Informedia@TRECVID 2012

Shoou-I Yu*, Zhongwen Xu*, Duo Ding*, Waito Sze*, Francisco Vicente*, Zhenzhong Lan*, Yang Cai*, Shourabh Rawat*, Peter Schulam*, Nisarga Markandaiah*, Sohail Bahmani*, Antonio Juarez*, Wei Tong*, Yi Yang*, Susanne Burger*, Florian Metze*, Rita Singh*, Bhiksha Raj*, Richard Stern*, Teruko Mitamura*, Eric Nyberg*, Lu Jiang*, Qiang Chen**, Lisa Brown**, Ankur Datta**, Quanfu Fan**, Rogerio Feris**, Shuicheng Yan***, Alex Hauptmann*, Sharath Pankanti** and Alexander Hauptmann*

*Carnegie Mellon University/**IBM Research/***National University of Singpore

In the first part of this three-part report we describe our system and novel approaches used in the TRECVID 2012 Multimedia Event Detection (MED) and Multimedia Event Recounting (MER) tasks. A separate section of the report (SIN) details methods and results for the Semantic Indexing task. The final section (SED) describes our approaches and results on the Surveillance Event Detection task.

Informedia E-Lamp @ TRECVID 2012 Multimedia Event Detection and Recounting (MED and MER)

Shoou-I Yu, Zhongwen Xu, Duo Ding, Waito Sze, Francisco Vicente, Zhenzhong Lan, Yang Cai, Shourabh Rawat, Peter Schulam, Nisarga Markandaiah, Sohail Bahmani, Antonio Juarez, Wei Tong, Yi Yang, Susanne Burger, Florian Metze, Rita Singh, Bhiksha Raj, Richard Stern, Teruko Mitamura, Eric Nyberg and Alexander Hauptmann

Carnegie Mellon University

Informedia Aurora @TREVID 2012 Semantic Indexing (SIN)

Lu Jiang, Alex Hauptmann

Carnegie Mellon University

Informedia CMU-IBM-NUS @TREVID 2012: Surveillance Event Detection (SED)

Yang Cai*, Qiang Chen**, Lisa Brown**, Ankur Datta**, Quanfu Fan**, Rogerio Feris**, Shuicheng Yan***, Alex Hauptmann*, Sharath Pankanti**

*Carnegie Mellon University/**IBM Research/***National University of Singapore

Informedia E-Lamp @ TRECVID 2012

Multimedia Event Detection and Recounting MED and MER

Shoou-I Yu, Zhongwen Xu, Duo Ding, Waito Sze, Francisco Vicente, Zhenzhong Lan, Yang Cai, Shourabh Rawat, Peter Schulam, Nisarga Markandaiah, Sohail Bahmani, Antonio Juarez, Wei Tong, Yi Yang, Susanne Burger, Florian Metze, Rita Singh, Bhiksha Raj, Richard Stern, Teruko Mitamura, Eric Nyberg and Alexander Hauptmann
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh PA,15213

Abstract

We report on our system used in the TRECVID 2012 Multimedia Event Detection (MED) and Multimedia Event Recounting (MER) tasks. For MED, it consists of three main steps: extracting features, training detectors and fusion. In the feature extraction part, we extract many low-level, high-level, and text features. Those features are then represented in three different ways which are spatial bag-of words with standard tiling, spatial bag-of-words with feature and event specific tiling and the Gaussian Mixture Model Super Vector. In the detector training and fusion, two classifiers and three fusion methods are employed. The results from both the official sources and our internal evaluations show good performance of our system. Our MER system utilizes a subset of features and detection results from the MED system from which the recounting is generated.

1. MED System

1.1 Features

In order to encompass all aspects of a video, we extracted a wide variety of low-level and high-level features. Table 1 summarizes the features used in our system. Among those features, most of them are widely used features in the community, for example, SIFT, STIP and MFCC. We extracted those features using standard code available from the authors with default parameters.

Table 1: Features used for MED'12 system

	Visual Features	Audio Features
Low-level features	1. SIFT (Sande, Gevers, & Snoek, 2010) 2. Color SIFT (CSIFT) (Sande, Gevers, & Snoek, 2010) 3. Motion SIFT (MoSIFT) (Chen & Hauptmann, 2009) 4. Transformed Color Histogram (TCH) (Sande, Gevers, & Snoek, 2010) 5. STIP (Wang, Ullah, Klaser, Laptev, & Schmid, 2009) 6. Dense Trajectory (Wang, Klaser, Schmid, & Liu, 2011)	1. MFCC 2. Acoustic Unit Descriptors (AUDs) (Chaudhuri, Harvilla, & Raj, 2011)
High-level features	1. Semantic Indexing Concepts (SIN) (Over, et al., 2012) 2. Object Bank (Li, Su, Xing, & Fei-Fei, 2010)	1. Acoustic Scene Analysis
Text Features	1. Optical Character Recognition	1. Automatic Speech Recognition

Besides those common features, we have two home-grown features which are Motion SIFT

(MoSIFT) and Acoustic Unit Descriptors (AUDs). We will introduce these two features in the following subsections.

1.1.1 Motion SIFT (MoSIFT) Feature

The goal of developing the MoSIFT feature is to combine the features from the spatial domain and the temporal domain. Local spatio-temporal features around interest points provide compact and descriptive representations for video analysis and motion recognition. Current approaches tend to extend spatial descriptions by adding a temporal component to the appearance descriptor, which only implicitly captures motion information. MoSIFT detects interest points and encodes not only their local appearance but also explicitly models local motion. The idea is to detect distinctive local features through local appearance and motion. Figure 1 demonstrates the MoSIFT algorithm.

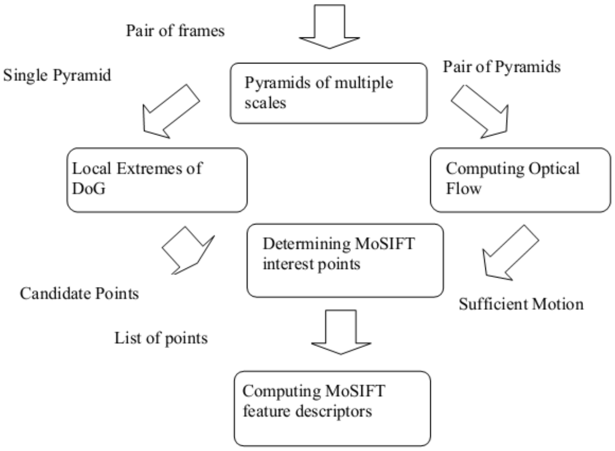


Figure 1: System flow chart of the MoSIFT algorithm.

