasets such as the Hollywood dataset [9].

2.1 Tracking Salient Points

Spatially salient points can be extracted at each frame and tracked over consecutive frames to form tracks $\{\mathbf{T}_1, \dots, \mathbf{T}_j, \dots, \mathbf{T}_N\}$. Our tracks (Fig. 2(a)) are formed by a 1-to-1 pairwise matching of SIFT descriptor over consecutive frames. We terminate any tracks where, between consecutive frames, a point travels more than 20 pixels in the x or y axes. We also restrict the length of acceptable tracks between $L_{min} = 5$ and $L_{max} = 30$ frames.

2.2 Static Appearance Descriptor

The 2D local appearance information along the track is described using the average SIFT descriptor. For a track \mathbf{T}_j of length k frames, we have a SIFT descriptor at each frame $\{S_1, S_2, \dots, S_k\}$. The static appearance information associated with the salient points tracked

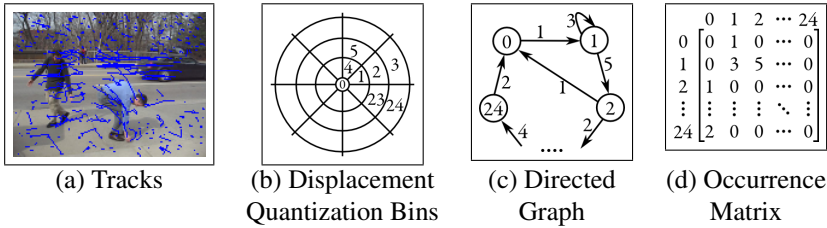


Figure 2: Trajectory Transition Descriptor (TTD). (a) Extract tracks. (a)→(b) Quantize track displacements into bins. (b)→(c) Sequential bin transitions along a track is transformed into a directed graph. (c)→(d) Occurrence matrix of the directed graph.

by \mathbf{T}_j is then represented as an average SIFT descriptor computed as:

$$S_j = \frac{1}{k} \sum_{i=1}^k S_i. \quad (1)$$

2.3 Motion Descriptor

The motion information captured by each track is represented using a trajectory transition descriptor (TTD). Specifically, we transform a trajectory vector $\mathbf{T} = [(x_1, y_1, f_1), (x_2, y_2, f_2), \dots, (x_k, y_k, f_k)]$ to a displacement vector $\mathbf{D} = [(x_2 - x_1, y_2 - y_1), (x_3 - x_2, y_3 - y_2), \dots, (x_k - x_{k-1}, y_k - y_{k-1})]$, where (x_i, y_i) is the spatial location of the track \mathbf{T} at frame f_i . To achieve scale invariance, the magnitude of the displacement vector is normalized by the largest displacement magnitude $\|\mathbf{D}\|_{max}$ along the same trajectory.

Each of the $k - 1$ displacements in vector \mathbf{D} is then quantized in magnitude and angle and assigned to one of the 25 bins depicted in Fig. 2(b). After quantizing, the displacement vector becomes $\hat{\mathbf{D}} = [b_1, b_2, \dots, b_{k-1}]$, where b_i is the quantization bin number. The sequential bin numbers in vector $\hat{\mathbf{D}}$ are used to construct a directed graph (Fig. 2(c)) which is then modelled as an ergodic Markov chain. The nodes of the graph corresponds to one of the 25 quantization states and the edge weights corresponds to the number of transition between the states. To follow the ergodic Markov chain formulation we initialize all edge weights in the directed graph with a negligible weight of 0.15.

The occurrence matrix representation of the connected graph (Fig. 2(d)) is row normalized to construct the transition matrix \mathbf{P} of the ergodic Markov chain. The motion descriptor of track \mathbf{T}_j is then computed as the stationary distribution vector π_j of the Markov chain, which can be obtained as the column average of matrix \mathbf{A}_n defined as:

$$\mathbf{A}_n = \frac{1}{n+1} (\mathbf{I} + \mathbf{P} + \dots + \mathbf{P}^n), \quad (2)$$

when $n \rightarrow \infty$ (\mathbf{I} is the identity matrix). n is set to 50 in this work to compute an approximate value of π_j .

