AT&T Research at TRECVID 2013: Surveillance Event Detection

Xiaodong Yang[†*], Zhu Liu[§], Eric Zavesky[§], David Gibbon[§],
Behzad Shahraray[§], YingLi Tian[†]

[†]City College of New York, CUNY
160 Convent Ave, New York, NY 10031
{xyang02, ytian}@ccny.cuny.edu

[§]AT&T Labs – Research
200 Laurel Ave, Middletown, NJ 07748
{zliu, ezavesky, dcg, behzad}@research.att.com

Abstract. This paper presents a general surveillance event detection system that participated in the interactive Surveillance Event Detection (iSED) task of TRECVID 2013. In the proposed system, a set of spatio-temporal features including Space-Time Interest Points (STIP) and Dense Trajectories are extracted, and a sliding temporal window is employed as the detection unit. Fisher Vector is used to encode low-level features as the representation of each sliding window. Both feature-level and decision-level fusions are used to combine multiple features. In order to deal with the highly imbalanced nature of surveillance data, the system performs detections using the CascadeSVMs algorithm according to each specific event and camera view. Two different interactive environments are evaluated, one focuses on high throughput and the other includes related result expansion. In the primary run evaluations, our system ranks the top in 2 out of 7 event detection tasks.

1. Introduction

Automatic event detection of video surveillance has many real-world applications for home security (e.g., AT&T Digital Life) and public security (e.g., IBM Smart Surveillance Solutions). In the past decades, most research of human activity analysis mainly experimented on relatively simple and clear scenes where only a limited number of actors perform explicit actions. This constrained scenario seldom holds in real-world surveillance videos due to the great challenges of large variances of viewpoint, scaling, lighting, cluttered background, etc. In order to bridge research efforts and real-world applications, TRECVID [8] provides the interactive Surveillance Event Detection (iSED) task to evaluate event detection in real-world surveillance settings. In TRECVID 2013 [13], iSED provides a corpus of 144-hour videos under five camera views from the London Gatwick International Airport. In this dataset, 99-hour videos can be used as the development set with annotations of temporal extents and event labels. Our system is evaluated on all the seven events,

[*]This work is carried out when the author worked as a research intern at AT&T Labs – Research.

i.e., CellToEar, Embrace, ObjectPut, PeopleMeet, PeopleSplitUp, PersonRuns, and Pointing.

The remainder of this paper is organized as follows. Section 2 introduces the overall system architecture. In Sections 3 and 4, we provide detailed procedures of feature extraction and video representation. Section 5 describes the learning algorithm and feature fusion. The interactive design is presented in Section 6. A variety of experimental results and discussions are presented in Section 7. Finally, Section 8 summarizes the approaches and performance of the system.

2. System Overview

As demonstrated in Fig. 1, the system consists of 4 major components: (1) low-level feature extraction, (2) video (sliding window) representation based on Fisher Vector, (3) event learning and prediction by CascadeSVMs, and (4) human interaction.

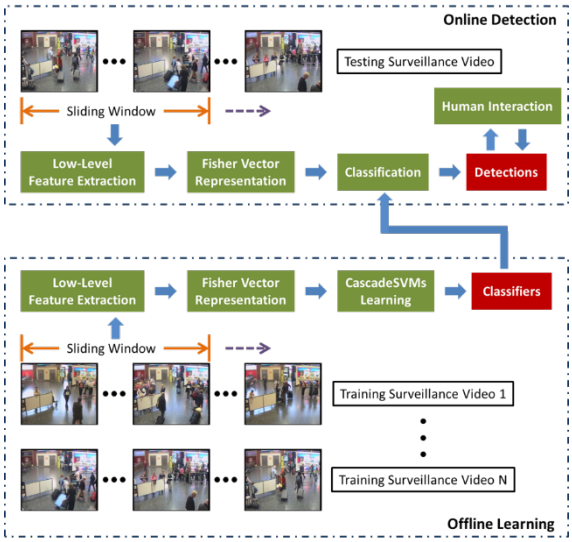

Figure 1: AT&T Research surveillance event detection system overview.