The algorithm takes a pair of video frames to find spatio-temporal interest points at multiple scales. Two major computations are applied: SIFT point detection and optical flow computation according to the scale of the SIFT points.

For the descriptor, MoSIFT adapts the idea of grid aggregation in SIFT to describe motions. Optical flow detects the magnitude and direction of a movement. Thus, optical flow has the same properties as appearance gradients. The same aggregation can be applied to optical flow in the neighborhood of interest points to increase robustness to occlusion and deformation. The two aggregated histograms
(appearance and optical flow) are combined into the MoSIFT descriptor, which now has 256 dimensions.

1.1.2 Acoustic Unit Descriptors (AUDs)

We have developed an unsupervised lexicon learning algorithm that automatically learns units of sound. Each unit is such that it spans a set of audio frames, thereby taking local acoustic context into account. Using a maximum-likelihood estimation process, we can learn a set of such acoustic units unsupervised from audio data.
Each of these units can be thought of as low-level fundamental units of sound, and each audio frame is generated by these units. We refer to these units as Acoustic Unit Descriptors (AUDs) and we expect that the distribution of these units will carry information about the semantic content of the audio stream. Each AUD is represented by a 5-state Hidden Markov Model (HMM) with a 4-gaussian mixture output density function.

Ideally, with a perfect learning process, we would like to learn semantically interpretable lower-level units, such as a clap, a thud sound, a bang, etc. Naturally, it is hard to enforce semantic interpretability on the audio learning process at that level of detail. Further, because the space of all possible sounds is so large, many different sounds will be mapped into single sounds at learning time, since we can only learn a finite set of units.

1.2 Feature Representations

In the previous section, we briefly describe the features we used in the system. In this section, we will describe the representations we used for the raw features extracted in Section 1.

Three representations were used in our system. They were K-means based spatial bag-of-words model with standard tiling (Lazebnik, Schmid, & Ponce, 2006), K-means based spatial bag-of-words with feature and event specific tiling (Viitaniemil & Laaksonen, 2009) and Gaussian Mixture Model Super Vector (Campbell & Sturim, 2006). Since the K-means based spatial bag-of-words model with standard tiling and Gaussian Mixture Model Super Vector are standard technology, we will focus on the K-means based spatial bag-of-words model with feature and event specific tiling. For simplicity, we will refer to it as tiling.

Spatial bag-of-words model is a widely used representation of the low-level image/video features. The central idea of the spatial bag-of-words model is to divide the image into some small tiles and compute bag-of-words for each tile. Figure 2 shows a couple of tiling examples.

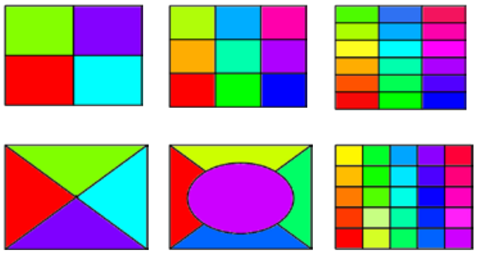


Figure 2: Examples of tiling

In general, the standard spatial bag-of-words tiling uses the 1x1, 2x2 and 4x4 tiling. However the use of those tilings is ad-hoc and some preliminary works have shown that other tilings might produce better performance (Viitaniemil & Laaksonen, 2009).

In our system, we systematically tested 80 different tilings to select the best one for each feature and each event. Table 2 shows the performance of feature specific tiling v.s. the standard tiling. The scores are computed from our internal experiments and are the average over 20 MED12 pre-specified events. The PMiss @ TER=12.5 metric is an official evaluation metric specified in the MED 2012 Evaluation Plan. A smaller PMiss score signifies better performance. From the table, we can see clearly that for all of the five features, the feature specific tiling performs consistently at least 1% better than the standard tiling.

Table 2: The performance of feature specific tiling and standard tiling

Feature	**SIFT**	**CSIFT**	**TCH**	**STIP**	**MOSIFT**
Feature Specific Tiling	0.4209	0.4496	0.4914	0.5178	0.4330
Standard Tiling	0.4325	0.4618	0.5052	0.5234	0.4456

Figure 3 shows an example of the performance of event specific tiling v.s. standard tiling on Event 25 (marriage proposal), which is a difficult event identified in our experiments. It can be seen clearly that the event specific tiling can noticeably improve the performance over standard tiling.

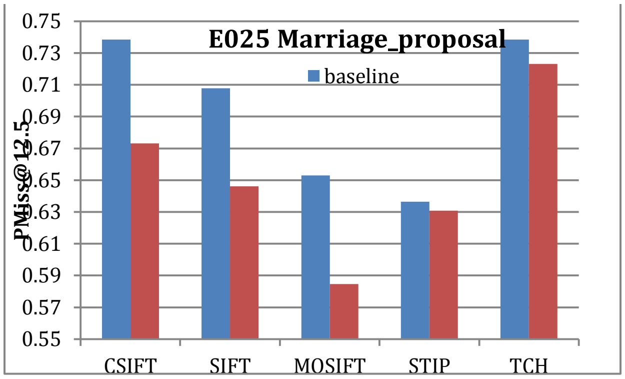


Figure 3: The comparison of event specific tiling and standard tiling on Event 25

1.3 Training and Fusion

We used the standard MED'12 training dataset for our internal evaluation and the training of the models for our submission. For our internal evaluation, the MED'12 training dataset was further divided into the training set and testing set by randomly selecting half of the positive examples into the training set and the other half into the testing set. The negative examples consisted of only NULL videos which do not have label information.

The two classifiers used in the system were kernel SVM and kernelized rigid regression. For simplicity, we will refer to it as kernel regression. For the K-means based feature representations, we used the Chi-squared kernel. For the GMM based representation RBF kernel was used. The parameters of the model were tuned by 5-fold cross validation and the PMiss @ TER = 12.5 metric was used as the evaluation metric.

For combining features from multiple modalities and the outputs of different classifiers, we used fusion and ensemble methods. More specifically, for the same classifier, we used three fusion methods to fuse different features. The fusion methods were early fusion, late fusion and double fusion (Lan, Bao, Yu, Liu, & Hauptmann, 2012). In early fusion, the kernel matrices from different features were normalized first and then combined together. In late fusion, the prediction scores from the models trained using different features were combined. In our system, we also used a fusion method called double fusion, which combines early fusion and late fusion together. Finally, the results from different classifiers were ensembled together. Figure 4 shows the diagram of our system.

