BUPT-MCPRL at TRECVID 2017*

Kaihui Zhou, Shanwei Zhao, Shizhe He, Yandong Zhu, Zhipeng Wang,
Zhongyu Fan, Xinkun Cao, Peng Li, Zhicheng Zhao, 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 for TRECVID 2017[1]. Our team participated in three tasks: surveillance event detection，instance search and multimedia event detection.

Surveillance Event Detection (SED): We submit two runs of SED task.

- **p-baseline:** Interactive results of our detection system. Based on the results of c-contrast, we manually fix false and missing detections within 25 minutes.
- **c-contrast:** Results of our retrospective detection system.

Instance Search (INS): We submit three runs for automatic search and one run for interactive search, and a brief description is as follows:

- **F_E_BUPT_MCPRL_1:** fine tune the rank score of F_E_BUPT_MCPRL_2 based on random forest classification extra score.
- **F_E_BUPT_MCPRL_2:** fine tune the rank score of F_E_BUPT_MCPRL_3 based on transcript and person re-identification extra score.
- **F_E_BUPT_MCPRL_3:** merge two rank list from location retrieval after person retrieval and person retrieval after location retrieval.
- **I_E_BUPT_MCPRL_4:** optimize rank list interactively based on F_E_BUPT_MCPRL_1.

Multimedia Event Detection (MED): We submit five 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-contrast1:** use ResNet152 instead of GoogleNet to extract feature based on p-baseline.
- **c-contrast2:** take the video with tag “relative” as positive to train the classifier based on p-baseline.
- **c-aduio:** use SoundNet to extract the audio-level features and then split into overlapping fixed length sound excerpts. A binary Support Vector Machine is used for scoring an event.
- **c-fusion:** combine the result of c-aduio and p-baseline with late-fusion.

1 Surveillance Event Detection

In this year SED evaluation, we pay attention to the detection of Embrace, Pointing, PersonRuns, ObjectPut, PeopleMeet and PeopleSplitUp. They are classified into key-pose based events and group events, which are detected by two different approaches. Explicitly, key-pose based events, which contain discriminative pose and thus can be detected by pose detectors. As to group events, the solution is based on

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

the analysis of pedestrians' tracks. Our system can be decomposed into retrospective part and interactive part. Our system achieves the best performance among all the participants in all of the six events.

1.1 Pedestrian Detection

Like last year, we take head-shoulder part detection scheme instead of the whole body, to reduce the influences of heavy occlusion in SED scenes. We choose the R-FCN detection framework as our pedestrian detecto. Compared with last year's results, we get 10% improvement and 2x faster speed in SED datasets. There are three main improvements. First, we use the ResNet-50 net as the backbone network. The feature representations of detector are more robust and discriminative. Second, we propose a hard-mining algorithm based on OHEM (Online Hard Example Mining), which selects the relatively difficult samples having large losses in the forward pathway, and activates them to update the model weights in the backward propagation. Unlike iteratively adding misclassification instances into ground-truths, the OHEM algorithm picks hard examples according to the current state of CNN model in each iteration. Hence, OHEM algorithm is more flexible and can get a better solution in the same epochs. In addition, OHEM can be conveniently embedded as a module in the end-to-end train process. Finally, the efficient design significantly reduces the memory and time cost. Comparing with our pedestrian detection result of last year, we get 10% improvement in MR-FPPI (Miss rate against false positives per image) and 2× faster speed in SED datasets.

