BUPT-MCPRL at TRECVID 2016*

Zhicheng Zhao, Menglai Wang, Rui Xiang, Shanwei Zhao, Kaihui Zhou,
Mei liu, Shizhe He, Yandong Zhu, Yanyun Zhao, Fei Su

Multimedia Communication and Pattern Recognition Labs,
Beijing University of Posts and Telecommunications, Beijing 100876, China
{zhaozc, zyy, sufei}@bupt.edu.cn

Abstract

In this paper, we describe BUPT-MCPRL systems and evaluation results at TRECVID 2016[12]. Our team participated in three tasks: surveillance event detection, instance search and multimedia event detection.

Surveillance Event Detection(SED): We improve our system significantly with pedestrian detection based on a modified version of Faster R-CNN.

- Embrace, ObjectPut, PersonRuns and Pointing: Faster R-CNN (key-posture detection) + rules
- PeopleMeet and PeopleSplitUp: Faster R-CNN (pedestrian detection) + Multi-Object Tracking + Spring Model + CNN (interaction detection)

Instance Search (INS): We submit three runs for automatic INS and one run for interactive search.

- **F_A_BUPT_MCPRL_1**: retrieve persons firstly, and then detect locations in candidate frames.
- **F_A_BUPT_MCPRL_2**: retrieve locations firstly, and then detect persons in candidate frames.
- **F_A_BUPT_MCPRL_3**: merge two rank lists from F_A_BUPT_MCPRL_1 and F_A_BUPT_MCPRL_2.
- **I_A_BUPT_MCPRL_4**: retrieve persons firstly, afterwards, optimize candidate frames interactively. Finally, retrieve the locations.

Multimedia Event Detection(MED): We submit two runs of MED task.

- **p-baseline:** use GoogleNet to extract the frame-level features and encode them according to VLAD. A binary Support Vector Machine is used for scoring an event.
- **c-contrast**: combine ResNet152 with GoogleNet to perform feature fusion.

1 Surveillance Event Detection

We focus on the events of Embrace, PeopleMeet, PeopleSplitUp, ObjectPut, PersonRuns and Pointing this year. We propose different approaches to detect above events. Same as last year, the overall system consists of two parts: the retrospective part and the interactive part. Pedestrian detection, pedestrian tracking and event detection are implemented in retrospective part while in interactive part, after fixing the false and missing rate, we develop an interactive system to determine the events. We achieve good results in all the six events thanks to the improvements we've made in our detecting algorithms.

1.1 Pedestrian Detection

Accurate pedestrian detection in highly crowded surveillance videos is a challenging task, since the regions of

*This work is supported by Chinese National Natural Science Foundation (61471049, 61372169, 61532018), and Network System and Network Culture Foundation of Beijing.

pedestrians in the videos may be largely occluded by other pedestrians. Like last year, we transform pedestrian detection into head-shoulder detection[2]. The part-based scheme effectively restrains the appearance variations of pedestrians caused by heavy occlusion.

This year, we adopt Faster R-CNN[1] to perform pedestrian detection. During our efforts of leveraging Faster R-CNN as a pedestrian detector, we observe two drawbacks which may decrease the performance as follows.

First, the convolutional features from Conv5_3 are of low resolution for detecting small pedestrians. A typical pedestrian instance in SED datasets has a small size (e.g., 32x32 pixels). When extracting features from a ROI (Region-of-Interest) in Conv5_3 (stride = 16 pixels), we find it is impossible to pool them into a fixed resolution of 7x7 pixels. These features are not discriminative, thus degrade the performance. Following [2], we address this problem by using skip pooling at multiple scales. As shown in Figure 1, for each proposal, we extract a fixed-size descriptor from several layers using ROI pooling. Each descriptor is L2-normalized, concatenated, scaled, and dimension-reduced (1x1 convolution) to produce a feature descriptor with a fixed-length 512x7x7. The descriptor is fed into the fully-connected (fc) layers to produce the classification score and bounding box coordinates. 1.3% relative improvement in MR (Miss Rate) is obtained.

