WARD@TRECVID 2016 Surveillance Event Detection

Xingzhong Du[1], Yi Yang[2], Xiaofang Zhou[1]
[1]The University of Queensland, [2]University of Technology Sydney

I. Abstract

In this year's competition [1], we made three changes based on last year's system [2]. The first change was using original videos instead of resized videos. The second change was training the code books under separated cameras. The final change was performing inner vector normalization after feature encoding. In the cross validation, some of the proposed changes had impact on the detection and together made the fusion more accurate. However, in the competition, this year's submission was inferior to the last year's submission. We investigated the submission and found that it was caused by using the original videos, which was proposed for the intention of increasing the motion saliency in the features. As a consequence, we will not use the original videos to extract features for the next round.

II. This Year's Retrospective System

The whole pipeline is same as last year but with three changes. In this section, we will explain why we made these changes for this year's submission, and display how these changes influenced the detection accuracy. For convenience, all the results reported in this part are measured by the actual detection cost rate (aDCR) [1].

A. Video Preprocessing

TABLE I. Detection under resized and original videos

	resize		origin	
event	dtwfv	idtwfv	dtwfv	idtwfv
CellToEar	1.0009	1.0018	1.0026	1.0022
Embrace	1.0050	1.0131	0.9197	0.9774
ObjectPut	1.0057	1.0059	1.0154	1.0218
PeopleMeet	0.9589	0.9516	0.9634	0.9710
PeopleSplitUp	0.9610	0.9595	0.9798	0.9752
PersonRuns	0.6634	0.6458	0.6193	0.6119
Pointing	0.9890	0.9956	0.9887	0.9908

The video preprocessing prepares the input for the feature extraction. In particular, each clip's length is 60 frames and 30-frame overlapped with the adjacent clips. It is worth noting that [3], [4] track the feature points for 15 frames by default. Therefore, for each video clip, we append 15 subsequent frames behind the original 60 frames. After the sliding, roughly 350k clips are generated. The clips whose size are 0 are removed at the end of video preprocessing. In the last year's system, all the video clips are resized to 320×240. This accelerates the feature extraction. However, the motion saliency is weaken by the resizing. This year, we use the original videos to increase the saliency in the extracted features. The results from the cross validation shows that features extracted from the original videos are significantly beneficial to the detection of Embrace and PersonRun.

[1]For more details about how to calculate aDCR, please refer to tftp://jaguar.ncsl.nist.gov/pub/SED16/TRECVid-SED16-EvaluationPlan.pdf

B. Code Book Learning

TABLE II. Detections under different code books

	all cameras		separated cameras	
event	dtwfv	idtwfv	dtwfv	idtwfv
CellToEar	1.0009	1.0018	1.0028	1.0016
Embrace	1.0050	1.0131	1.0052	1.0122
ObjectPut	1.0057	1.0059	1.0065	1.0070
PeopleMeet	0.9589	0.9516	0.9653	0.9586
PeopleSplitUp	0.9610	0.9595	0.9528	0.9465
PersonRuns	0.6634	0.6458	0.6665	0.6423
Pointing	0.9890	0.9956	0.9851	0.9901

We use fisher vector [5] to encode the raw features. During the encoding process, one important component is the code book. In last year's system, the code book was learned on the sampled feature points from the random video clips under all the cameras. Considering that the angles of the cameras are different, we change the input of the code book learning. In particular, we learn a code book for each camera. That is to say, given N cameras and K features, now we choose to learn $N \times K$ code books for fisher encoding instead of previous N code books. We expect such separated code books capture more camera-specific information to improve the detection. To investigate the impact, we use resized videos with all-camera code book as the baseline and compare the results from the those from resized videos with separated-cameras codebooks. The results reported in Table II show that this change did not have significant impact on the detection accuracy.

C. Feature Encoding

This year, we still extract dense trajectory [3] and improve dense trajectory [4] to perform detection. This method consists of four steps. The first step learns a projection matrix based the raw features by whiten Principle Components Analysis (whiten PCA). After dimension reduction, the dimension of the raw features are reduced by a factor of two. The second step learns a code book which is Gaussian Mixture Model (GMM) based on the reduced features. The components in GMM act as the visual words for inferring the soft assignment information, and the amount of the components is set to 256. The third step transforms the reduced features of a clip into fisher vectors by the soft assignment information, and averages them into one fisher vector for this clip. The last but not the least step is the normalization. Existing normalization process includes power and l_2 normalization [5]. In this year's submission, we apply a further inner normalization. This applies 5 separated

normalization processes on the whole fisher vectors. This is the because the raw feature have 5 components, namely, trajectory, HOG, HOF, MBHx and MBHy. We want to make the discriminative powers from different features be equal in the learning. The results are reported in Table III. It indicates that this change had impact on Embrace, PeopleMeet and PersonRun.