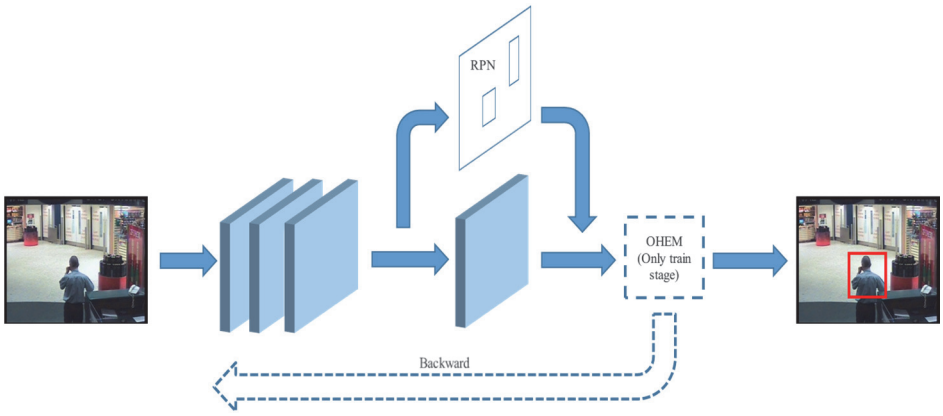

Figure 1. The architecture of our R-FCN pedestrian detector.

1.2 Pedestrian Tracking

Our multiple objects tracker extends the one we developed last year [2]. To better identify pedestrian, we explore the use of spatial-temporal cues. Furthermore, to deal with the error-prone pixel distance caused by camera distortion, we apply a scene-aware adaption model for pixel distance regularization. The two extension are described as follows.

1.2.1 Trajectory Appearance Embedding Model

Most current trackers only use the appearance cues, which are extracted at bounding box level and often suffer from the occlusion. To extract better appearance descriptors of trajectory, we design a spatial-temporal model to learn the embedding feature of trajectory rather than directly aggregate bounding box features. A feature extractor is made up of convolution layers with recurrent connections, as shown in Figure 2. In each iteration, a bounding box RoI is fed into the model. The number of recurrent iteration depends on the length of input trajectory. We use the L2 normalization result of hidden state as the embedding feature.

During training, triplet loss $L_{triplet} = [\alpha - (d_{AC} - d_{AB})]_+$ is used, where $[z]_+ = \max\{0, z\}$ and α controls the minimum distance between negative pair (d_{AC}) and positive pair (d_{AB}). The triplet inputs are

(a) (b)

Figure 2. (a) Our trajectory feature extractor. It is made up of CNN and RNN. Every time step, a bounding box of the trajectory is feed into the CNN. The number of recurrence equals to the length of the trajectory. We use the hidden state of the last time step as the embedding feature of the input trajectory. (b) The feature extractor is trained using triplet loss. During training, a triplet of trajectories is fed into three feature extractors with shared weights. The difference between two embedding features is computed using Euclidean distance.

three trajectories A, B and C. A and B come from the same pedestrian while the C is sampled from another pedestrian.

1.2.2 Distance Regularization

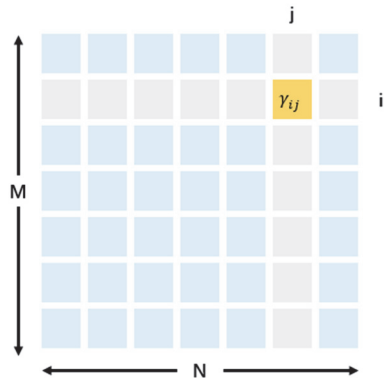

Figure 3. The image plane is decomposed into $M \times N$ grids. In each grid, we compute a scaling factor γ_{ij}.

Distance is one of the most important metrics used in multiple object tracker. In the detection of trajectories, the similarity of them should be computed to keep reasonable motion. However, the pixel-level distance cannot represent true distance since their scales depend on the closeness between camera and the object. Without camera inner-parameters, it is impossible to remap image plane to an undistorted world plane. Therefore, we seek to solve this issue approximately based on pure image cues. We first decompose the image into $M \times N$ grids as shown in Figure 3. A scaling factor is computed for each grid as described below:

