PKU_ICST at TRECVID 2017:
Instance Search Task

Yuxin Peng, Xin Huang, Jinwei Qi,

Junchao Zhang, Junjie Zhao, Mingkuan Yuan,

Yunkan Zhuo, Jingze Chi, and Yuxin Yuan

Institute of Computer Science and Technology,

Peking University, Beijing 100871, China.

pengyuxin@pku.edu.cn

Abstract

We participated in all two types of Instance Search (INS) task in TRECVID 2017: automatic search and interactive search. For the automatic search, our approach consists of two stages: **similarity computing** and **result re-ranking**. In the similarity computing stage, we first conducted two search processes: location-specific search and person-specific search, and then the final results were obtained by score fusion of the above two processes. In the location-specific search process, we adopted two methods to get the location score: (1) Approximate K-means (AKM) based search using handcrafted features, and (2) DNN-based search using deep features. In the person-specific search process, we also adopted two methods, namely (1) deep face recognition, and (2) text-based search. After getting the scores of location-specific and person-specific search, we conducted instance score fusion from two directions: searching the specified person based on location-specific results, and searching the specified location based on person-specific results. Then we further integrated the above two directions to obtain the final instance search results, which can simultaneously capture the cues of both location and person. In the result re-ranking stage, we applied semi-supervised learning based re-ranking method, which aims to filter noisy shots in the top ranked results, for improving the accuracy of automatic search. For the interactive search, we first used the same approach with the similarity computing stage in automatic search to get the original query scores. Then, we also adopted interactive query expansion strategy, and the scores of expanded and original queries were merged to effectively boost the search accuracy. The official evaluations showed that our team ranked 1st in both automatic and interactive search.

1 Overview

In TRECVID 2017[1], we participated in all two types of Instance Search (INS) [2] tasks: automatic search and interactive search. In both automatic search and interactive search, our team ranked 1st among all teams.

For the automatic search, our approach consists of two stages: **similarity computing** and **result re-ranking**. In the similarity computing stage, we first conducted two search processes: **location-specific search** and **person-specific search**, and then performed score fusion from two directions: searching the specified person based on location-specific results, and searching the specified location based on person-specific results. We further integrated the above two directions to obtain the final instance search results, which can simultaneously capture the cues of both location and person. In the result re-ranking stage, we applied semi-supervised learning based re-ranking method, which aims to filter noisy shots in the top ranked results, for improving the accuracy of automatic search. For the interactive search, we adopted interactive query expansion strategy based on automatic search, and the scores of expanded and original queries were merged to effectively boost the search accuracy.

We totally submitted 7 runs including 6 automatic runs and 1 interactive run. The official evaluation results of our 7 runs are shown in Table 1. Table 2 gives the detailed explanation of brief descriptions in Table 1. The framework of our system is shown in Figure 1. In the 6 automatic runs, the notations “A” and “E” specify whether the video examples were used or not, and the methods of two runs are the same if the only difference between them is the notation “A” or “E”. The difference between Run1_A/E and Run2_A/E is that Run2_A/E incorporates text-based person search method based on the methods of Run1_A/E. Compared to Run2_A/E, Run3_A/E applies semi-supervised learning based re-ranking strategy to improve the search results. Run4 is an interactive search run with human feedback based on automatic search.

Table 1: Results of our submitted 7 runs on Instance Search task of TRECVID 2017.

Type	ID	MAP	Brief description
Automatic	PKU_ICST_RUN1_A	0.448	A+D+F
	PKU_ICST_RUN1_E	0.471	A+D+F
	PKU_ICST_RUN2_A	0.531	A+D+F+T
	PKU_ICST_RUN2_E	0.549	A+D+F+T
	PKU_ICST_RUN3_A	0.528	A+D+F+T+S
	PKU_ICST_RUN3_E	0.549	A+D+F+T+S
Interactive	PKU_ICST_RUN4	0.677	A+D+F+T+H

Table 2: Description of our method.