Second, there are many false predictions caused by a lack of hard example mining. In original Faster R-CNN, heuristic strategy is utilized for hard example mining, while it is infeasible for pedestrian detection. We adopt a rough but effective strategy. In our implementation, a three-class classifier which includes three steps, is learned to deal with this problem. Firstly, we train a pedestrian/background classifier and detect pedestrians on the training set. Secondly, we collect hard negative instances from the outputs in the last step: their Intersection-over-Union (IOU) values which overlap with ground-truth bounding boxes are less than 0.5 and scores are greater than 0.8. All the hard negative instances are regarded as a new class, i.e. the hard negative class. Finally, we retrain a three-class classifier via fine-tuning the last classifier. In this way, we force the networks to learn the differences between pedestrian and hard negatives. We obtain a surprising 14% improvement in MR.

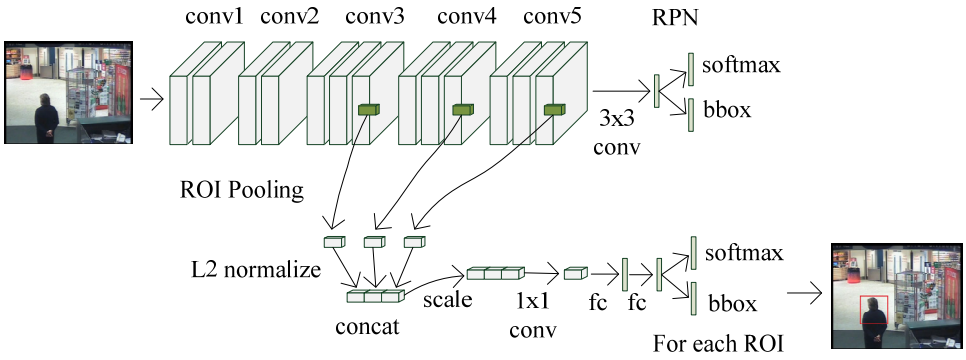

Figure 1 The architecture of the modified Faster R-CNN.

1.2 Pedestrian Tracking

We extend our contribution of multi-target tracking algorithm in previous work[3]. It is modeled as a hierarchical bipartite problem[4] in a tracking by detection paradigm. This year, improvements are as follow:

[2] In this paper, we refer pedestrian detection to head-shoulder detection.

1. Motion and location information are included in affinities calculation during low-level tracklets association, which was unexplored last year.

2. Efforts are made to reduce computation complexity, and the runtime is shortened to 1/20.

3. We adopt a Naive Bayesian model to restrict the contribution of occluded bounding boxes to the update of appearance of tracklets. In this way, the tracking accuracy is improved.

1.3 Event Detection

The surveillance event detection system of this year is different from our previous version[3]. We use Faster R-CNN and a spring model (details in Sec. 2.3.2) as our main methods.

1.3.1 Embrace, Pointing, PersonRuns and ObjectPut

We observe that four events (Embrace, PersonRuns, Pointing and ObjectPut) can be distinguished by their key-poses as shown in Figure 2. Hence we employ the key-poses to train Faster R-CNN, whose backbone network is VGG-16 net[5] and was pre-trained on the ImageNet dataset[6]. The detection of four events is based on a similar scheme. Some differences are described as below:

1. we sample a part of negative examples from the training video clips.

2. To accelerate event detection and simplify our model, we do not adopt skip pooling strategy, which is used in pedestrian detection.

3. Specially, a PersonRuns event usually lasts a long time. In this respect, it is quite different from the other three events. In order to use temporal information to assist our detection, we merge the optical flow features into original static pictures before feeding them to the convolutional layers. Based on TDD[7] and TVL1 optical flow algorithm [8], we extract optical flow features.

To generate complete events, we employ some fusion methods to concatenate the proposal boxes obtained by Faster R-CNN. For example, we concatenate the boxes in adjacent frames if their IOU is greater than 0.5. Nevertheless, if frame interval is greater than a threshold, we generate a new event. Based on this scheme, we determine the start and end frame number of an event eventually.