We first compute the mean width (or height) $\bar{w}_{ij}$ of pedestrian detection boxes centered at grid Grid_{ij}. Then we normalize all the values to range between 0 and 1. We denote the normalized value of Grid_{ij} to be $\bar{w}_{ij}^{\Delta}$ and the minimum value to be $\bar{w}_{i^*j^*}^{\Delta} = \arg\min_{i,j} \bar{w}_{ij}^{\Delta}$. And finally we set the scaling factor of Grid_{ij} to be $\gamma_{ij} = \bar{w}_{i^*j^*}^{\Delta} / \bar{w}_{ij}^{\Delta}$. To compute the regularized distance between two objects A and B located at Grid_A and Grid_B respectively, we simply need to evaluate the integral below:

$$D_{AB} = \int_A^B \gamma_{grid(s)}\, ds$$

1.3 Event Detection

We split the six events into key-pose based event and group event. We formulate the detection of key-pose based event as object detection problem while detect group event based on trajectory analysis. To better deal with false alarms in the detection of ObjectPut, we also employ a spatial-temporal network for false alarm filtering.

1.3.1 Embrace and Pointing

We use R-FCN as our key-pose detector, whose backbone network is VGG-16 net. Instead of OHEM, we employ offline hard example mining, and we adopt multi-class (Embrace, Pointing, PersonRuns and ObjectPut) training strategy to train our model. This strategy has two advantages. On one hand, the multi-class model can detect key-poses of different events at the same time. On the other hand, for the key-poses of some different events are similar to a certain degree, this strategy forces the model to further distinguish the differences among the key-poses of these events. It reduces the number of false alarms because the model will not confuse similar key-poses of different class.

1.3.2 PersonRuns

For PersonRuns Event, the overall framework and network share similar configurations as that introduced in pedestrian detection (see Figure 1). To incorporate the appearance and motion information of PersonRuns event, we merge the optical flow features into the original static images as the input. Beyond that, to combine temporal information, we stack consecutive t frames surrounding the current input before feeding them to convolutional layers.

1.3.3 ObjectPut

For ObjectPut Event, the detecting system mainly consists of two stages: key-pose detection and key-

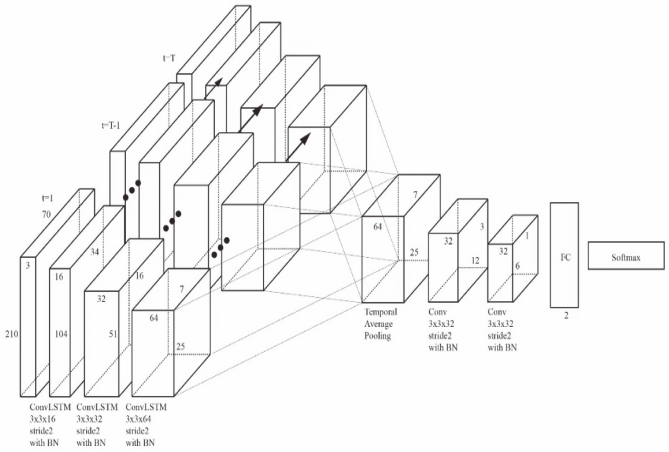

Figure 4. The architecture of CLITP network. We place three-ConvLSTM layers at the beginning for spatio-temporal feature extraction. By applying temporal average pooling, we can fuse the features of all the time steps and summarize the feature of the entire sequence.

pose sequence classification. In the first stage, we detect key-poses of ObjectPut event like other key-pose based events. By concatenating these key-poses in chronological order, we can retrieve series of key-pose sequences. In the second stage, we propose a CLITP [6] (ConvLSTM Integrated with Temporal Pooling, CLITP) model (shown in Figure 4) to obtain temporal representations and classify these key-pose sequences. In this way, our architecture can reduce the number of false positives while maintain the number of true positives with the combination of spatial and temporal information.

1.3.4 PeopleMeet and PeopleSplitUp

We follow the pedestrian trajectory-based spring model approach presented in last year's evaluation [2]. We briefly describe our method below.

PeopleMeet and PeopleSplitUp have completely different properties compared to the other four events. For example, undetermined number of objects and no discriminative pose exists make it impossible to solve in an object detection framework. We find that the occurrence of PeopleMeet and PeopleSplitUp can be

modeled using a spring system in which the pedestrians are connected by virtual springs. When pedestrians are meeting (or splitting up), the springs will be squeezed (stretched), giving rise to the increment of potential energy of the whole system. Note that the system can be applied to any number of pedestrians.