Abbreviation	Description
A	**A**pproximate K-means (AKM) based location search
D	**D**NN-based location search

F	**F**ace recognition
T	**T**ext-based person search
S	**S**emi-supervised learning based re-ranking
H	**H**uman feedback

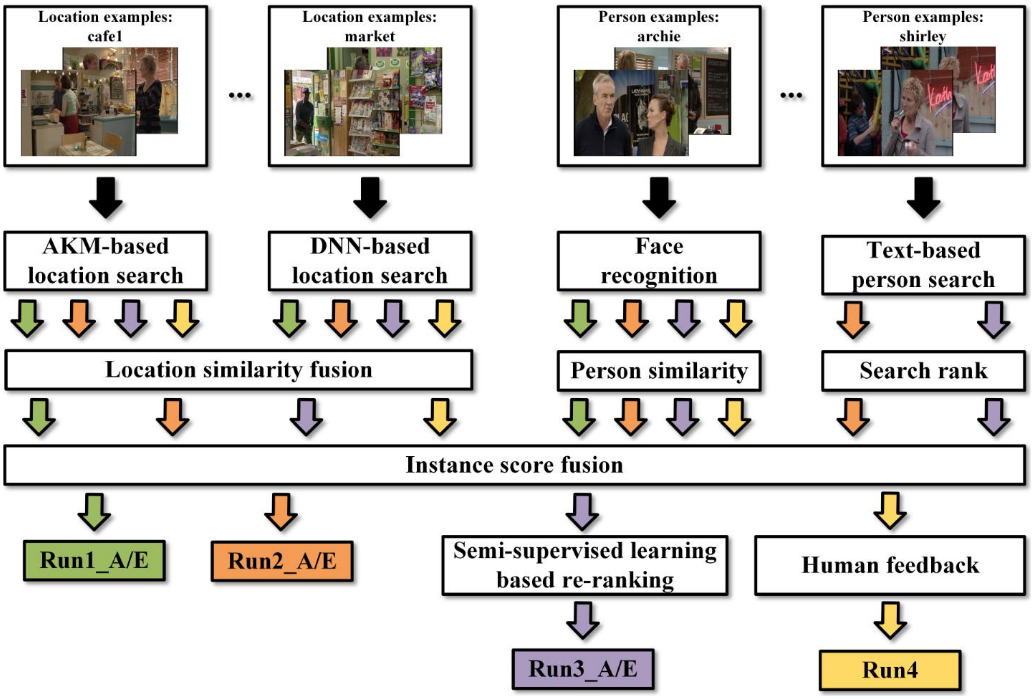

Figure 1: Framework of our instance search approach for the submitted 7 runs.

2 Similarity Computing Stage

2.1 Location-specific Search

We adopted two methods to conduct location-specific search, namely AKM-based search and DNN-based search, which used location features from handcrafted features and deep features respectively. Finally, we took the advantages of both the two methods by fusion strategy.

2.1.1 AKM-based Search

For AKM-based location search, we first extracted keypoint-based BoW features by AKM algorithm, and then conducted location search by cosine distance measurement. The keypoint-based BoW features of shots were extracted in the following three steps:

(1) First, we extracted the keypoint features of each key frame. Three detectors including Harris Laplace[3], Hessian Affine[4] and MSER[5] with two descriptors including 128-dimensional SIFT descriptor[6] and 192-dimensional ColorSIFT descriptor[7]. So there were 6 kinds of keypoint features generated for each key frame as a result.

(2) Second, for each kind of keypoint feature, we used AKM algorithm to cluster them into

one-million cluster centroids and built a visual vocabulary with these cluster centroids.

(3) At last, as each shot consisted of several key frames, we assigned each keypoint of all the key frames into the nearest centroid, where the word weights were determined by the keypoint-to-word similarity and region of interest (ROI), and generated a one-million dimensional BoW feature. As a result, we generated 6 BoW features for each shot corresponding to 6 kinds of keypoint features, as shown in Figure 2.