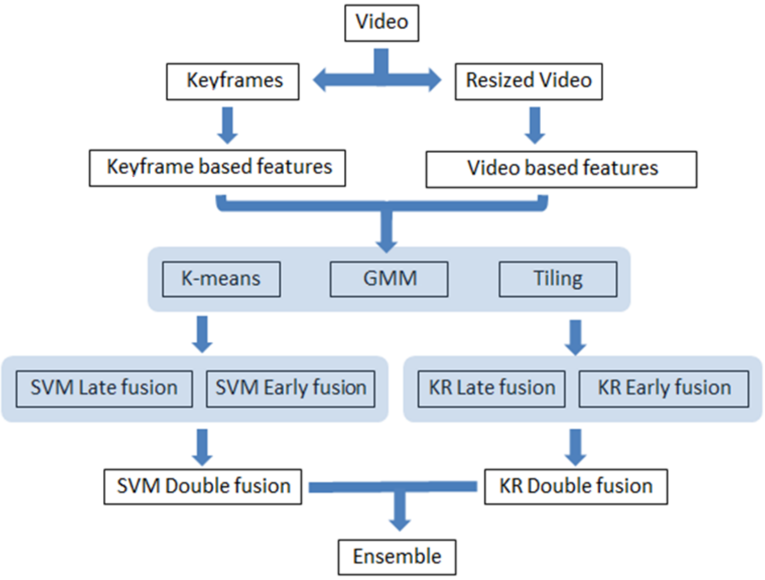


Figure 4: The diagram of the system

1.4 Submission

In the following section we describe in detail the runs we submitted to NIST. Table 3 shows the official performance of each submission.

1.4.1 Pre-Specified Submission

1.4.1.1 Submission 1: CMU_MED12_MED12TEST_PS_MEDFull_EKFull_AutoEAG_p_ensembleKRSVM_1

In this submission, using the features described in the previous section, we did the following to generate this run:

1. For each feature, train a SVM classifier and a kernel regression model.
2. Late fusion of all the results from SVM classifiers and kernel regression respectively.
3. Early fusion of all features except ASR.
4. Train a SVM classifier and a kernel regression model using 3 respectively.
5. Double fusion of SVM classifiers in 2 and 4.
6. Double fusion of kernel regression model in 2 and 4.
7. Ensemble of 5 and 6.

1.4.1.2 Submission 2: CMU_MED12_MED12TEST_PS_MEDFull_EK10Ex_AutoEAG_c_KRLF_1

1. For each feature, train a kernel regression model.
2. Late fusion of all the results from 1.

1.4.1.3 Submission 3: CMU_MED12_MED12TEST_PS_MEDFull_EKFull_AutoEAG_c_SVMLF_1

1. For each feature, train a SVM classifier.
2. Late fusion of all the results from 1.

4.1.4 Submission 4: CMU_MED12_MED12TEST_PS_MEDFull_EKFull_AutoEAG_c_BOB_1

1. Form all prediction results from step 1-7 in Submission 1 into a pool.
2. For each event, find the candidate in the pool which has the best performance.
3. Combine the candidates of each event together to form the submission.

1.4.2 Ad-Hoc Submission

1.4.2.1 Submission 5: CMU_MED12_MED12TEST_AH_MEDFull_EKFull_AutoEAG_p-SVM_1

The following features were used: SIFT, CSIFT, Transformed Color Histogram (TCH), Motion SIFT (MoSIFT), STIP, Dense Trajectory (DT), MFCC, SIN and Object Bank. Different from our pre-specified EKFull submission, we did not use GMM Super Vector and tiling representations. To get the detection results, the following steps were performed, which pretty much followed the pre-specified submission:

1. For each feature, train a SVM classifier.
2. Late fusion of the scores of each feature obtained from step 1.
3. Early fusion of the distance matrices of all the visual and acoustic features, and then use the obtained distance matrix to compute the kernel matrix.
4. Train a SVM classifier based on the kernel obtained by step 3.
5. Double fusion of the results from step 2 and step 4.

1.4.2.2 Submission 6: CMU_MED12_MED12TEST_AH_MEDFull_EK10Ex_AutoEAG_c-KR_1

Same features as Submission 5 were used in this submission. In our previous experiment, SVM tends to over fit the limited positive exemplars. Thus for EK10 we used kernel regression with Chi-squared kernel as the classifier. As we only have 10 positive exemplars for training, it is trickier to tune the regularization parameter of kernelized rigid regression by cross-validation. We have observed in our experiment that fixing the parameter to 1 usually yields good performance, though not necessarily the best. We therefore set regularization parameter as 1 for all the events. To get the detection results, the following four steps were performed, which pretty much followed the pre-specified submission:

1. For each feature, train a kernel regression model.
2. Late fusion of the prediction scores of each feature obtained from step 1.
3. Early fusion of the distance matrices of all the visual and acoustic features, and then use the obtained distance matrix to compute the kernel.
4. Train a rigid regression classifier based on the kernel obtained by step 3.
5. Double fusion of the scores obtained from step 2 and step 4.

Table 3: The official performance of the 6 submissions

Task	Train Type	EAG	SYSID	NDC	PFa	PMiss
Pre-Specified	EKFull	AutoEAG	p-ensembleKRSVM_1	0.637	0.0341	0.2113
	EKFull	AutoEAG	c-SVMLF_1	0.6584	0.0341	0.2325
	EKFull	AutoEAG	c-BOB_1	0.6427	0.0341	0.2168
	EK10Ex	AutoEAG	c-KRLF_1	0.8588	0.0345	0.4286
Ad-Hoc	EKFull	AutoEAG	p-SVM_1	0.6494	0.0354	0.208
	EK10Ex	AutoEAG	c-KR_1	1.1078	0.0568	0.3982

2. MER System

2.1 Features

We included the following aspects in our MER submission:

- Relationships
    - Visual features that are relevant to the event
    - Audio features that are relevant to the event
    - Co-occurrence of the visual concepts (SIN'11)
- Observations
    - Event-Relevant Visual Concepts
    - Video-Distinctive Visual Concepts
    - ASR Transcripts
    - Event-Specific Object Bank Results
    - Audio Concepts (Noisemes)

2.2 Visual and Audio Concepts

We use the histogram of each video semantic class aggregated over the whole video clip. To use the visual concepts, we first generated a bipartite graph matching of Object Bank classes and SIN'11 concepts for the MED12 dataset. The process flow is shown in the Figure 5.