Most recent work on action recognition demonstrates that local spatio-temporal features are more robust to posture, occlusion, illumination, and cluttered background compared to global features. A spatio-temporal feature extraction usually includes two phases: detection, i.e., a feature detector localizes interest points in a spatio-temporal space; and description, i.e., a feature descriptor computes representations of detected points. Space-Time Interest Point (STIP) [4] employs 3D Harris corner detector to detect sparse points with large gradient magnitude in both spatial and temporal domains. Histogram of Gradients (HOG) and Histogram of Optical Flow (HOF) are then computed and concatenated as descriptors. However, it is restrictive to have large intensity changes in both spatial and temporal dimensions, which could result in insufficient detections. On the other hand, dense sampling has shown to

improve action recognition over the state-of-the-art sparse interest point detectors [10]. Dense Trajectories (DT) [9] densely samples interest points at multiple spatial scales. The sampled points are tracked over a dense optical flow field and reinitialized every few frames. Several local descriptors, e.g., Trajectory and Motion Boundary Histogram (MBH), are then extracted from the space-time volumes aligned with the trajectories. In this system, STIP-HOG/HOF, DT-Trajectory, and DT-MBH are used as the low-level features to characterize human actions.

After local feature extraction, feature encoding is commonly used to aggregate the low-level features to represent images and videos. This can be done in two ways [1]: (1) by pooling the coded visual words (e.g., soft quantization with average pooling, sparse coding with max pooling) and (2) by recording the differences between features and visual words (e.g., Fisher kernel encoding, super vector encoding). The relative displacements between descriptors and visual words capture the feature distribution information that helps to retain some information lost in the quantization process. An evaluation of recent feature encoding methods in [1] identifies the superiority of Fisher Vector in image classification. In this system, Fisher Vector is employed with spatial pyramids [6] to encode local spatio-temporal features.

Having obtained the above video representations, the event models can be learned by linear SVMs solvers [3]. However, the surveillance data is highly imbalanced because positive events are far less frequent than negative ones, e.g., the sequences of CellToEar are only 0.31% of the entire video. The CascadeSVMs proposed by Yang et al. [11] is used to handle this difficulty. In order to combine multiple features, we employ both feature-level and decision-level fusions. A simple post processing is performed over the positive classification results to determine temporal localization of each event and further remove false alarms. It is observed that most positive samples continuously last for a number of frames as temporal extents of most events cover several sliding windows. So neighboring positive predictions are grouped into a merged detection, which is assigned a higher confidence score than those isolated positive predictions.

Contrary to typical search systems employed in TRECVID evaluations, a triage-based interaction style is utilized in these experiments. In this style, results from an automatic baseline are ingested into the system and roughly explored based on the order of decreasing likelihood. Two slightly different interactive systems were evaluated: a *linear* interface focusing purely on in-order and single result visualization as well as a *non-linear* interface that also visualizes temporally adjacent results and the entire list of results to be evaluated. Several detection permutations are explored in submissions using interaction output of the linear interface and different decision thresholds.

3. Low-Level Feature Extraction

We extract a set of local spatio-temporal features including STIP-HOG/HOF [5], MoSIFT [2], ActionHOG [11], DT-Trajectory [9], DT-HOG [9], DT-HOF [9], and DT-MBH [9]. Based on empirical evaluations of single features and their combinations, STIP-HOG/HOF, DT-Trajectory, and DT-MBH are kept in the final evaluation.

3.1. STIP-HOG/HOF

STIP detector combined with HOG/HOF descriptors has been widely used in action recognition and detection tasks [5]. STIP detects interest points by searching significant variations in both space and time. The second moment matrix at each spatio-temporal point is given by $\mu(\cdot;\sigma,\tau) = g(\cdot;s\sigma,s\tau) * (\nabla L(\cdot;\sigma,\tau)\nabla L(\cdot;\sigma,\tau)^T)$; where σ and τ are spatial and temporal scales, g is a Gaussian smoothing function, and ∇L is space-time gradient. The detected interest points correspond to the local maxima of $H = \det(\mu) - k\mathrm{trace}^3(\mu)$. A dense sampling of the spatio-temporal scales (σ,τ) is used instead of performing the scale selection as in [4]. This has been shown to give promising recognition performance and reduce computational cost [5].