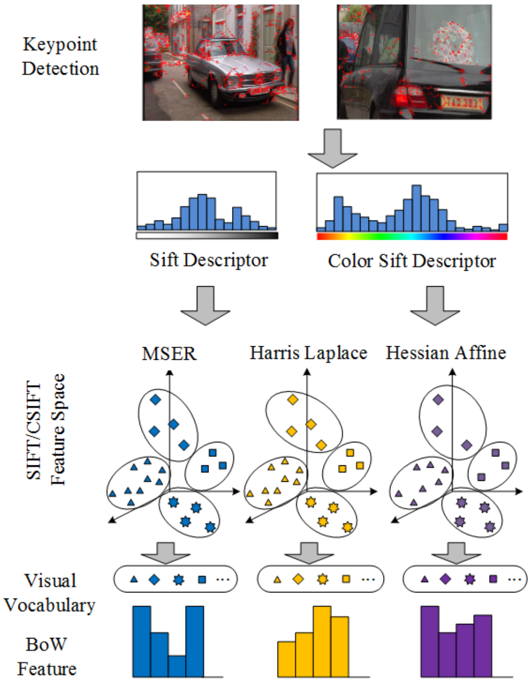

Figure 2: Extracting keypoint-based BOW features

After getting 6 kinds of keypoint-based BoW features, we conducted location search by cosine distance measurement based on each kind of keypoint-based BoW feature and obtained 6 groups of similarity scores. Finally, we integrated them using late fusion strategy to get the AKM-based location similarity:

$$AKM_{ij} = \frac{1}{6}\sum_k BOW_{ij}^{(k)} \quad (1)$$

where AKM_{ij} denotes the AKM-based location similarity score between shot *i* and location *j*. $BOW_{ij}^{(k)}$ denotes the similarity score between shot *i* and location *j* based on *k*th BoW feature.

2.1.2 DNN-based Search

We also exploited DNN models to promote location-specific search. Three widely-used DNN models, namely VGG-16[8], GoogLeNet[9], and ResNet-152[10], were adopted to extract location-specific DNN features. We also adopted progressive training strategy to further improve the discrimination of models. Our framework included two stages: model training and location search.

(1) Model Training

In the training stage, (1) we pre-trained the DNN models with the Places365[11] dataset, which is a large dataset for scene understanding. The models could be properly initialized to fit for the location-specific search task. (2) We made various kinds of transformations, including image blurring, noising and scaling, etc., on the provided location example images, which formed the training set to fine-tune the pre-trained models. (3) We adopted progressive training strategy for promoting the discrimination of the models. First, we conducted the location search using the features extracted from the models obtained by previous step. Then the frames in the top N (set to be 20 here) returned shots as well as their transformations were selected to expand the training set. Finally, we continued to fine-tune the models using the newly expanded training set. The selected shots increased the size and diversity of the training set, to promote the models to learn more discriminative features.

(2) Location Search

In the location search stage, we extracted DNN features from three fine-tuned models respectively, and calculated the similarity scores based on cosine distance measure. For exploiting the complementarity among the three models, we averagely fused the similarity scores of them to get the DNN-based location similarity:

$$DNN_{ij} = \ 1/3\left(VGG_{ij} + GOOGLE_{ij} + RESNET_{ij}\right) \qquad (2)$$

where DNN_{ij} denotes the DNN-based location similarity score between shot i and location j. VGG_{ij}, $GOOGLE_{ij}$ and $RESNET_{ij}$ denote the similarity scores based on VGG-16, GoogLeNet, and ResNet-152 features respectively.

2.1.3 Location Similarity Fusion

After obtaining the scores of AKM-based and DNN-based location similarity for each location example and test shot, we performed late fusion of these two kinds of similarity scores to get the final location similarity $sim_{ij} = 2 \cdot AKM_{ij} + DNN_{ij}$, and further obtained the retrieval ranking of location-specific search.

2.2 Person-specific Search

In person-specific search process, we adopted two kinds of methods, including face recognition and text-based search.

2.2.1 Deep Face Recognition

We detected faces from the video key frames by MTCNN[12], and extracted a 4096-dimensional feature vector for each face image by VGG-Face model[13]. The faces in query person examples were also detected and represented as 4096-dimensional feature vectors. However, we noticed that there were some "bad" faces in query examples, which caused bad retrieval performance due to blur and difficulties to distinguish. So we detected and removed such “bad” faces by following strategy: For the given 4 query faces of a specific person, we calculated the similarity s_{ij} between i-th and j-th query face. We defined $c_i = \sum_{j \neq i} s_{ij}$ as the confidence score of the i-th query face. Then the i-th query face would be detected as “bad” face if $c_i + \theta < \sum_{j \neq i} c_j / 3$, where θ was set to be 0.05 here.