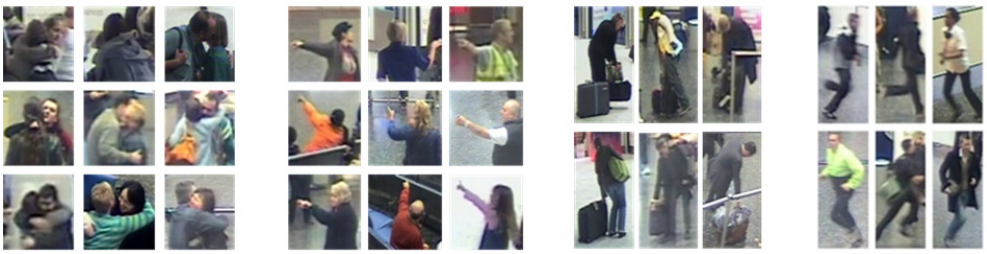

Figure 2 The samples of Embrace, Pointing, ObjectPut and PersonRuns events, from left to right.

1.3.2 PeopleMeet and PeopleSplitUp

We present a new approach for detecting “PeopleMeet” and “PeopleSplitUp” events. Different from last year’s method[3], we propose a spring model to detect above two events rather than rule-based method. Specifically, the proposed framework of “Pedestrian detection – Pedestrian tracking – Group events detection” is shown in Figure 3.

In this framework, we also exploit a CNN classifier to detect the occurrence of interaction in a group of

pedestrians. It reduces the number of false alarms while keeping the correct ones. With these modifications, the result outperforms last year's approach.

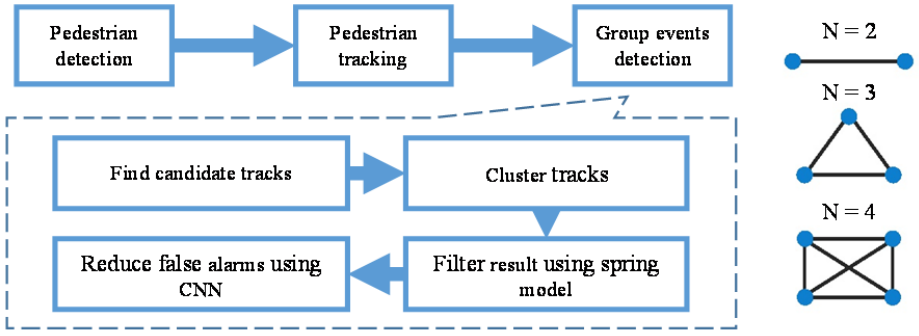

Figure 3 Left: Detection framework. The details of group events detection is shown in the dash-line box. Right: Spring model constructed on groups of 2, 3 and 4 pedestrians, respectively

The modifications carried out in "Group events detection" module are explained below:

A spring model is introduced to give an abstraction of "PeopleMeet" and "PeopleSplitUp". Concretely, a virtual spring is linked between every two people in group. For the group of N (=2, 3, 4) pedestrians, the model is built respectively. As shown in Figure 3, the spring model is actually a complete graph in which edges are springs. All springs are initialized by the same "stiffness factor" with different initial length. We set the initial energy of a group to 0 and then observe the changes of potential energy. As Hooker's Law states, the force is linearly change with the distance of a spring:

$$F = kx$$

while the potential energy of that spring is calculated as follows:

$$\mathrm{E} = \frac{1}{2}\mathrm{kx}^2$$

For a group of N pedestrians, we can compute the potential energy at time t based on the following equation:

$$E(t) = \sum_{i=1}^{N} k[x_i(t) - x_{i0}]^2$$

The intuition of this model is simple: if pedestrians come close in t, the potential energy of that group of pedestrians should gain an obvious increment. If pedestrians are moving forward side by side, the energy will have a little change, as shown in Figure 4.