HOG/HOF descriptors are computed based on the space-time neighborhoods of detected interest points to capture the local appearance and motion information. The neighborhood size $(\Delta_x, \Delta_y, \Delta_z)$ is defined by $\Delta_x = \Delta_y = 18\sigma$ and $\Delta_t = 18\tau$. Each STIP volume is subdivided into $3\times3\times2$ cuboids. Each cuboid generates a 4-bin histogram of gradient orientation and 5-bin histogram of optical flow. The histograms are then normalized and concatenated as the HOG/HOF descriptor.

3.2. DT-Trajectory

Dense Trajectories [9] provides an alternative to the joint space-time based interest point detectors. It is motivated by the difference between 2D space domain and 1D time domain in videos, as well as the success of dense sampling over sparse detection in image classification. As suggested in [9], interest points are densely sampled on a grid spaced by 5 pixels at 8 spatial scales spaced by a factor of $1/\sqrt{2}$. The sampled points are tracked by median filter in a dense optical flow field. A trajectory is removed once it reaches a length of 15 frames to avoid drifting in the tracking process. In order to assure dense sampling, a new trajectory is initialized if there is no tracked point within a 5×5 neighborhood. The trajectories with large sudden displacements and static trajectories are pruned.

DT-Trajectory characterizes the shape of a trajectory that is used to capture local motion cues. As each trajectory has a fixed length of 15, its shape is described by a sequence $(\Delta P_t, \ldots, \Delta P_{t+14})$ of point displacements $\Delta P_t = (P_{t+1} - P_t) = (x_{t+1} - x_t, y_{t+1} - y_t)$. DT-Trajectory is finally represented as the normalized sequence vector $(\Delta P_t, \ldots, \Delta P_{t+14}) / \sum_{i=1}^{t+14} \|\Delta P_t\|$.

3.3. DT-MBH

The space-time volume aligned with a trajectory is used to extract local descriptors. The size of a volume is 32×32 pixels and 15 frames. The volume is subdivided into a grid of $2\times2\times3$ spatio-temporal bins. In order to suppress camera motion induced by optical flow, MBH is employed in [9] to separately compute spatial derivatives $(I_{xx}, I_{xy}, I_{yx}, I_{yy})$ of horizontal and vertical components (I_x, I_y) of optical flow I. The motion boundaries generated by the gradients $(I_{xx}, I_{xy}, I_{yx}, I_{yy})$

of the two separated optical flow components (I_x, I_y) reduce most camera motion in background and highlight foreground motion. Similar to the HOG descriptor, gradient orientations of I_x and I_y are quantized into 8-bin histograms that are then normalized using L2 norm.

4. Video Representation

We employ Fisher Vector [7] combined with spatial pyramids [6] to represent each sliding window. Fisher Vector provides a feature aggregation scheme based on Fisher kernel that shares the benefits of both generative and discriminative models. Fisher Vector describes each feature descriptor by its deviation with respect to the parameters of a generative model. The spatial pyramids are then used to roughly incorporate the spatial layout of the video scene.

4.1. Fisher Vector

Fisher Vector chooses the Gaussian mixture model (GMM) as the generative model $U_\lambda(x) = \sum_{k=1}^{K} \pi_k u_k(x)$, u_k denotes the kth Gaussian component:

$$u_k(x) = \frac{1}{2\pi^{\frac{D}{2}} |\Sigma_k|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}(x-\mu_k)' \Sigma_k^{-1} (x-\mu_k)\right\}, \quad \forall k : \pi_k \geq 0, \ \sum_{k=1}^{K} \pi_k = 1. \tag{1}$$

where the feature descriptor $x \in \mathbb{R}^D$; K is the number of Gaussian components; π_k, μ_k, and Σ_k are the mixture weight, mean vector, and covariance matrix, respectively. Σ_k is assumed to be a diagonal matrix with the variance vector σ_k^2. The GMM parameters $\lambda = \{\pi_k, \mu_k, \Sigma_k, \ k = 1, \dots, K\}$ are estimated based upon a large set of training descriptors using the Expectation-Maximization (EM) algorithm to optimize the Maximum Likelihood (ML).