.

1.4 Evaluation Results

The overall result on this year's evaluation shows that our proposed method is promising and effective in all the six events. The result is shown in Table 1. It can be proved that deeper architecture with R-FCN achieves better result in the detection of key-pose based events. In addition, the proposed CLITP model provides a better filtering for large amount of false alarms. Also, with the help of more accurate trajectory appearance model, our tracking method also gives rise to the better detection of group events.

Table 1. Evaluation result of our proposed method.

	Inputs			Actual Decision DCR Analysis							
Title	#Targ	#NTarg	#Sys	#CorDet	#Cor!Det	#FA	#Miss	RFA	PMiss	DCR	Dec. Tresh
Embrace	173	20	91	71	0	20	102	1.99718	0.590	0.5996	0.7380
ObjectPut	348	50	76	26	0	50	322	4.99295	0.925	0.9503	0.6010
PeopleMeet	323	185	249	64	0	185	259	18.47393	0.802	0.8942	0.7380
PeopleSplitUp	176	115	141	26	0	115	150	11.48380	0.852	0.9097	1.0000
PersonRuns	63	14	38	24	0	14	39	1.39803	0.619	0.6260	0.9620
Pointing	929	148	277	129	0	148	800	14.77914	0.861	0.9350	0.9300

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. To retrieve specific persons in specific locations, we need to consider which methods and features are more appropriate for locations or person 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_E_BUPT_MCPRL_1	39.1
F_E_BUPT_MCPRL_2	38.7
F_E_BUPT_MCPRL_3	37.7
I_E_BUPT_MCPRL_4	51.2

2.1 Location retrieval

For locations retrieval, we use two independent methods to extract features. The first one extracts local and global features to describe the image. In our experiment, we use Hessian-Affine detector with RootSIFT descriptor, MSER detector with RootSIFT descriptor and CNN features extracted from conv5 of AlexNet as local features. Then we adopt Bag of Words (BoW) model on local features to represent the images. As for global features, we adopt spatial pyramid pooling [3] and cross-dimensional weighting [4] for convolutional layers of ResNext [5]. For the second method, we fine-tune a publicly available VGG-16

model, which has been trained on Places365 dataset, to fit with the task. We then extract the fc6 features for location retrieval. Subsequently, feature fusion scheme is followed to improve the initial retrieval performance.

2.2 Person retrieval

As for person retrieval, we use three kinds of methods including face retrieval, person re-identification and transcript-based search.

For face retrieval, we first apply face detection based on dlib functions. Then we extract face features using dlib's state-of-art face recognition model built with deep learning. Some re-sample strategies are also considered in this step. Then we adopt query expansion for face retrieval that conducts retrieval again for top N retrieval results of query.

For person re-identification, we use the Faster R-CNN method to detect persons in the video and then fine-tune the model with query examples and video key frames. Finally, fully-connected layer features are extracted for retrieval. The frames, whose person re-identification scores are higher than a threshold, will be added a score to the final scores.

For transcript-based search, we search the person name of query in the transcript. For each person, the shots whose transcripts including the name will be added a score to the final scores.

2.3 Merge results from location and person retrieval

To combine results of locations retrieval and person retrieval, we could first retrieve locations based on location queries and set a location threshold to determine which frames are relative to the location query. Then we conduct person retrieval in these frames. Correspondingly, we can first retrieve persons, set a person threshold and then conduct location retrieval.

Since we can get two alternative ranks by using different order of locations retrieval and person retrieval, we then conduct a rank fusion. We cross the rank results from tow orders and take higher rank position for the same frame. We then apply some re-ranking skills based on person re-identification, transcript and random forest classification to the final scores. For random forest classification, we conduct binary classification for each frame based on multiple scores of location and person scores. We train the random forest based on 2016 instance search topics. Then the frames, whose classification labels are 1, will be added a score to the final scores.

Finally, we consider the maximum frame score as the shot score and rank the video shots for evaluation.