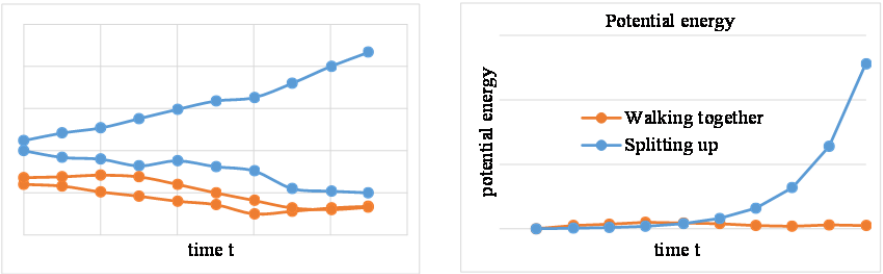

Figure 4 The evolution of potential energy. The left figure shows the tracklets of two groups of people. The pedestrians w.r.t. blue lines are splitting up while the pedestrians w.r.t. orange lines are walking together, side by side. The right figure shows the changes of corresponding potential energy.

The spring model reduces the number of parameters and is easily to be implemented. For example, we no longer need to consider the angle of motion and the number of pedestrians, while our previous approach need to tune multiple parameters manually.

We remove those groups from the results if their changes in potential energy are less than a given threshold, and then feed remained groups to the next phase for further discriminant analysis. Note that the result may contain false alarms since the spring model is unaware of the interactions of pedestrians, which drives us to design a classifier to recognize the occurrence of interactions.

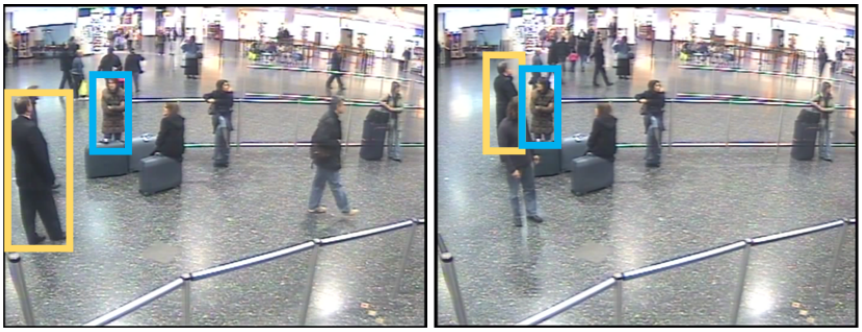

Figure 5 False alarm case. The man in yellow box is walking toward the woman in blue box, stopping by her side. The spring model would consider it as a meet event but there is no interaction between the man and the woman.

We train a CNN to recognize the "head orientation" of pedestrians. To the best of our observation on training videos, especially in camera 3 and 5, there exist lots of chances that one people walks toward another, stopping by the side of him (her) with no interaction happening. An example is shown in Figure 5. As defined in guidelines, events of this kind do not belong to "PeopleMeet" or "PeopleSplitUp". To address this issue, we train a CNN model based on the "Cifar-10" network with a 4-class softmax loss layer to detect 4 different directions of head, i.e.,. back, front, left and right.

1.4 Interactive System

The interactive system is an extension of the automatic system. This year, we completely rewrite the manual intervention interface in order to improve the labelling efficiency. The interface is shown in Figure 7.

In the interface, there are four sub-windows responsible for displaying video clips of events detected by our system automatically. The frames displayed in these four sub-windows are arranged in a particular order. Concretely, if the frame number of frame displayed in sub-window #1 is T, then the frame number of frames displayed in sub-window #2 to #4 are $T + \Delta, T + 2\Delta, T + 3\Delta$, respectively. Here, we set Δ to 25. This design endows us with a wide range of observation and make it easier to discover missing detections and false alarms. Besides utilizing four sub-windows, we also implement the addition, deletion, copying and modification of events with hot-keys provided. This gives us the ability to revise the results of automatic system efficiently.