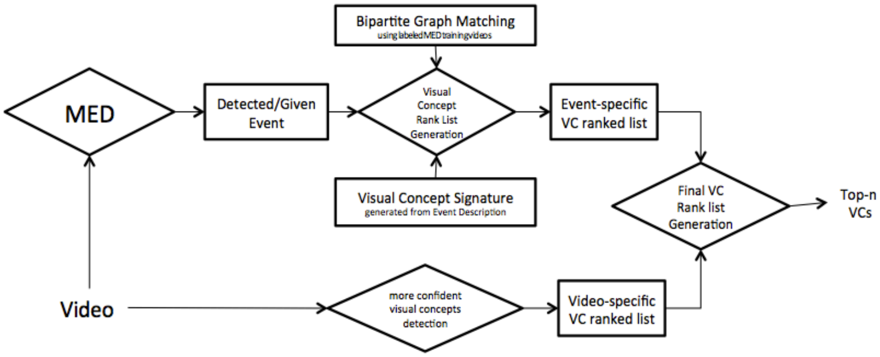

Bipartite Graph Matching
using labeled MED training videos
MED
Detected/Given Event
Visual Concept Rank List Generation
Event-specific VC ranked list
Visual Concept Signature
generated from Event Description
Final VC Rank list Generation
Top-n VCs
Video
more confident visual concepts detection
Video-specific VC ranked list

Figure 5: Flow chart of visual and audio concepts processing

The Noiseme semantic audio concepts similarly indicate “non_linguistic_audio” information in the video. (e.g. “speech”, “music”, “noise” etc.). We use the histogram of each audio concept in the video to mention that in this video we can mainly hear the sound of music, singing or noise. We again use Bipartite Graph Matching to map the Noisemes to the events. All the audio concepts are ranked based on their percentage in the video.

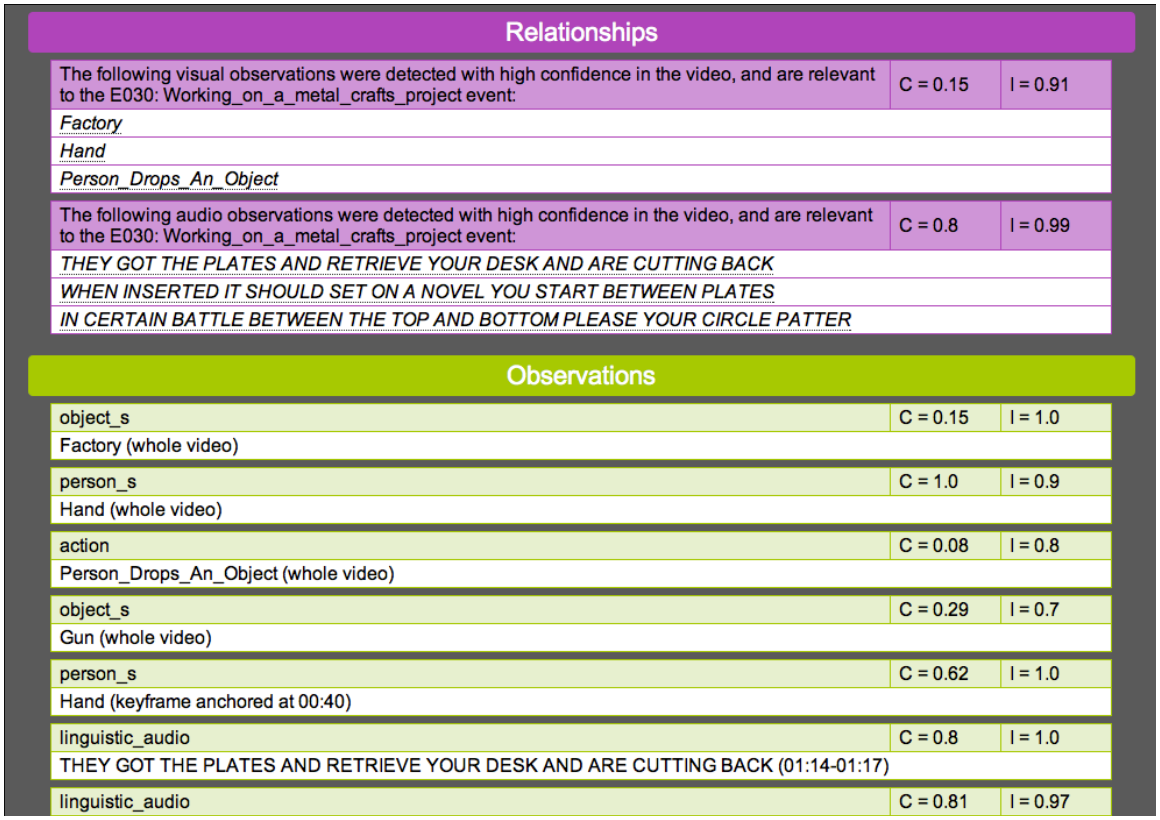
2.3 ASR Transcripts

Automatic speech recognition transcripts that indicate “linguistic_audio” information in the video. (e.g. “okay”, “hello”, “she didn’t” etc.). We use TF-IDF according to the word-level ASR confidence to calculate the relevant of each ASR word result to the event kit. We then rank the ASR Transcripts according to their relevance to the event.

2.4 An Example of Our Recounting Submission

The requirements of the submission were that all Multimedia Event Recounting (MER) participants are required to produce a recounting for 30 selected video clips where it is known that the clip contains a specific MER event. There will be five events chosen from the MED pre-specified events list, and six video clips per event. The system's recounting summarizations were to be evaluated by a panel of judges.

An example of our recounting submission is shown in Figure 6.



Relationships

The following visual observations were detected with high confidence in the video, and are relevant to the E030: Working_on_a_metal_crafts_project event:	C = 0.15	I = 0.91
Factory		
Hand		
Person_Drops_An_Object		

The following audio observations were detected with high confidence in the video, and are relevant to the E030: Working_on_a_metal_crafts_project event:	C = 0.8	I = 0.99
THEY GOT THE PLATES AND RETRIEVE YOUR DESK AND ARE CUTTING BACK		
WHEN INSERTED IT SHOULD SET ON A NOVEL YOU START BETWEEN PLATES		
IN CERTAIN BATTLE BETWEEN THE TOP AND BOTTOM PLEASE YOUR CIRCLE PATTER		

Observations

object_s	C = 0.15	I = 1.0
Factory (whole video)		

person_s	C = 1.0	I = 0.9
Hand (whole video)		

action	C = 0.08	I = 0.8
Person_Drops_An_Object (whole video)		

object_s	C = 0.29	I = 0.7
Gun (whole video)		

person_s	C = 0.62	I = 1.0
Hand (keyframe anchored at 00:40)		

linguistic_audio	C = 0.8	I = 1.0
THEY GOT THE PLATES AND RETRIEVE YOUR DESK AND ARE CUTTING BACK (01:14-01:17)		

linguistic_audio	C = 0.81	I = 0.97

Figure 6: An example of our MER submission

2.5 Performance

Table 4 shows the performance of our MER system compared to the average performance of other submitted systems. We achieve significantly better performance in the two MER tasks, which shows the effectiveness of our MER system.