TABLE III. DETECTIONS UNDER DIFFERENT NORMALIZATIONS

	power + l_2		inner power + inner l_2	
event	dtwfv	idtwfv	dtwfv	idtwfv
CellToEar	1.0009	1.0018	1.0002	1.0028
Embrace	1.0050	1.0131	0.9614	0.9829
ObjectPut	1.0057	1.0059	1.0128	1.0092
PeopleMeet	0.9589	0.9516	0.9470	0.9649
PeopleSplitUp	0.9610	0.9595	0.9723	0.9694
PersonRuns	0.6634	0.6458	0.6339	0.6434
Pointing	0.9890	0.9956	0.9906	0.9864

D. Submission Generation

The generation of submission starts with model training where detector are created for each event under Camera 1, 2, 3, 5 on different feature settings. The training consists of three steps. In the first step, we treat the clips which have 50% overlap with the ground truth as the positive, then use LIBLINEAR [6] to train detectors and two-fold cross-validation to choose parameters. But in LIBLINEAR, we need to implement the probability function by ourselves. In this year's submission, the probability function is implemented as [7], which is more robust than the curve fitting. The python code can be downloaded from *https://github.com/domainxz/pytools.git*. We verify this code by reproducing the action recognition experiment in [4]. The results show this code can work properly. The third step of model training learns a threshold for each detector, then applies Non-Maximum Suppression (NMS) to merge the adjacent positive clips. When the model training is finished, We will have 7×4 detectors per feature. Each of them only focuses on one event under one camera. Considering the three changes we made this year. There are 8 detection results for each event under each camera.

The final submission is a combination of the selected detection results. This year, we selected the detections on the original videos with different codebooks and normalization methods. Table IV reports the fusion results from our validation experiment and final competition. In a word, even though the proposed method improved the detection in the validation, but it failed to improve in the competition.

TABLE IV. FUSION UNDER LAST YEAR'S AND THIS YEAR'S SETTINGS

	validation		submission	
event	2015	2016	2015	2016
CellToEar	1.0013	1.0008	1.0046	1.0140
Embrace	0.9251	0.8409	0.8680	0.8646
ObjectPut	1.0034	1.0133	1.0160	1.0044
PeopleMeet	0.9172	0.9200	0.8939	0.9269
PeopleSplitUp	0.8821	0.8712	0.8934	0.8909
PersonRuns	0.6426	0.5325	0.5768	1.0303
Pointing	0.9869	0.9826	1.0140	1.0057

III. RESULT ANALYSIS

It is the first time we saw such a big performance divergence between validation and final test. So we looked inside the aDCR and expected to get some clues. We extracted the amount of ground true, true positive and false positive from the returned results and reported them in Table V. Because the test data is an increment to last year's, the comparison shows that this year's submission made too much false positives than last year while decreased considerable true positives. We think this is because we extract features on the original videos. The motion saliency indeed increases but does not make the detection improve. Next time, we will roll back to use the resized videos.

TABLE V. GROUND TRUE (GT), TRUE POSITIVE (TP) AND FALSE POSITIVE (FP) COMPARISON

	2015			2016		
event	GT	TP	FP	GT	TP	FP
CellToEar	54	0	8	77	0	28
Embrace	138	62	552	173	42	215
ObjectPut	289	7	70	348	3	26
PeopleMeet	256	100	495	323	92	424
PeopleSplitUp	152	57	467	176	41	248
PersonRuns	50	28	238	63	27	1237
Pointing	794	35	101	929	30	76

IV. ACKNOWLEDGMENT

This work was supported by the ARC project (Grant No. DP150103008).

REFERENCES

[1] G. Awad, J. Fiscus, M. Michel, D. Joy, W. Kraaij, A. F. Smeaton, G. Qunot, M. Eskevich, R. Aly, G. J. F. Jones, R. Ordelman, B. Huet, and M. Larson, "Trecvid 2016: Evaluating video search, video event detection, localization, and hyperlinking," in *Proceedings of TRECVID 2016*. NIST, USA, 2016.

[2] X. Du, X. Li, X. Zhou, and A. Hauptmann, "Ward-cmu @ trecvid 2015," in *Proceedings of TRECVID 2015*, 2015.

[3] H. Wang, A. Kläser, C. Schmid, and C. Liu, "Action recognition by dense trajectories," in *CVPR*, 2011, pp. 3169–3176.

[4] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *ICCV*, 2013, pp. 3551–3558.

[5] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *ECCV*, 2010, pp. 143–156.

[6] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin, "LIBLINEAR: A library for large linear classification," *JMLR*, vol. 9, pp. 1871–1874, 2008.

[7] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *ADVANCES IN LARGE MARGIN CLASSIFIERS*. MIT Press, 1999, pp. 61–74.