For a set of descriptors $X = \{x_1, \dots, x_N\}$ extracted from a sliding window, the soft assignment of descriptor x_i to component k is defined as:

$$\gamma_i^k = \frac{\pi_k u_k(x_i)}{\sum_{j=1}^{K} \pi_j u_j(x_i)}. \tag{2}$$

The Fisher Vector of X is then represented as $\Psi(X) = \{\rho_1, \tau_1, \dots, \rho_K, \tau_K\}$, where ρ_k and τ_k are D-dimensional gradients with respect to mean vector μ_k and standard deviation σ_k of the kth component:

$$\rho_k = \frac{1}{N\sqrt{\pi_k}} \sum_{i=1}^{N} \gamma_i^k \left(\frac{x_i - \mu_k}{\sigma_k}\right), \tag{3}$$

$$\tau_k = \frac{1}{N\sqrt{2\pi_k}} \sum_{i=1}^{N} \gamma_i^k \left[\frac{(x_i - \mu_k)^2}{\sigma_k^2} - 1 \right]. \quad (4)$$

Compared to the Bag-of-Words (BoW) based approaches, Fisher Vector has several advantages: (1) BoW is a particular case of Fisher Vector where only the gradient to the mixture weights of GMM is utilized. The additional gradients with respect to mean vectors and standard deviations in Fisher Vector provide extra distribution information of descriptors. (2) We can compute Fisher Vector upon a much smaller vocabulary that facilitates a lower computational cost. (3) Fisher Vector performs well with simple linear classifiers that are efficient in terms of both training and testing.

This system follows the two schemes introduced in [7] to normalize Fisher Vector, i.e., L2 normalization and power normalization. The L2 normalization is proposed to remove the dependence on the proportion of event-specific information contained in a video, in other words, to cancel the effect of different amount of foreground/background information contained in different video segments. The power normalization is motivated by the observation that, as the number of Gaussian components increases, Fisher Vector becomes more peaky around zero in a certain dimension, which negatively impacts the dot-product in the following steps. The power normalization $f(z) = \text{sign}(z)|z|^{\alpha}$ with $0 < \alpha \leq 1$ is applied to each dimension z in Fisher Vector. We choose $\alpha = 0.5$ (the Hellinger kernel) to perform the signed square-rooting operation. In this system, the power normalization is first applied and then the L2 normalization.

4.2. Spatial Pyramid

The spatial pyramid [6] is applied to capture the rough geometry of a video scene. It spatially subdivides a video into a set of regions where statistics of low-level descriptors are pooled. This system uses the 3-level spatial pyramids with 1×1, 3×1, and 2×2 grids as illustrated in Fig. 2. Fisher Vector is computed from each grid and they are concatenated as the video representation. The temporal pyramids introduced in [5] are not used due to the explosion of feature dimension and memory requirement.

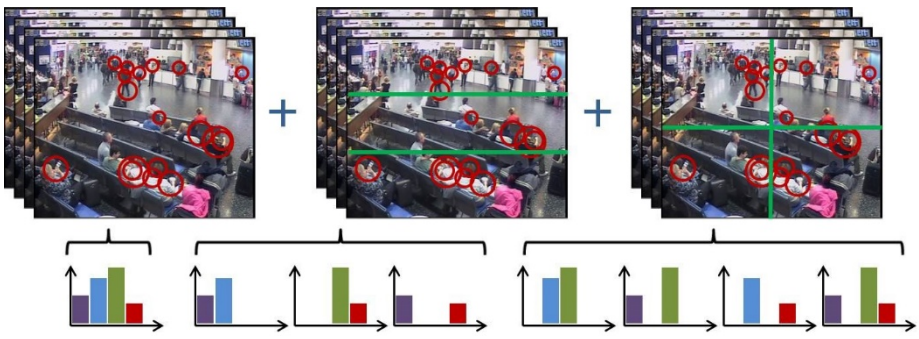

Figure 2: The spatial pyramids of 3 levels with 1×1, 3×1, and 2×2 grids. Fisher Vectors are generated from each grid and concatenated as the final video representation.