2.4 Conclusion

This year, we optimize last year's retrieval system [2] from multiple perspectives. We optimize the convolutional neural network (CNN) and to extract more powerful features. We also combine more information for person retrieval.

3 Multimedia Event Detection

The framework of our MED system includes three parts: the vision-level feature extraction, the sound-level feature extraction and the design of the event classifier. We extracted two different types of features, and then fuse them with a late fusion scheme. Finally, we use a trained classifier for per event to predict the score. The results are summarized in Table 3. More details will be given in the following sections.

Table 3. Results for MED

Result (InfAP200%)	PS_10EX	Platform
Our c-contrast1	0.255	SML
Our p-baseline	0.203	**SML**
Our c-fusion	0.184	**SML**

3.1 Vision-level feature

For the Vision-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. After extracting frame-level features from sample, 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.

3.2 Sound-level feature

For the Sound-level feature representation, we use a deep convolutional architecture - SoundNet for audio feature extraction. The network learns rich natural sound representations by capitalizing on large amounts of unlabeled sound data collected in the wild, leveraging the natural synchronization between vision and sound to learn an acoustic representation using two-million unlabeled videos. It has proved SoundNet has a very good performance on three standards, publicly available datasets: DCASE Challenge, ESC-50, and ESC-10. For better performance, a standard data augmentation is conducted where each training sample is split into overlapping fixed length sound excerpts, which are used to extract feature and train. During inference, we averagely predict across all windows as the final score.

3.3 Event classifier

We use the linear Support Vector Machine (SVM) to predict 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.

3.4 Conclusion and Results

In this year, we focus on evaluating the power of the sound representation and proposing an effective algorithm based on deep learning framework in the large-scale video classification. From the evaluation results, we achieved the second in the 10EX Events (25.5%) in the c-contrast1 submission on MED17EvalPre-Specified (PS) Events. However, the result of sound representation (c-fusion) is far from satisfaction, even though it actually gains a little better performance on categories E53, E55, E57, compared with the vision representation only (p-baseline).

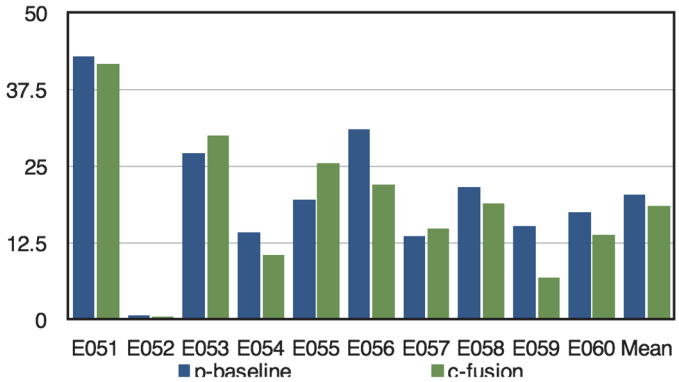

Figure 5. p-baseline result and c-fusion result

References

[1] George Awad, Asad Butt, Jonathan Fiscus et.al. TRECVID 2017: Evaluating Ad-hoc and Instance Video Search, Events Detection, Video Captioning and Hyperlinking, Proceedings of TRECVID 2017.

[2] Zhicheng Zhao et al. "BUPT-MCPRL at TRECVID 2016." Proc. TRECVID. 2016.

[3] Kaiming He et al. "Spatial pyramid pooling in deep convolutional networks for visual recognition." European Conference on Computer Vision. Springer, Cham, 2014.

[4] Kalantidis Yannis, Clayton Mellina, and Simon Osindero. "Cross-dimensional weighting for aggregated deep convolutional features." European Conference on Computer Vision. Springer International Publishing, 2016.

[5] Saining Xie et al. "Aggregated residual transformations for deep neural networks." arXiv preprint arXiv:1611.05431 (2016).

[6] K. Zhou, Y. Zhu, and Y Zhao. A Spatio-temporal Deep Architecture for Surveillance Event Detection Based on ConvLSTM. VCIP, 2017