2.4 Feature Channel Selection

The average SIFT S_j (Eq. 1) and motion descriptor π_j is then used to construct a bag of word (BoW) model for action representation with 1000 words for each type of descriptor. To take advantage of the spatial distribution of the tracks, we use all six spatial grids (1×1 , 2×2 ,

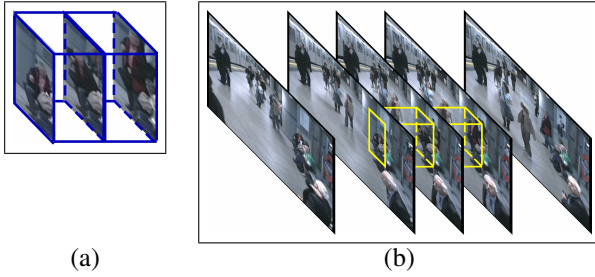


Figure 3: Action detection. (a) Action cuboid. (b) Action localized spatially and temporally.

3×3 , $h3 \times 1$, $v3 \times 1$, and $o2 \times 2$) and the four temporal grids ($t1$, $t2$, $t3$, and $ot2$) proposed in [9]. The combination of the spatial and temporal grids gives us 24 channels each for S_j and π_j , yielding a total of 48 channels. We employ the greedy channel selection routine, similar to [9], to select at most 5 of the 48 channels because these 48 channels contain redundant information. We start with the null set ($C_0 = []$) and add a channel c such that the cross validation average precision (AP) of $C_{i+1} = C_i \cup c$ is the maximum over all available c . Then we remove a channel c from C_{i+1} if AP of C_{i+1} is less than $C_{i+1} \setminus c$. We repeatedly add and remove channels till the size of C_i reaches 5. The channel combination C_i with the best overall AP is then selected as the best channel combination. Note that unlike [9] and [10], these channels are selected using cross validation on the training data rather than validation on the testing data.

3 Detection

An action is contained within a 3D spatio-temporal cuboid or window illustrated in Fig. 3(a) which we call the action cuboid. The action cuboid is represented by the multi-channel BoW histogram of all features contained within the cuboid. Our task is to slide a cuboid spatially and temporally over a video and locate all occurrences of the action in space and time (see Fig. 3(b)). To determine whether a candidate action cuboid contains the action of interest, a Support Vector Machine (SVM) classifier is learned.

More specifically, we construct a positive training set of manually annotated action cuboids and a negative training set of videos without the actions from which random cuboids can be selected. An SVM is then trained using the multi-channel χ^2 kernel [9]:

$$K(H_i, H_j) = \exp \left[- \sum_{c \in C} \frac{1}{A_c} D_c(H_i^c, H_j^c) \right] \quad (3)$$

where $H_i^c = \{h_{in}^c\}$ and $H_j^c = \{h_{jn}^c\}$ are two histograms extracted in channel c , A_c is the normalization parameter [9] and

$$D_c(H_i^c, H_j^c) = \frac{1}{2} \sum_{n=1}^{N_c} \frac{(h_{in}^c - h_{jn}^c)^2}{h_{in}^c + h_{jn}^c}. \quad (4)$$

The SVM is trained using the hard negative mining procedure outlined in [9]. At each iteration, an SVM is trained to select the optimal channel combination (Section 2.4). The positive cuboids are scored using the trained SVM and the scores are normalised, via logistic regression, to between -1 and 1. Subsequently we run the trained SVM on all the negative

videos and add any cuboids with a normalized SVM score greater than -0.75 to the negative training set. In addition to the hard negative mining, we perform a positive mining to relax the reliance on the ground truth annotation. Specifically, at each iteration, we run the trained SVM on all the positive locations within a search region created by increasing a ground truth action cuboid by 25% in all three dimensions. The cuboid with the maximum score at each search region is then added to the training set. With the increased positive and negative sets, the SVM is re-trained. This iterative mining and re-training process is terminated when the detection performance on a validation set stops improving.

During testing we slide a cuboid along the spatial and temporal axis (with overlap) and obtain a score for each cuboid using the trained SVM. Once the score at all locations are obtained, we iteratively find the maximum score in the video as an action cuboid. To avoid repeated detection of the same cuboid, we set all scores within a found action cuboid to $-\infty$. In addition, if two detected cuboids overlaps more than 25% we disregard the cuboid with the lower score as a duplicate detection. The overlap percentage is obtained as the intersection volume divided by the union volume.