5. Model Learning and Feature Fusion

The sliding window scheme (60-frame window steps in every 15 frames) in this system generates quite imbalanced data as shown in Fig. 3. Most (5/7) positive events appear in less than 2% of the entire video sequences. CellToEar and PeopleSplitUp are the least and most frequent events that are only 0.31% and 4.37% of the training video sequences, respectively. The CascadeSVMs [11] scheme is employed to overcome this imbalance. In each stage of this algorithm, the same-amount positive and negative samples are used to train a classifier that favors to the positive class. This leads the classifier in each stage to a high detection rate but a high false alarm rate as well. By cascading multiple classifiers, it is able to remove considerable false alarms but maintain a reasonable detection rate. In order to reduce intra-class variance and memory consumption, the models are learned according to each specific event and camera view. This system therefore includes 35 models of 7 events under 5 camera views.

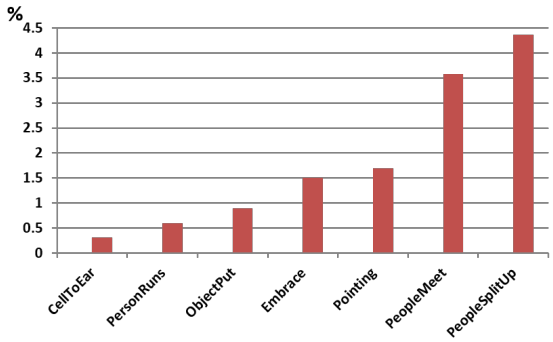

Figure 3: Proportions of video sequences containing positive events in the training set.

As introduced in Section 3, three low-level features are extracted from each sliding window, i.e., STIP-HOG/HOF, DT-Trajectory, and DT-MBH. Each low-level feature generates a corresponding Fisher Vector. We employ both early fusion (feature-level fusion) and late fusion (decision-level fusion) to combine the three feature sets, as illustrated in Fig. 4. The fusions in feature-level and decision-level are executed before and after classification, respectively. The feature-level fusion aims to combine different features to generate a new feature vector that explicitly takes account of the relationship between different feature sets. Simple concatenation, Adaboost, and Random Forests are widely used in the feature-level fusion [12]. The decision-level fusion combines outputs of classifiers to make the final prediction. Popular decision-level fusion methods include minimum, maximum, median, majority voting, weighted sum, and geometric mean [12]. In this system, the simple concatenation and weighted sum are employed for early fusion and late fusion, respectively.

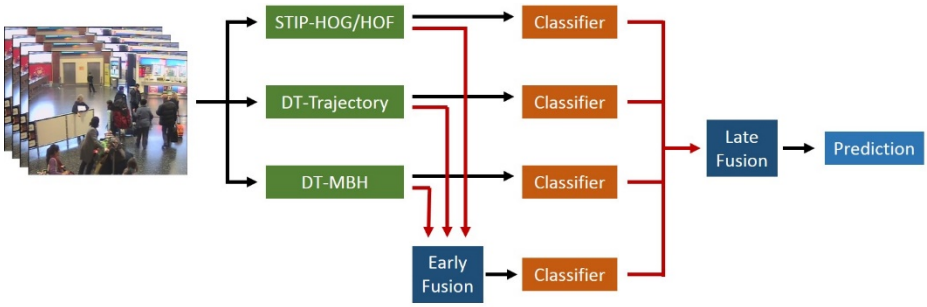

Figure 4: Illustration of early fusion and late fusion in combining multiple low-level features.

Because of the sliding window scheme used in our system, an event might span several different windows. After the classifier prediction, a simple post processing is applied to group continuous positive windows to determine the final temporal location of a detected event. In the merging process, two positive predictions disconnected by less than ω negative predictions can be merged together, where ω is set to 3 in this system. The merged detections are assigned higher scores than those isolated positive predictions in the human inspection process.

6. Human Interaction

Two different interactive environments are evaluated: one focuses on high throughput while the other also includes related result expansion. These two interactive systems are described in detail in the following subsections.

6.1. High Throughput User Interface

Fig. 5 illustrates the first user interface, which is simple yet effective in removing false events. The main goal here is to let the user focus on only one event at a time and make decision for the target event in a minimum amount of time. The event type and the progress information of the current interactive session are shown in the top row. The user is able to pause/resume the timer if necessary. The event video clip is pre-generated and played back in the middle at the original resolution. The associated information for the current event is shown above the video clip, which includes the video ID, event boundary, and machine generated score. The action label shows the manual decision about this event, and the original action is set to *INIT*. The *Previous* and *Next* buttons are mainly for browsing purposes, since in the practice, once the user makes decision on the current event, the UI will automatically moves to the next event in the pipeline.

