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 Singapore[+]

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) It can naturally separate the video specific information from the noisy local features. (c) 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

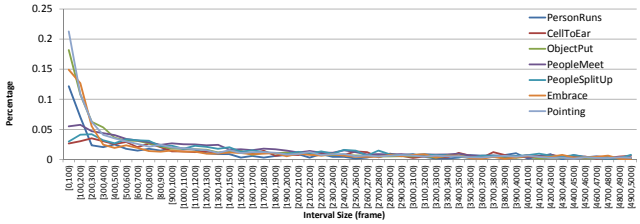

Figure 2: Distribution of frame intervals between each consecutive events pair in SED development set.

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 $\mathbf{D} = \{\mathbf{d_{t'}} || \mathbf{t}^{'} - \mathbf{t}| < \delta_{\mathbf{t}}\}$ 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.