Then, similar to DNN-based location search, we adopted progressive training strategy for promoting the discrimination of the VGG-Face model. First, we conducted the person search using the features extracted from the VGG-Face model. Second, the faces in the top N (set to be 50 here) returned shots were selected to form the training set, then used to fine-tune the VGG-Face model, and the fine-tuned model were denoted as VGG-Face-ft model.

Finally, we extracted a 4096-dimensional feature vector based on the VGG-Face-ft model for each face image, and calculated the similarity cos_{ij} between the shot i and person j. The faces in the top N (set to be 200 here) returned were selected to train an SVM[14] model, and svm_{ij} denotes the classification score of predicting shot i as person j. We combined the similarity and classification scores to get the final similarity as $sim_{ij} = cos_{ij}+svm_{ij}$.

2.2.2 Text-based Search

The process of text-based search was similar to that we performed last year[15]. We used the transcripts of videos provided by NIST to perform text-based person search for each topic. Besides the person’s name pointed out explicitly by the topic, we extended the person’s information for search by retrieving some structured data from related Wikipedia webpage, such as nick name, character name, names of the specific person’s family and his/her closest friends, etc. For each topic, we generated a list of shots whose transcripts included the keywords of the topic. The text-based search results were used for instance score fusion. Please see Section 2.3 for details.

2.3 Instance Score Fusion

So far we have gotten the location similarity from the location-specific search, as well as the person similarity from the person-specific search. This year, each query topic was to find a given person in a given place, which required the fusion of both location and person similarity. Our fusion strategies that comprehensively considered the location and person similarity were adopted as the following two directions:

(1) We searched the specified person from candidate location shots. We first got candidate location shots with a considerable probability to contain the specified locations. Specifically, top-N (N > 1000) returned shots were selected as the candidate location shots from the location-specific results. Then, we used text-based person search results to modify person similarity and got the score s_1 as following:

$$s_1 = \mu \cdot \text{similarity}_{\text{person}} \tag{3}$$

where μ was the text-based bonus parameter. We set $\mu > 1$ if the shot existed in text-based person search results, and $\mu = 1$ otherwise. By the text-based bonus parameter μ, we could use the auxiliary information from the text to retrieve the shots with query topic. Next we rank the candidate location shots by the score s_1 to generate the location-based ranking list. Moreover, for those shots not included in top-*N* location-specific results, we set $s_1 = 0$.

(2) We searched the specific location from candidate person shots. Similarly, top-*M* (*M* > 1000) returned shots from the person-specific results were selected as the candidate person shots with a considerable chance to contain the specified persons. We also used text-based person search results to modify location similarity and got the score s_2 as following:

$$s_2 = \mu \cdot \text{similarity}_{\text{location}} \tag{4}$$

Then we used s_2 to rank the candidate person shots to generate the person-based ranking list. Similarly, we set s_2 to be 0 for the shots not in the top-*M* person-specific results.

(3) Moreover, in order to make full use of location-based ranking list and person-based ranking list to improve the search performance, we proposed a fusion strategy on s_1 and s_2. In this strategy, the fusion score of a shot would be calculated as:

$$s_f = \omega(\alpha s_1 + \beta s_2) \tag{5}$$

where α and β were weight parameters to balance s_1 and s_2, and ω was a bonus parameter. We set $\omega > 1$ if the shot simultaneously existed in the top-*N* location-specific results and top-*M* person-specific results, otherwise $\omega = 1$. The bonus parameter ω could help to highlight the common shots of both location-specific and person-specific search results, which were more likely to be the right instances. It could be easily seen that the final instance score preserved information of both location and person aspects, and the fusion of them could improve the instance search accuracy.

(4) Finally, we proposed a time sequence based ranking algorithm to refine the fused results. It was inspired by the fact that the information in video is continuous, and adjacent shots in time sequence usually contain similar content. In detail, we recalculated the scores of shots in fusion score ranking list as following:

$$s_f^{(i+k)} = s_f^{(i)} + \theta k \tag{6}$$

where $s_f^{(i)}$ denoted the score of i-th shot and $s_f^{(i+k)}$ denoted the score of $(i + k)$-th shot, namely i-th shot's neighbor shot in time sequence. k was the index gap between these two shots $(-T < k < T)$ and θ was a parameter to adjust the similarity score. We used the adjusted scores to rank shots and got the final shot ranking list, which could enhance the instance search performance.