4 Automated Annotation of Training Data

Our training set consists of a set of video clips without the action $V_{i=1\dots N_1}^-$, clips known to contain at least one instance of the action $V_{i=1\dots N_2}^+$ and one clip which has been manually annotated by a cuboid $Q_{k=1}$ around the action. From V_i^- and V_i^+ a set of spatially overlapping instances or cuboids $X_{i,j}^-$ and $X_{i,j}^+$ are extracted respectively, where $j = 1 \dots M_i$ are the number of cuboids in video i . Our objective is to automatically select positive cuboids/instances $X_{i,j=j^*}^+$ from each video i in the positive clip set V_i^+ to add to the positive cuboid set Q_k which initially contains only the annotated positive cuboid $Q_{k=1}$. This is effectively a multi-instance learning (MIL) problem [4]. However, the unique nature of our problem poses serious challenges to a conventional MIL method, that is, within each positive clip there are much more negative instances (typically in the order of thousands) than positive ones (typically less than 5). Furthermore, these negative instances contain many background activities that are visually similar to the action of interest, thus can be easily mis-detected as positive instances by the conventional MIL algorithms, as will be shown in our experiments in Sec. 5. To solve this problem, we formulated a simple yet effective solution to the MIL problem with an assumption that a single positive instance has been annotated.

Our solution is based on a greedy k nearest neighbour (kNN) algorithm. A kNN classifier [4] represents classes by a set of known samples (Q_k for action and $X_{i,j}^-$ for non-action) and classifies a new sample ($X_{i,j}^+$) by checking the class of the closest known sample. kNN classifier accuracy increases as the number of known samples increases. Therefore, we want to iteratively grow the positive action cuboid set Q_k such that at each iteration we capture more of the intra-class variation and the kNN classifier becomes stronger. To that end, at each iteration we use the current kNN classifier to select one cuboid X_{i^*,j^*}^+ from the set of all available cuboids $X_{i,j}^+$ which is the closest to the action class represented by Q_k . To do this, (i^*, j^*) is selected as

$$\{i^*, j^*\} = \arg \min_{i,j} \left[\min_k d(X_{i,j}^+, Q_k) - \min_{l,m} d(X_{i,j}^+, X_{l,m}^-) \right], \quad (5)$$

Table 1: Average Precision



	CMU						i-LIDS		
	Push Button	1 Hand Wave	Pick up	Jumping Jacks	2 Hand Wave	MAP	Sit Down	Stand Up	MAP
AC25_SVM	0.41	0.42	0.56	0.22	0.57	0.44	0.14	0.03	0.08
AC50_SVM	0.82	0.58	0.75	0.46	0.64	0.65	0.22	0.16	0.19
ACAuto_SVM	0.94	0.45	0.68	0.36	0.54	0.59	0.21	0.11	0.16
FV_SVM	0.09	0.11	0.46	0.26	0.48	0.28	0.04	0.03	0.04
ACML_SVM	N/A	N/A	N/A	N/A	N/A	N/A	0.06	0.02	0.04
Single Training Video									
	0.48	0.33	0.50	0.26	0.61	0.44			
	0.95	0.56	0.70	0.54	0.46	0.64			
	0.32	0.17	0.14	0.85	0.67	0.43			
KTH dataset used as training set									
	N/A	N/A	N/A	N/A	0.58	N/A			

where $d(X, Y)$ is the distance function.

$$d(X, Y) = \sum_{c \in \{c_{1 \times 1 \times t1}^{SIFT}, c_{1 \times 1 \times t1}^{TTD}\}} D_c(H_X^c, H_Y^c), \quad (6)$$

where $D_c(H_X^c, H_Y^c)$ is defined in Eq. 4 and $\{c_{1 \times 1 \times t1}^{SIFT}, c_{1 \times 1 \times t1}^{TTD}\}$ are the BoW histogram for both the average SIFT and TTD, obtained using the $1 \times 1 \times t1$ grid representation. After adding X_{i^*, j^*} to Q_k our kNN classifier is automatically updated. In order to reduce the risk of including false positive cuboids into our positive training set Q_k , we take a very conservative approach, that is from each positive clip we only select one positive cuboid. Specifically, all other cuboids $X_{i^*, j \neq j^*}^+$ from video i^* are removed from set $X_{i, j}^+$ before the next iteration.