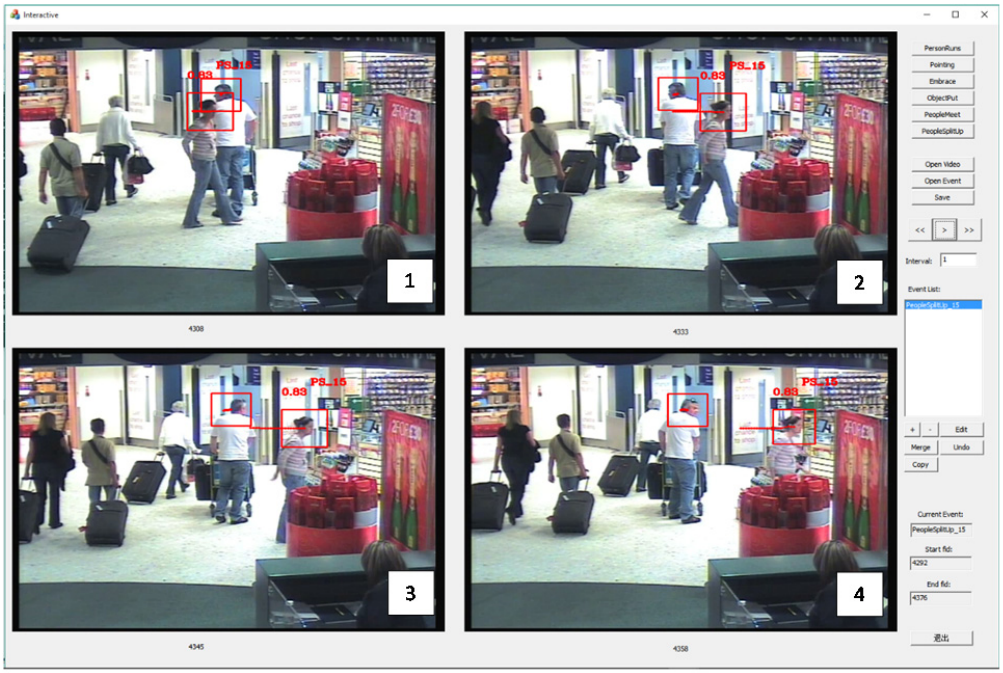

Figure 7 The interface of our interactive system.

1.5 Experimental Results

The final results of our interactive task are listed in Table 1. Our methods outperform the other systems. It is clear that CNN-based method can achieve better performances than traditional approaches.

Table 1 The actual DCR and minimum DCR of the 2016 retrospective result

Event	#CorDet	#FA	#Miss	ActDCR	MinDCR
Embrace	60	18	113	0.6622	0.6602
ObjectPut	21	54	327	0.9666	0.9646
PeopleMeet	58	114	265	0.8774	0.8774
PeopleSplitUp	36	75	140	0.8329	0.8294
PersonRuns	22	11	41	0.6563	0.6563
Pointing	156	207	773	0.9354	0.9321

1.6 Conclusion

We take the advantages of CNN to explore semantic cues and visual proofs in surveillance videos which leads to the improvement of our methods. However, there still exists a large space to improve our system and make it robust for practical usage.

2 Instance Search

This year, we propose a similar search framework for both automatic and interactive search tasks. Video key frames with a sample rate of 2 fps are extracted for retrieval. We consider the maximum frame score as the shot

score and rank the video shots as the evaluation results. To retrieve specific persons in specific locations, we explore which method and feature are more appropriate for locations or persons retrieval respectively. The results are summarized in Table 2. More details will be given in the following sections.

Table 2. Results for each run

Run ID	mAP
F_A_BUPT_MCPRL_1	19.4
F_A_BUPT_MCPRL_2	18.6
F_A_BUPT_MCPRL_3	23.0
I_A_BUPT_MCPRL_4	28.5

2.1 Locations retrieval

For locations retrieval, local and global features are extracted to describe the image content. In our experiment, we use Hessian-Affine detector with RootSIFT descriptor, MSER detector with RootSIFT descriptor and CNN features extracted from conv5 (VGG-19 model) as local features. As for global features, we use CNN features extracted from fc6. Subsequently, feature fusion scheme is followed to improve the initial retrieval performance. We use similar details to system in last year [3] for this year's location retrieval.