Table 4: Official performance of MER

	MER-to-Event	MER-to-Clip	Combined (0.4*E+0.6*C)
Average of submitted systems	68.75%	43.72%	0.54
CMU_ELamp-MER-System	85.56%	66.30%	**0.74**

3. Acknowledgments

This work has been supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20068. The U.S. government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

References

Campbell, W., & Sturim, D. (2006). Support vector machines using GMM supervectors for speaker verification. *IEEE Signal Processing Letters*.

Chaudhuri, S., Harvilla, M., & Raj, B. (2011). Unsupervised Learning of Acoustic Unit Descriptors for Audio Content Representation and Classification. *Interspeech*.

Chen, M., & Hauptmann, A. (2009). *MoSIFT: Reocgnizing Human Actions in Surveillance Videos.* Carnegie Mellon University. Carnegie Mellon University.

Lan, Z., Bao, L., Yu, S.-I., Liu, W., & Hauptmann, A. G. (2012). Double Fusion for Multimedia Event Detection. *MMM.*

Lazebnik, S., Schmid, C., & Ponce, J. (2006). Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. *CVPR.*

Li, L.-J., Su, H., Xing, E., & Fei-Fei, L. (2010). Object Bank: A High-Level Image Representation for Scene Classification and Semantic Feature Sparsification. *NIPS.*

Over, P., Awad, G., Michel, M., Fiscus, J., Sanders, G., Shaw, B., et al. (2012). TRECVID 2012 -- An Overview of the Goals, Tasks, Data, Evaluation Mechanisms and Metrics. *Proceedings of TRECVID 2012.*

Sande, K. E., Gevers, T., & Snoek, C. G. (2010). Evaluating color descriptors for object and scene recognition. *TPAMI*.

Viitaniemil, V., & Laaksonen, J. (2009). Spatial Extensions to Bag of Visual Words. *CIVR.*

Wang, H., Klaser, A., Schmid, C., & Liu, C.-L. (2011). Action recognition by dense trajectories. *CVPR.*

Wang, H., Ullah, M. M., Klaser, A., Laptev, I., & Schmid, C. (2009). Evaluation of local spatio-temporal features for action recognition. *BMVC.*

Informedia Aurora @TREVID 2012
Semantic Indexing (SIN)

Lu Jiang, Alex Hauptmann
Carnegie Mellon University

1 Features

For this year's SIN submission we used three features: SIFT, Color SIFT (CSIFT) and Motion SIFT (MoSIFT). SIFT and CSIFT (with Harris-Laplace detectors) describe the gradient and color information of images. MoSIFT describes both the optical flow and gradient information of video clips. Compared with 2011's submission, we only use 3 features instead of 5 features.

2 Label Set

In this year's submission we used the SIN 2011's label set instead of SIN 2012's label set, as we incorrectly used the label set proposed on the task's webpage.

3 Classifiers

The cascade SVM classifiers are adopted as our classifier, which is essentially the same algorithm as [1]. However, this year we implemented the cascade SVM algorithm on Hadoop to accelerate the SIN extraction. Now the kernel computation component and the SVM testing component are released at [2]. Using the Hadoop cascade SVM we can finish SIN training and testing in 36 hours on PDL Open Cloud platform which consists of 50 compute nodes with 2x quad-core Intel E5440 (2.83GHz, 12MB L2 cache, 1333 MHz FSB).

4 Submitted Runs

We submitted 4 runs for the individual concept runs.

- Run1: Safe run as last year using 3 features (SIFT-CSIFT-MOSIFT) cascade SVM model.
- Run2: Late fuse Run 1 with random forest models trained on SIFT features. The experiments show that random forest trained on the SIFT feature yields the best improvement over Run1. Generally random forest models are worse than SVM models, so during the fusion, SVM's performance are fused according to the following formula:

  Fusion_score = 0.8 * SVM_score + 0.2 *Random_forest_score
- Run3: Label propagation on Run2. According to the relation between concepts, the prediction of individual concept could be reinforced by its related concepts. Our goal is to boost the accuracy of the concept detector by its related concept detectors.

  We regard the concept detector score as a message which is propagated through the graph. It consists of two steps forward and backward passes and after the propagation each node receives the message from all the other nodes in its connected component.

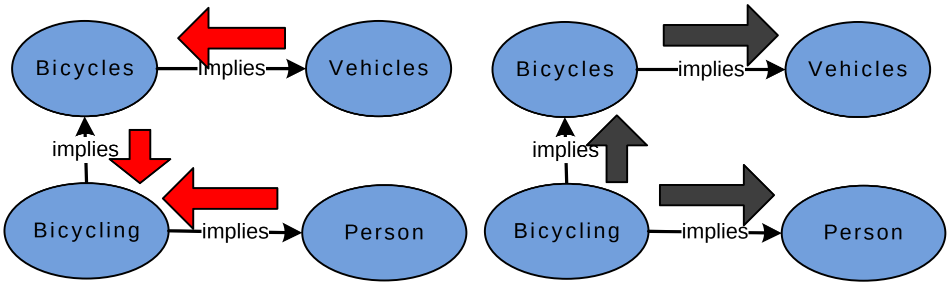


Fig.1. an illustrative example of label propagation on the sun-graph on bicycles.

According to our experiments on development set, this improvement is considerable on SIN 2011's evaluation dataset, where our current best run without propagation is 0.1508 and after the propagation the number is improved to 0.1575. The winner's best score is 0.173 (since they use additional features).

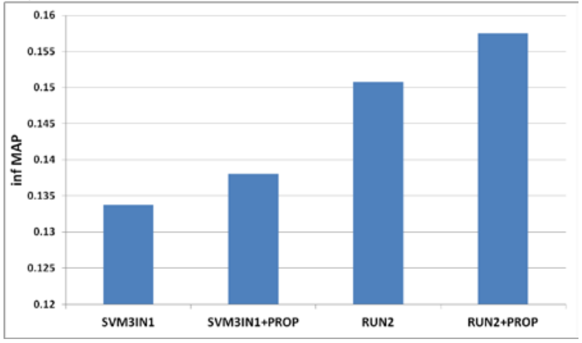


Fig. 2 the performance of label propagation on SIN 2011's development set.