Right below the event video are all possible actions that the user may choose. They are *Reject*, *Accept*, *ExpandLeft*, *ExpandRight*, *Split*, and *Skip*. The *Reject* and *Accept* actions let the user to reject the events from the final list and accept it with its original boundary, respectively. The *ExpandLeft* action doubles the event duration by moving its starting frame backward. Similarly the *ExpandRight* action doubles the duration by shifting its ending frame foreword. These two actions are useful for adjusting the

boundaries of the event that either starts before the detected starting frame or ends after the detected ending frame to make sure the center of the reported events overlap with the span of the actual events. When there are two isolated events found in the current window, the user can evenly split the single event into two by the *Split* action. The *Skip* action is a special action when the user cannot make decision and do not want to spend more time on the current event. Usually, the *Skip* action is considered as a *Reject* action, but it gives the user an option to revisit the difficult events later if time allows. In addition to these actions, the user can also accept the events with adjusted boundaries, for example, moving the start frame 1, 2, or 3 seconds forward, or moving the end frame 1, 2, or 3 seconds backward. All these action selections are arranged in a group, so the user can easily provide feedback for a sequence of events without much movement of the mouse.

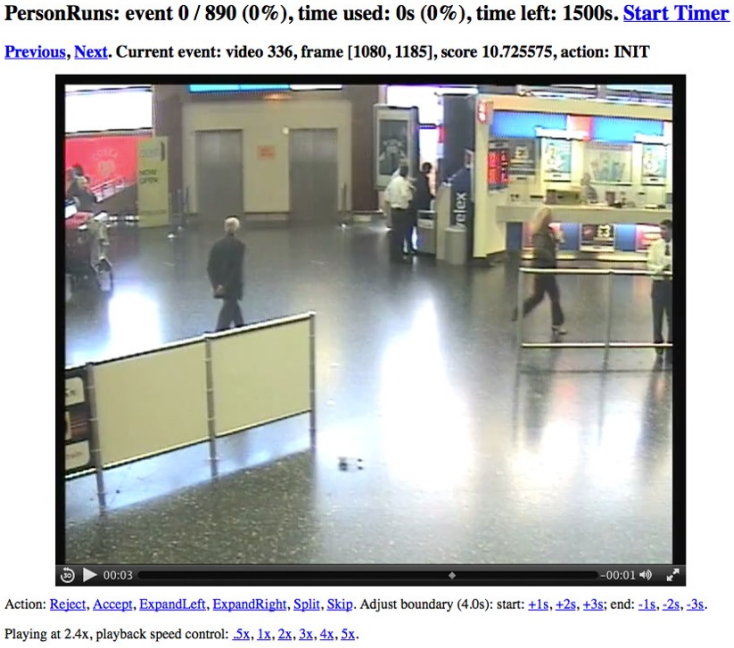

Figure 5: Illustration of the UI for high throughput.

Since the number of machine detected results is typically large, much more than the number of events that a user can go through within 25 minutes, this interface tries to increase the throughput by playing back video in a faster rate. Depending on the events, the user may choose to playback the video in a wide range of rates, from half to five times real time. The choice of the playback rate is kept for all following events unless the user decides to change it again.

6.2. Triage User Interface

When creating the interface for iSED evaluation, a *triage* interactive mode (as opposed to an *exploration* mode) is chosen for experimentation for three reasons: the

evaluation metric is very sensitive to correct and incorrect detections, the automatic detector baseline is fair, and the entire sample space is not that large. The interface is shown in Fig. 6.

Icons of all detected events are listed in the first row. The current event is displayed in the middle at a higher resolution. The user can playback the entire events at different playback rate. To simplify the interface, some of the controls (e.g., playback rate change) are implemented by keystroke instead of mouse clicking. For the current event, its temporally adjacent events are shown on the left hand side, and its associated information is presented on the right hand side. At the bottom, the recent labeled events are presented on the right hand side, and the left hand side is reserved for a collaborative labeling mode where two users can go through the results simultaneously. This collaboration mode is not fully implemented for this evaluation task.