2.2 Persons retrieval

For persons retrieval, we need to consider how to detect persons in each frame and which features describe persons well and provide good discriminant between different persons. In our implementation, we use faces to distinguish different persons. We use dlib tool as a face detector to look for faces at multi-scale frames. Based on detected faces in each frame, we put them into convolution networks based on VGG-face [9] and Openface [10] networks. We acquire 512 and 128 dimension features respectively and conduct L2 normalization.

Since this is the first year to retrieve specific persons in specific locations, there is no ground truth to manually tune parameters for the fusion of different features. Here, we employ query-adaptive fusion scheme [11] to the faces features fusion which has been proved efficient last year [3]. This method makes progress obviously by our own observation.

Because the query for each person has 4 images and some images include side face, we conduct query expansion scheme which means the face retrieval is re-conducted for several former of the first retrieval results of query. This method helps recall more correct faces.

2.3 Merge results from locations and persons retrieval

Then, two alternative rank orders, i.e., location retrieval-person retrieval and person retrieval-location retrieval are leveraged respectively. For first case, we set a location threshold to determine which frames are relative to the location query. Then we conduct person retrieval in these frames , and vice-versa for the second case.

Afterwards, we conduct a rank fusion: we re-rank results from two orders and take higher rank position for the same keyframe. According to the evaluation results, this scheme increases the MAP about 4 percent. Finally, we consider the maximum frame score as the shot score and rank the video shots as the evaluation results.

2.4 Conclusion

This year, we adopt previous retrieval system for this year's location retrieval. We also use a query-adaptive fusion scheme to the faces features fusion, which is of great importance to identify feature effectiveness. In the next year, we will explore on more effective convolutional networks features.

3 Multimedia Event Detection

The framework of our MED system includes three parts: the frame-level feature extraction, the video-level feature representation and event classifier. We first extract the static frames from videos and get their frame-level features. Then fuse them to get video-level features. Finally, we use a trained classifier for per event to predict scores. In these parts we compare different methods, both deep learning based methods and traditional methods, and choose best ones on the validation set.

3.1 Frame-level feature extraction

For the frame-level feature extraction, we choose a very deep CNN structure. An inception structure based GoogleNet and ResNet are applied in our MED system to extract frame-level features for their high precision and good generalization ability. We remove low-level features like Improved Dense Trajectory (IDT) because they provide less information compared with CNN descriptors and are not robust for complex camera movements.

3.2 Video-level feature representation

For the video-level feature representation, we choose the traditional Vector of Locally Aggregated Descriptors (VLAD) to fuse frame-level descriptors. This method obtains a more stable performance than RNN or LSTM structure when video scenes are varied. Furthermore, it takes the advantages of generative model to represent the content of a video, and meanwhile, it does not need any labeled videos to train. Note that it can achieve better performance than netVLAD and Temporal Kennel CNN when the labeled videos are limited. Table 3 gives a comparison. It shows that compared with other traditional feature aggregating methods like Fisher Vector, the VLAD representation is faster and more concise.

Tabel 3:test on a MED16 valid subset (train on 7230 videos,test on 800 videos)

VLAD(baseline)	0.640
netVLAD	0.525
Temporal Kennel CNN	0.564

3.3 Event classifier

We use the linear Support Vector Machine (SVM) to prediction scores for every event. In this module, the labeled videos are treated as training samples to get a support vector for each event. We normalize the distance from the test sample to the support vector as the score of corresponding event.

At last, we also propose a fusion method for our different methods in above modules. Before score prediction,a vector stitching is applied to video-level features which are generated by different methods.Most unstrong video features are removed in previous modules,so the stitched feature always provides a more stable and precise prediction score than the feature generated by single method.As the result,it improves the final MAP.