- Run4: Brave ideas on Run3. For some inaccurate concepts, we did some aggressive propagation (same algorithm as RUN3 but with aggressive parameters) to improve them. In addition, we filtered out the top 2000 blank, black and junk frames in the rest concepts.

We submitted 2 runs for the pair run which is to detect pairs of unrelated concepts instead of detects simple concepts. Our general idea is as follows: training individual concept detectors and then enhancing the prediction of pair concept using the related concept detectors. For example for the pair concepts "[901] Beach + Mountain" , we used the concepts like "Beach", "Mountain", "Valleys", "Rocky_Ground", "Outdoor", "Lakes", "Islands". The difference between the two runs lie in the different weights in combing the final score.

- Run5 employs the average score for each related concepts.

- Run6 applies the score based on the concepts' prediction accuracy in the development set.

Experimental Results

In this section we summarize our results. Tab.1 shows our results in the full run. Our observation is that

- Random Forest + SVM may not improve (probably hurt) the performance.
- Aggressive label propagation helps a little bit on SIN 2012 dataset

We extrapolate that for the label propagation, the reason why no significant improvement is that our individual concept detector is not as good as others (since we used the last year's training dataset). In addition, more than half of the concepts are isolated in the concept relation graph and therefore the label propagation doesn't change their prediction values

Tab. 1. The final results of our individual concept detection run

RUN NAME	INF AP
F_A_CMU4_4	0.204174
F_A_CMU3_1	0.202609
F_A_CMU1_3	0.202087
F_A_CMU2_2	0.201457

Tab.2 shows our results for the pair run. We think enhance the pair detection with the related concepts seems correct approach and weighting the parameters according to concept's accuracy seems to be better.

Tab. 2. The final results of our pair concept detection run

RUN NAME	INF AP
P_A_CMU6_1	**0.0482**
P_A_CMU5_2	0.0393

Future work

In our opinion, a promising direction is to learn the concept correlation from the data on which the labels are propagated. In addition, we plan to extend the propagation idea to another perspective i.e. to enhance a frame concept prediction by those in the same video clip.

Acknowledgments

This portion of the work has been supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20066. The U.S. government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

References

[1] Lei Bao, Longfei Zhang, Shoou-I Yu, Zhen-zhong Lan, Lu Jiang, Arnold Overwijk, Qin Jin, Shohei Takahashi, Brian Langner, Yuanpeng Li, Michael Garbus, Susanne Burger, Florian Metze, and Alexander Hauptmann, Informedia @ TRECVID 2011; Trecvid Video Retrieval Evaluation Workshop, NIST, Gaitherburg, Md, December 2011
[2] Cascade SVM on Hadoop: https://code.google.com/p/cascadesvm/
[3] http://www-nlpir.nist.gov/projects/tv2012/tv11.sin.relations.txt

CMU-IBM-NUS@TRECVID 2012: Surveillance Event Detection[*]

Yang Cai[†] Qiang Chen[‡+] Lisa Brown[‡] Ankur Datta[‡] Quanfu Fan[‡] Rogerio Feris[‡]
Shuicheng Yan[+] Alex Hauptmann[†] Sharath Pankanti[‡]
Carnegie Mellon University[†] IBM Research[‡] National University of Singpore[+]

1 Introduction

We present a generic event detection system evaluated in the SED task of TRECVID 2012. It consists of two parts: the retrospective system and the interactive system. The retrospective system uses MoSIFT [2] as low level feature, Fisher Vector encoding [1] to represent samples generated by sliding window approach and linear SVM for event classification. For interactive system, we introduce event-specific visualization schemes for efficient interaction and temporal locality based search method for user feedback utilization. Among the primary runs of all teams, our retrospective system ranked 1st for 4 / 7 events, in terms of actual DCR.

2 Fisher Vector Encoding for Retrospective Event Detection

2.1 Framework

We use MoSIFT as our low level feature and Fisher Vector encoding (FV) to represent detection windows upon MoSIFT. Then, linear SVM is used to train classification model based on annotated positives and randomly sampled negatives. For testing, multiscale detection is applied and non-maximum suppression is used to exclude duplicate detections on single event.

2.2 Fisher Vector Encoding

Fisher Vector encoding utilizes a Gaussian mixture model (GMM) $U_\lambda(x) = \sum_{k=1}^{K} \pi_k u_k(x)$ trained on local features of a large image set using Maximum Likelihood (ML) estimation. The parameters of the trained GMM are denoted as $\lambda = \{\pi_k, \mu_k, \Sigma_k, k = 1, \cdots, K\}$, where $\{\pi, \mu, \Sigma\}$ are the prior probability, mean vector and diagonal covariance matrix of Gaussian mixture respectively. This GMM is used for description of low level feature.

Then for a set of low level features $X = \{x_1, \cdots, x_N\}$ extracted from a clip of videos y, the soft assignments of the descriptor x_i to the kth Gaussian components γ_{ik} is computed by: $\gamma_{ik} = \frac{\pi_k u_k(x_i)}{\sum_{k=1}^{K} \pi_k u_k(x_i)}$. And the FV for X is denoted as $\phi(X) = \{u_1, v_1, \cdots, u_K, v_K\}$ while u_k and v_k is defined as $u_k = \sum_{i=1}^{N} \frac{1}{N\sqrt{\pi_k}} \gamma_{ik} \frac{x_i - \mu_k}{\sigma_k}$ and $v_k = \sum_{i=1}^{N} \frac{1}{N\sqrt{2\pi_k}} \gamma_{ik} [\frac{(x_i - \mu_k)^2}{\sigma_k^2} - 1]$ while σ_k are square root of the diagonal values of Σ_k. The FV has several good properties: (a)Fisher Vector encoding is not limited to computing visual word occurrence. It also encodes additional the distribution information of the feature points, which will perform more stable when encoding a single feature point. (b) Fisher Vector encoding is not limited to computing visual word occurrence. It also encodes additional the distribution information of the feature points, which will perform more stable when encoding a single feature point. (c) it can naturally separate the video specific information from the noisy local features(b) we can use linear model for this representation. We build efficient implementation for FV which can reach the speed of 10 times faster than real time.