5 Experiments

Datasets – We test our algorithm on the CMU action detection dataset  and a new action detection dataset extracted from the i-LIDS database . The CMU dataset contains 5 actions (Fig. 4(f)) in 46 clips totalling 14 mins of footage at 30 fps (160×120 pixels). Half of each action clips were used for training and the rest with other action clips for testing. The new i-LIDS action detection dataset was constructed from the i-LIDS underground train station videos to demonstrate action detection in a non-staged busy environment. i-LIDS dataset has 2 actions, Stand-up and Sit-down (see Fig. 1(c)) in 40 clips totalling 10 min of footage at 25 fps (320×240 pixels). Among the 40 clips, 11 clips (5 mins) contain busy activities of underground passengers without sit down or stand up actions. The remaining 29 clips have 24 stand up and 26 sit down instances. Half of them were used for training and half for testing. Of the 11 non-action clips in our i-LIDS dataset we used only 4 clips in training and the remaining 7 were used for testing. Note that the i-LIDS dataset is much more challenging than the CMU dataset because the actions were unstaged and more importantly featured with much more busy background activities.

Settings – Feature channel selection (Section 2.4) and SVM regularization parameter selection were obtained using a 3-fold cross validation. Since our feature space is sparse, the sliding window was moved at steps of 15 pixels in both spatial axes and 10 frames in the temporal axis. For performance evaluation we plot the precision-recall curve and also compute the mean average precision (MAP). The summary of our experiments can be found in Table 1, Fig. 4, and Fig. 5.

Automated vs. Manual Annotation – We first compare our automated annotation method described in Sec. 4 (ACAuto_SVM) to a detector using a fully annotated training set (AC50_

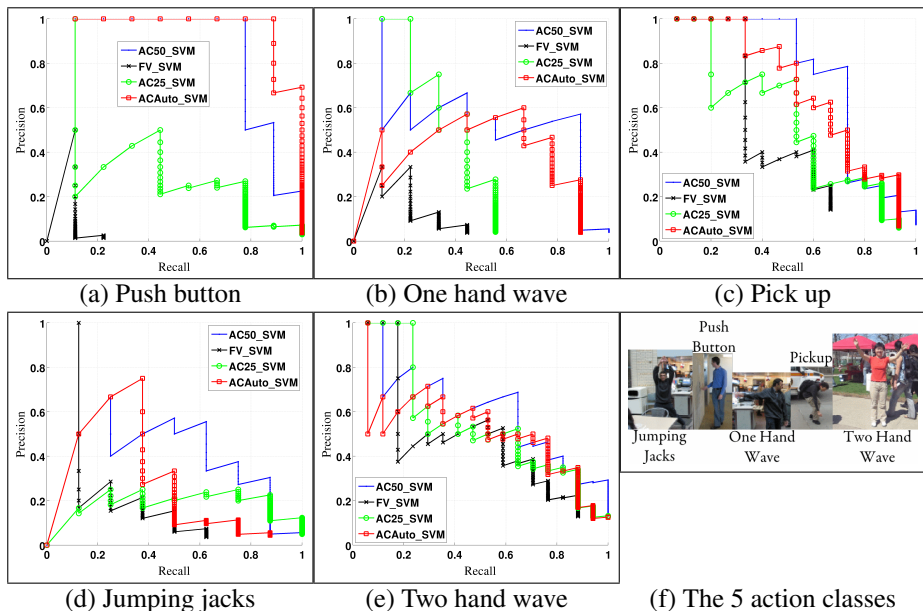


Figure 4: Action detection performance on CMU dataset.

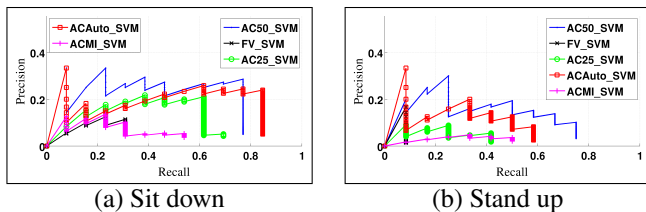


Figure 5: Precision recall curves for i-LIDS dataset.