Figure 6: Illustration of the UI for triage.

7. Experimental Results

TRECVID iSED 2013 provides 99-hour videos for development and 15-hour videos for evaluation. All videos are captured by 5 fixed cameras with the frame resolution 720×576 at 25fps. We downsample all videos to half size in the low-level feature extraction but keep the full resolution in the human interaction. We train the GMM with 128 Gaussian components and utilize the 8-grid spatial pyramid in this system. Hence, the dimensions of Fisher Vectors are 331776, 61440, and 393216 for STIP, DT-Trajectory, and DT-MBH, respectively. In the human interaction process, 25 minutes are allowed for each event. The experiments reported in this paper are performed on a server that comprises 32 cores (2.0GHz), 256GB memory, and 15TB disk.

We first compare our system to other best systems in TRECVID iSED 2013 by the primary metric Actual Detection Cost Rate (ADCR) and the secondary metric Minimum Detection Cost Rate (MDCR) in Table 1. The rank column denotes our rankings among all participants in terms of ADCR. We achieve the best performance

in two event tasks, i.e., ObjectPut and PeopleSplitUp. The Detection Error Tradeoff (DET) curves of all events are shown in Fig. 8. These curves represent event-averaged miss detection probabilities vs. false alarm rates through varying a detection threshold.

Table 1: Comparison of our system and other best systems in TRECVID iSED 2013.

Event	Rank	ADCR of Other Best Systems	AT&T Research Primary Run				
			ADCR	MDCR	#CorDet	#FA	#Miss
CellToEar	2	0.9057[1]	0.9908	0.9904	3	19	191
Embrace	4	0.6540[1]	0.7540	0.7439	50	121	125
ObjectPut	1	0.9889[1]	**0.9806**	0.9803	21	44	600
PeopleMeet	3	0.8704[2]	0.9181	0.9115	44	49	405
PeopleSplitUp	1	0.8484[2]	**0.7781**	0.7771	64	367	123
PersonRuns	4	0.5850[1]	0.7508	0.7244	36	266	71
Pointing	2	0.9564[2]	0.9659	0.9655	53	48	1010

[1]the result attributes to CMU, [2]the result attributes to BUPT-MCPRL

Because of the time constraint (25 mins) for each event detection task in the interactive process, a user is not able to verify all of the automatic detections generated by the system. In order to investigate the effect of human interaction, we compare the results of different schemes on combining human verified samples and unprocessed ones in Fig. 7. Experiment 1 shows the result by using both human verified samples and the remaining ones. Experiment 2 is the case without human interaction. Experiment 3 corresponds to the performance based on human verified samples only. As shown in this figure, most results in Experiments 1 and 3 are improved compared to Experiment 2, especially for Embrace, PeopleMeet, and PersonRuns. This improvement clearly demonstrates the effectiveness of human interaction, i.e., (1) a number of false alarms are removed and (2) the boundary adjustment functions enforce the centers of some drifted detections fall into the ground truth range. Experiments 1 and 3 achieve comparable results in most events. The results of PersonRuns and Pointing in Experiment 3 slightly outperform the ones in Experiment 1 because most of the unprocessed samples are false alarms. The performance of PeopleSplitUp in Experiment 1 is much better than that in Experiment 3. This is because the increase of correct detections in unprocessed samples outweighs the increase of false alarms. Our primary run is based on Experiment 1.

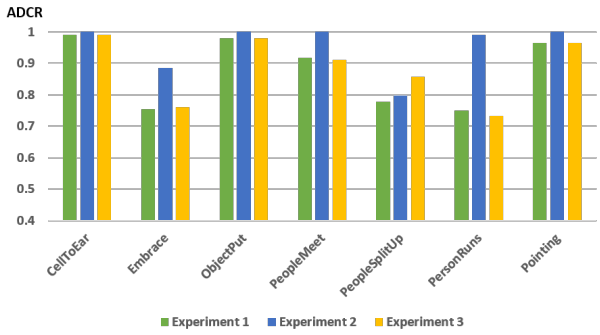

Figure 7: Different schemes on combining verified samples and unprocessed ones. The results in Experiments 1-3 correspond to using both human verified samples and the remaining ones, automatic detections without human interaction, and human verified samples only.