Power Normalization and L2 Normalization: It is easy to observe that as the number of Gaussians increases, Fisher vectors become sparser so that the distribution of features in a given dimension becomes more peaky around zero. As introduced in [1], we also use a combination of power normalization and l2 normalization for each fisher vector encoding features. Suppose z is one dimension of the ϕ, the power normalization is defined as $f(z) = sign(z)|z|^\alpha$ where $0 \leq \alpha \leq 1$ is a parameter of the normalization and we choose $\alpha = 0.5$ in all the experiments and then followed by $l2$ normalization.

[*] Equal contributions by Yang Cai and Qiang Chen

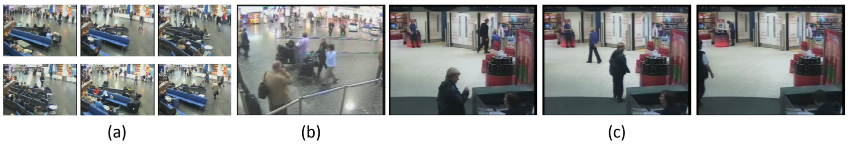


Figure 1: Illustration of visualization schemes. (a) is using ”Many low-resolution units” for ”PersonRuns”, (b) is using ”Few high-resolution units” for ”CellToEar” and (c) is using ”Contextual units” for ”PeopleSplitUp”.

2.3 Efficient Implementation

Compared with standard BoW, the computation cost of FV is much fewer. For BoW, the computation mainly comes from the Vector Quantization(VQ) step which has the complexity of $\mathcal{O}(NDM)$ where N is the number of local features, D is the dimensionality of local feature and M is the codebook size. For FV, the cost has two part that one part is the GMM assignment calculation γ_{ik} which has complexity of $\mathcal{O}(NDK)$ where K is the GMM model size, another part is the FV calculation which often takes much less time than the first part(usually $\leq 1\%$). Then we can see that since we usually use much less number of Gaussians for FV (usually 128 or 256) than the number of visual words for BoW (usually a few thousands) the computation for FV is very highly efficient compared to standard BoW. The experiment shows that our implementation can produce the FV 10 times faster that real time excluding the local feature extraction part cost.

2.4 Multiscale detection and Non-maximum suppression

Ideally, we need to search over different scales and different step size to locate the exact event in the video sequences. However, it is unpractical for current sliding window framework. For example, the maximum length of PersonRuns event in the Dev dataset is 1000 frames while the minimal length is 10 frames – such diversity of event duration brings a lot of search space and the computation cost is too high. Instead of this exhaustive search, we select three scales which are closest to the average duration of each event and accept the scale with best performance.

NMS is widely used in many computer vision tasks, e.g. edge detection or object detection. In SED task, NMS will set all scores in the current neighborhood window that are lower than the maximum value in that window to zero (or lowest value).The current score of the sliding window is then compared to this maximum value. If lower it is set to zero otherwise the value is unchanged. We use NMS to suppress the multiple detection for single event.

3 Interactive Event Detection System

We attempted to address two central problems of an interactive surveillance event detection system: (1) detection results visualization and (2) user feedback utilization. Because of the limited time available for interaction, the system design was driven by efficiency considerations from both these two perspectives. Specifically, in this year system, we proposed two techniques for the two aspects respectively, which are introduced as follow.

3.1 Event-specific Detection Results Visualization

In a surveillance video where tens or even hundreds of people appear simultaneously in one camera, it’s not surprising to take one several minutes to verify a correct event detection. To help user more efficiently capture video content, we experimented with several presentation schemes and designed an event-specific visualization approach by finding good presentation schemes for different events.

We define a single detection result as a visualization unit or unit. In our interactive system, a unit is presented by repeatedly playing the detected video segment at twice original speed. Given the limited space of a screen and the limited perception ability of an user, the problem then turns to how to arrange these units to better trade-off the visualization quantity (e.g. the number of units in one screen) and visualization quality (e.g. the clearness of each unit). We specifically explored following three presentation schemes.

Many low-resolution units: As shown in Figure 1(a), we presented multiple low-resolution units in a screen. It leveraged the fact that users can simultaneously capture the rough content of multiple units. Due to the roughness of such simultaneous capture, it’s only favored by events which can be captured by a glance, such as ”PersonRuns”. For other sophisticated events, however, it doesn’t benefit the performance due to low-resolution units.

Few high-resolution units: Due to the impreciseness of previous scheme, we presented the units with higher resolution at the expense of fewer units in a screen (see Figure 1(b)). This presentation

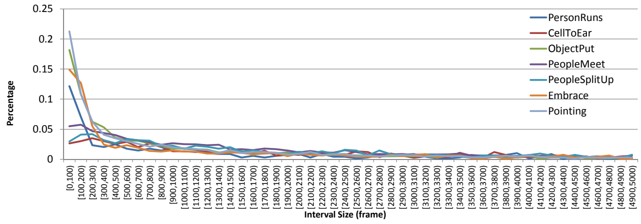


Figure 2: Distribution of frame intervals between each consecutive events pair in SED development set.

scheme is helpful for events whose action is small, weak and always lying in a tiny sub-region of the whole frame, such as ”CellToEar”, ”ObjectPut” and etc.

Contextual units: Instead of only presenting the unit corresponding to a detection result, this scheme also presented the contextual units, which are neighbor windows next to the detection. It helped the verification of slightly drifted true positives. The middle unit of Figure 1(c) shows an slightly drifted detection of ”PeopleSplitUp”, which started with a person walking away from an airport agent. Since it missed the moment they were together, it’s very hard for user to judge if the detection is a true positive or a false alarm. However, by providing at the context (the first and third units in Figure 1)(b), the problem can be easily solved.

Even different events favor different presentation schemes, in practice, we didn’t use only one for the interaction of curtain event. Because the good presentation scheme for a event is just in general sense and unnecessarily true for all specific cases (e.g. children’s running may also need detailed looking). In the interactive system for this year submission, we organized these schemes into one integrated interface.

3.2 Temporal Locality Based Search

By analyzing the distribution of events in temporal domain, we observed an interesting ”clustered” distribution pattern for some events. To see this, we calculated the frame intervals between each consecutive events pair in a video and then counted the numbers of pairs dropping into quantized interval bins. In Figure 2 which visualizes the interval distribution, we can easily see that, for some events (e.g. Pointing, ObjectPut and etc.), most of the intervals are very small, which indicates an clustered distribution of them. In other words, if we see an event at somewhere, we are likely to see another one near to it. Based on such temporal locality, we proposed a interactive searching method focusing on saving miss detections.