SVM). As seen in Table 1 for both the CMU and i-LIDS datasets, comparable results are obtained using our automated annotation detector. This drop in performance is small considering that AC50_SVM used 10 times more manually annotated data than ACAuto_SVM. We also note that i-LIDS dataset has a much lower overall performance because it is a much more challenging dataset containing larger intraclass variation which makes training difficult and a busier environment with sever occlusions and complex background activities. We illustrate some of detection results in Fig. 6 which highlight the difficult nature of the problem.

Greedy kNN vs. Standard MIL – We test our greedy kNN solution of the MIL problem to the popular MI-SVM formulation [2] implemented by Yang [18] using the i-LIDS dataset. Table 1 shows that the detector trained using our greedy kNN annotation (ACAuto_SVM) performs 4 times better than the detector using the MI-SVM annotation (ACMI_SVM). This is because that the MI-SVM mis-detects those visually similar background actions as the actions of interest. As is evident when averaged over the i-LIDS dataset, the greedy kNN annotated action cuboids has a 43% overlap with manual annotation but the MI-SVM annotation only has a 11% overlap.

Automated Annotation vs. Smaller Training Set – We also train a detector (AC25_SVM) using a smaller manually annotated training set (25% of original dataset size) and com-

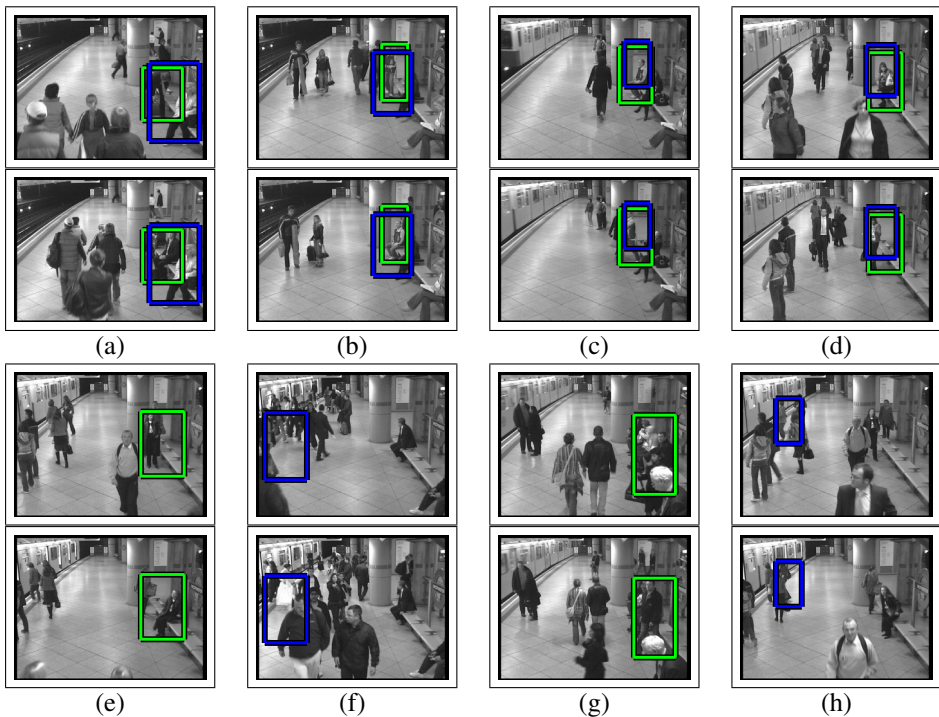


Figure 6: Detection results on i-LIDS dataset: Green - ground truth, blue - detection by ACAuto_SVM. (a-d) True detection. (e,g) Missed detection. (f,h) False detection.

pare that to using our automated annotation of training set (ACAuto_SVM). We observe using automated annotation has an 34% and 100% increase in MAP on CMU and i-LIDS datasets respectively. Note that AC25_SVM uses 5 times more manually annotated data than ACAuto_SVM yet performs poorly, showing the effectiveness of our proposed automated annotation algorithm.