8. Conclusion

In this paper we have presented detailed implementation of our iSED system participated in TRECVID 2013. Our system starts from extracting low-level features of STIP-HOG/HOF, DT-Trajectory, and DT-MBH from each sliding window. Fisher Vector is then employed to aggregate the low-level features. We combine multiple features by both feature- and decision-level fusions. The CascadeSVMs algorithm is utilized to learn the detection models corresponding to the each specific event and camera view. We design two interactive environments with one focusing on high throughput and the other one including related result expansion. In the evaluations of 7 event detection tasks, our system ranks 1st in 2 events and achieves top 3 performances in 5 events.

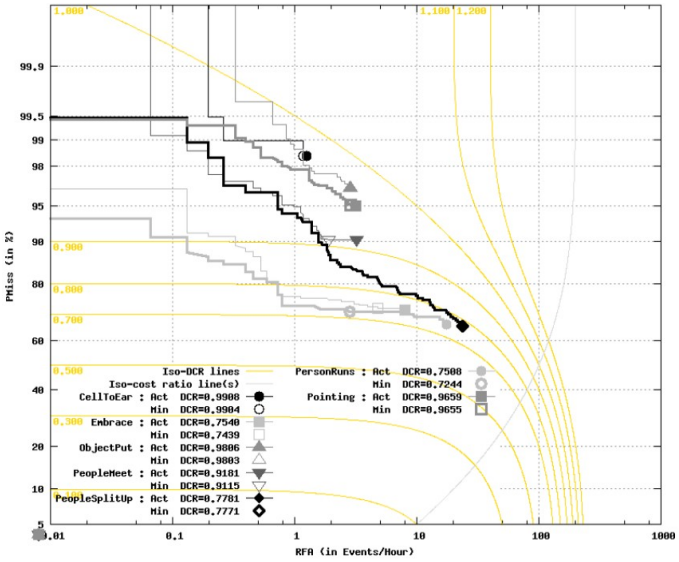

Figure 8: Detection Error Tradeoff (DET) curves of each event.

References

1. K. Chatfield, V. Lemtexpitsky, A. Vedaldi, and A. Zisserman. The Devil Is in the Details: An Evaluation of Recent Feature Encoding Methods. *British Machine Vision Conference*, 2011.
2. M. Chen and A. Hauptmann. MoSIFT: Recognizing Human Actions in Surveillance Videos. *CMU-CS-09-161*, 2009.

3. R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 2008.
4. I. Laptev. On Space-Time Interest Points. *International Journal of Computer Vision*, 2005.
5. I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning Realistic Human Actions from Movies. *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.
6. S. Lazebnik, C. Schmid, and J. Ponce. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.
7. J. Sanchez, F. Perronnin, T. Mensink, and J. Verbeek. Image Classification with the Fisher Vector: Theory and Practice. *International Journal of Computer Vision*, 2013.
8. A. Smeaton, P. Over, and W. Kraaij. Evaluation Campaigns and TRECVID. *ACM Workshop on Multimedia Information Retrieval*, 2006.
9. H. Wang, A. Klaser, C. Schmid, and C. Liu. Dense Trajectories and Motion Boundary Descriptors for Action Recognition. *International Journal of Computer Vision*, 2013.
10. H. Wang, M. Ullah, A. Klaser, I. Laptev, and C. Schmid. Evaluation of Local Spatio-Temporal Features for Action Recognition. *British Machine Vision Conference*, 2009.
11. X. Yang, C. Yi, L. Cao, and Y. Tian. MediaCCNY at TRECVID 2012: Surveillance Event Detection. *NIST TRECVID Workshop*, 2012.
12. P. Atrey, M. Hossain, A. Saddik, and M. Kankanhalli. Multimodal Fusion for Multimedia Analysis: A Survey. *Multimedia System*, 2010.
13. P. Over, G. Awad, M. Michel, J. Fiscus, G. Sanders, W. Kraaij, A. Smeaton, and G. Quéenot, "TRECVID 2013 - An Overview of the Goals, Tasks, Data, Evaluation Mechanisms and Metrics," Proceedings of TRECVID 2013.