Let d_t be a system detection whose middle frame is t. Let $\mathbf{D}$ be a set of system detections. Let δ_t be a predefined short interval. When user labeled one system detection d_t as true positive, the temporal locality search method retrieves a set of neighbor detections $\mathbf{D} = \{\mathbf{d}_{\mathbf{t}'} ||\mathbf{t}^{'} - \mathbf{t}| < \delta_{\mathbf{t}}\}$ to users. Then user can quickly go through the list and search for miss detections.

4 Experiments

4.1 Evaluation of Retrospective Event Detection

Experimental Setting: For an ideal event detection framework, we focus on the efficiency and effectiveness of training and testing stages. For training stages, we use Fisher Vector encoding as the representation of video events which allows us to use linear classifier to obtain efficiency and good performance. We first trained a GMM model with 256 codebooks. Each MoSIFT feature is first reduced to 80 dims using PCA. No SPM is utilized in this year. The final dimension is $2 \times 80 \times 256 = 40960$. We perform hard samples mining on the training set so that the learned classifier is more generalized. 2-fold cross validation is used to obtain the thresholding of final output. At testing stages, ideally, exhaustive search over temporal space should be utilized. However, two factors avoid this: (1) high cost for dense search. (2) unbalanced output at different scales. Thus, we calculated the mean temporal duration of each events and select 30, 60, 120 as the testing frame windows and select best performance window size as the final result. Since DCR evaluation is highly nonlinear, we also perform threshold prediction in which we use topK threshold and min DCR threshold on the training set as observation to predict the final best thresholds for DCR.

Results: We show our primary run result using Fisher Vector encoding (*CMU12_FV*) on retrospective task in Table 1 compared with the results of CMU Bag-of-Words of last year (*CMU11_BoW*) and the other teams’ best primary run results this year (*Others12_Best*). Please note the test video of 2012 is a subset of last year’s. It is shown that our *CMU12_FV* is better than *CMU11_BoW*. We had similar observation in our experiments on development set. In terms of the actual DCR, our

Table 1: The actual DCR and minimum DCR comparisons of primary runs among *CMU12_FV*, *Others12_Best* and *CMU11_BoW*.

		CMU12_FV		*Others12_Best*		*CMU11_BoW*	
	Rank	ActDCR	MinDCR	ActDCR	MinDCR	ActDCR	MinDCR
CellToEar	1	**1.0007**	1.0003	1.0040	0.9814	1.0365	1.0003
Embrace	1	**0.8000**	0.7794	0.8247	0.8240	0.8840	0.8658
ObjectPut	2	1.0040	0.9994	**0.9983**	0.9983	1.0171	1.0003
PeopleMeet	3	1.0361	0.9490	**0.9799**	0.9777	1.0100	0.9724
PeopleSplitUp	1	**0.8433**	0.7882	0.9843	0.9787	1.0217	1.0003
PersonRuns	1	**0.8346**	0.7872	0.9702	0.9623	0.8924	0.8370
Pointing	3	1.0175	0.9921	**0.9813**	0.9770	1.5186	1.0001

Table 2: The actual DCR comparision between different interaction stategies on development set and evaluation set.

	Development Set				Evaluation Set	
	Retro	*Naive*	*ESpecVis*	*ESpecVis+TLRerank*	*Retro*	*ESpecVis+TLRerank*
CellToEar	1.0008	1.0014	1.0008	1.0009	1.0007	1.009
Embrace	0.9519	0.9547	0.9344	0.9115	0.8	0.6696
ObjectPut	1.0033	1.0026	1.0024	1.0023	1.004	1.0064
PeopleMeet	0.9381	0.9338	0.9334	0.9361	1.0361	0.9786
PeopleSplitUp	0.8972	0.9416	0.889	0.8863	0.8433	0.8177
PersonRuns	0.761	0.7528	0.7511	0.7366	0.8346	0.6445
Pointing	1.0168	1.0109	1.0134	1.0084	1.0175	0.9854

system achieved best performance in four events this year. It shows good results on ”PersonRuns”, ”PeopleSplitUp”, ”Embrace” and ”PeopleMeet”, while in other tasks the results are still close to random. Other localized method should be used to tackle these failure tasks.

4.2 Evaluation of Interactive Event Detection

Experimental Setting: Besides reporting the formal evaluation results provided by NIST, we also included the developing experimental results, to exam the effectiveness of proposed interaction methods. Specifically, in our developing experiments, we used ”Dev08” as training and ”Eval08” as testing. Instead of using the 25 minutes interaction walltime of formal evaluation setting, the developing experiments used an interaction walltime of 5 minutes for each event. In table 2, we compared the actual DCR on development set and evaluation set (primary runs) for 4 interaction strategies: (1)no interaction (*Retro*), (2)scanning system detections only with ”many low-resolution units” visualization discussed in Section 3.1 (*Naive*), (3)scanning system detections using event-specific visualization (*ESpecVis*) and (4)scanning system detections using both event-specific visualization and temporal locality search (*ESpecVis+TLSearch*).

Results: In developing experiments, compared to *Retro*, *Naive* only shown significant improvements on event ”PersonRuns” which is very easy to identify. On other events, the performance even dropped dramatically (e.g. ”PeopleSplitUp”) after this naive interaction. By adopting the better event-specific visualizations, *ESpecVis* shown improvements over *Retro* on more events than *Naive*. Specifically, for events ”Embrace” and ”PeopleSplitUp” which *Naive* didn’t do well, *ESpecVis* demonstrated performance gain by providing high resolution visualization and event context. By further adding temporal locality search, we observed larger improvements on ”PersonRuns” and ”Embrace” for *ESpecVis+TLSearch* compared with *ESpecVis*. Since the these events shown relative high temporal locality as demonstrated in Figure 2, the temporal locality search has high probability to save miss detections. However, we also found the current interaction techniques were not effective on some events, such as ”CellToEar” and ”ObjectPut”. There are two-fold reasons. First of all, the current visualization method still has difficulty in presenting these events with tiny and weak actions, especially in complected scenes. Secondly, one necessary condition for temporal locality search to be effective is user can find some true positives during interaction. Since the retrospective system still cannot get reasonable detections on these events, the proposed temporal locality cannot benefit the performance much.

As for formal evaluation, it basically shared the same performance changing trends of the one on development set. Due to the the longer interaction time (25 minutes) used in formal evaluation, we observed greater improvement in terms of absolute values.

References

[1] J. S. Florent Perronnin and T. Mensink. Improving the fisher kernel for large-scale image classification. In *ECCV*, 2010.

[2] M. yu Chen and A. Hauptmann. Mosift: Reocgnizing human actions in surveillance videos. In *CMU-CS-09-161*, 2009.