Automated vs. No Annotation –If we can perform action detection without spatial localization (i.e. treating it as an action recognition rather than action detection problem), we can reduce the amount of manual annotation needed. To test this we train an SVM (FV_SVM) using full frame information and run our detection as a sliding window search only in the temporal axis. As expected FV_SVM achieves a poor MAP in particular for the i-LIDS dataset with more complex background activities (see Table 1).

Comparison to Alternative Approaches on CMU dataset– We compared our results on the CMU dataset with those reported by the existing 4 action detection methods in Table 1. It can be seen that our method significantly outperform the 2 template matching based method in [8, 20]. Our result is slightly inferior to that of [2]. However, it is worth pointing out that their method relies on head detection and background subtraction. None of them is a problem for the CMU dataset where in most video clips there were not many moving people in the background and the target actions occupied much of the camera view. However, for real world surveillance scenes in the i-LIDS dataset, where the actions of interest were performed in the mix of large number of passing-bys and were much smaller in scale and under occlusion, the method in [2] will not work because neither face detection nor background subtraction can be performed reliably.

6 Conclusion

We have presented a novel approach for action detection in crowded environments. Our method can be seen as an extension of the existing 2D object detection approaches to the problem of 3D action detection. To reduce manual annotation while preserving large training set size to cope with intra-class variation, we presented a greedy kNN algorithm for automated annotation of positive training data using a single manually annotated clip. Extensive experiments on real-world action detection datasets demonstrate that our detector trained with minimal annotation can achieve comparable results to that learned with full annotation, and outperforms existing methods.

References

- [1] S. Ali and M. Shah. Human action recognition in videos using kinematic features and multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32:288–303, 2010. ISSN 0162-8828.
- [2] S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In *Advances in Neural Information Processing Systems 15*, pages 561–568. MIT Press, 2003.
- [3] C.M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1st ed. 2006. corr. 2nd printing edition, October 2007.
- [4] A.A. Efros, A.C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *IEEE International Conference on Computer Vision*, pages 726–733, Nice, France, 2003.
- [5] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 99, 2009.
- [6] HOSDB. Imagery library for intelligent detection systems (i-lids). In *IEEE Conf. on Crime and Security*, 2006.
- [7] Y. Hu, L. Cao, F. Lv, S. Yan, Y. Gong, and T.S. Huang. Action Detection in Complex Scenes with Spatial and Temporal Ambiguities. In *IEEE International Conference on Computer Vision*, 2009.
- [8] Y. Ke, R. Sukthankar, and M. Hebert. Event detection in crowded videos. In *IEEE International Conference on Computer Vision*, October 2007.
- [9] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.
- [10] J.G. Liu, J.B. Luo, and M. Shah. Recognizing realistic actions from videos 'in the wild'. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1996–2003, 2009.
- [11] P. Matikainen, M. Hebert, and R. Sukthankar. Trajectons: action recognition through the motion analysis of tracked features. In *IEEE International Conference on Computer Vision*, 2009.

- [12] J.C. Niebles, H.C. Wang, and F.F. Li. Unsupervised learning of human action categories using spatial-temporal words. *International Journal of Computer Vision*, 79(3), September 2008.
- [13] R. Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976 – 990, 2010.
- [14] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local svm approach. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 3, pages 32–36, August 2004.
- [15] J. Sun, X. Wu, S. Yan, L.F. Cheong, T.S. Chua, and J. Li. Hierarchical spatio-temporal context modeling for action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2004–2011. IEEE, 2009.
- [16] G. Willems, T. Tuytelaars, and L.J. Van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In *European Conference on Computer Vision*, pages II: 650–663, 2008.
- [17] X. Wu, C.W. Ngo, J. Li, and Y. Zhang. Localizing volumetric motion for action recognition in realistic videos. In *ACM Multimedia (ACM MM)*, Oct 2009.
- [18] J. Yang. Mill: A multiple instance learning library. URL <http://www.cs.cmu.edu/~juny/MILL>.
- [19] W. Yang, Y. Wang, and G. Mori. Efficient human action detection using a transferable distance function. In *Asian Conference on Computer Vision*, 2009.
- [20] J.S. Yuan, Z.C. Liu, and Y. Wu. Discriminative subvolume search for efficient action detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2442–2449, 2009.