3.4 Conclusion and Results

we focus on proposing a effective algorithm based on deep learning frameworks for large scale video classification.We also compared these deep learning based methods with other traditional methods,and then choosed the most suitable algorithm for our MED system.As the result,We achived the highest Mean Average Precision (MAP) both in the 10EX and 100EX Events (35.4% and 49.0%,respectively) in the c-contrast submission on MED16EvalSub Pre-Specified (PS) Events.

Tabel 4: MED2016 Pre-Specified Result

Result(MAP)	PS_SUB_10EX	PS_SUB_100EX	Platform
Our p-baseline	0.336	0.469	SML
Our c-contrast	0.354	0.490	SML

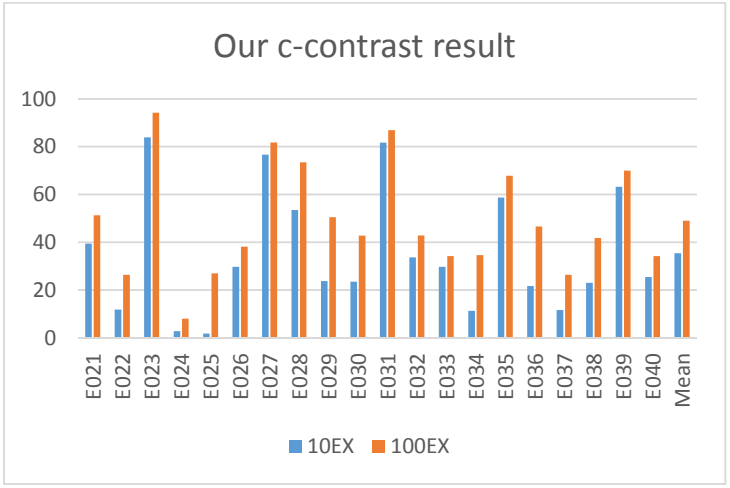

Figure 8: Our c-contrast result on MED16 EvalSub Pre-Specified (PS) Events.

References

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *Nips*, pp. 1–10, 2015.

[2] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick, "Inside-Outside Net: Detecting Objects in Context with Skip

Pooling and Recurrent Neural Networks," *Arxiv*, pp. 1–24, 2015.

[3] Q. Chen, M. Liu, X. Li, et al, "BUPT-MCPRL at TRECVID 2015," *Proc. TRECVID 2015 Work.*, 2015.

[4] B. Yang and R. Nevatia, "Multi-target Tracking by Online Learning of Non-linear Motion Patterns and Robust Appearance Models," *Comput. Vis. Pattern Recognit. (CVPR), 2012 IEEE Conf.*, vol. 1, pp. 1918–1925, 2012.

[5] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *ImageNet Chall.*, pp. 1–10, 2014.

[6] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.

[7] L. Wang, Y. Qiao, and X. Tang, "Action Recognition with Trajectory-pooled Deep-convolutional Descriptors," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. 7-12-NaN-2015, pp. 4305–4314.

[8] C. Zach, T. Pock, and H. Bischof, "A Duality Based Approach for Realtime TV-L 1 Optical Flow," *Pattern Recognit.*, vol. 4713, no. 1, pp. 214–223, 2007.

[9] Parkhi, Omkar M., Andrea Vedaldi, and Andrew Zisserman. "Deep face recognition." British Machine Vision Conference. Vol. 1. No. 3. 2015.

[10] B. Amos, B. Ludwiczuk, M. Satyanarayanan,"Openface: A general-purpose face recognition library with mobile applications", CMU-CS-16-118, CMU School of Computer Science, Tech. Rep., 2016.

[11] L. Zheng, S. Wang, L. Tian, F. He, Z. Liu, and Q. Tian. "Query adaptive late fusion for image search and person re-identification", CVPR, 2015.

[12] G. Awad, J. Fiscus, M.Michel, D.Joy, W.Kraaij, A. F. Smeaton, G.Quénot et.al., TRECVID 2016: Evaluating Video Search, Video Event Detection, Localization, and Hyperlinking. Proceedings of TRECVID 2016.