3 Result Re-ranking Stage

In result re-ranking stage, we adopted semi-supervised re-ranking algorithms to refine the ranked results obtained from above fusion step. We noted that there were still a few noisy shots in the top ranked shots, although most of the top ranked shots were correct and looked similar to each other. In order to filter these noisy shots, we further proposed a semi-supervised learning based re-ranking algorithm[15] to refine the top-ranked results. The detail of this algorithm was described as below and Figure 3:

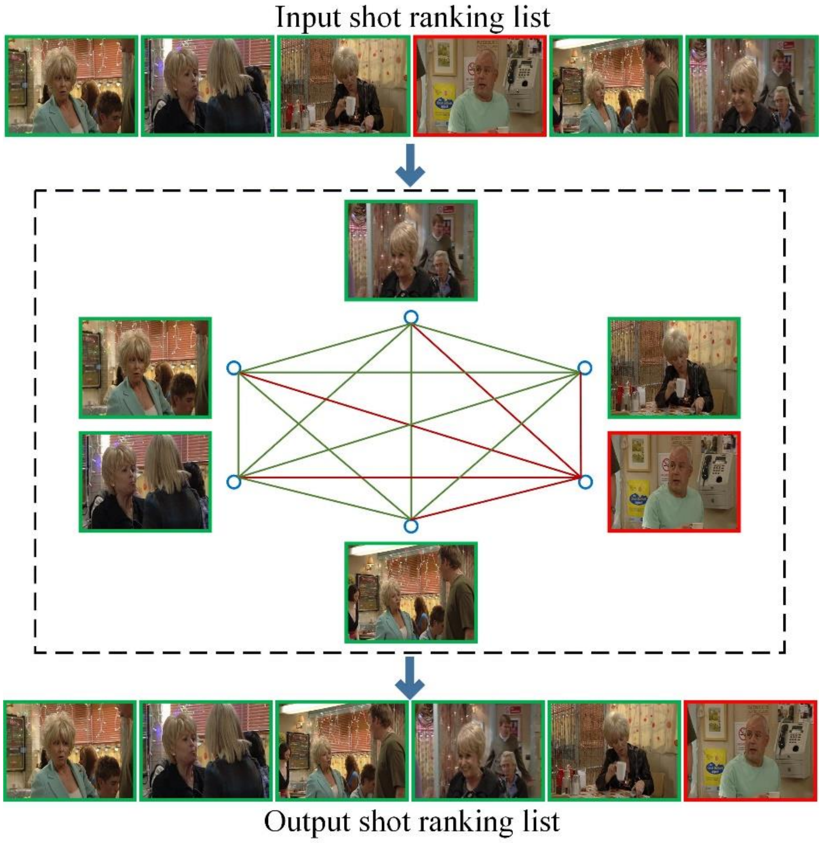

Figure 3: Our semi-supervised re-ranking algorithm. Green rectangles mean right results, while red ones mean wrong results. Green edges in the graph mean high similarity between shots, while red ones mean low similarity.

(1) Given the data matrix of 1000 top-ranked shots F and L, where F_i meant the DNN feature vector for location of a key frame and L_i meant the shot ID of vector F_i, $i \in \{1, 2, \ldots, n\}$ where $n > 1000$ meant there were n key frames from 1000 shots.

(2) Initialized the affinity matrix W with all zeros, and updated as following:

$$W_{i,j} = \frac{F_i \cdot F_j}{|F_i| \cdot |F_j|}, i, j \in \{1, 2, \ldots, n\}, i \neq j \tag{7}$$

(3) Generated the k-NN graph:

$$W_{i,j} = \begin{cases} W_{i,j}, & F_i \in kNN(F_j); \\ 0, & otherwise. \end{cases} \tag{8}$$

where $kNN(F_j)$ denoted the set of k-nearest neighbors of F_j.

(4) Constructed the matrix: $S = D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$, where D was a diagonal matrix with its *(i, i)*-element equal to the sum of the *i*-th row of W.

(5) Iterated $G_{t+1} = \alpha SG_t + (1-\alpha)Y$ until convergence, where G_t denoted the refined result in *t*-th round and we set $G_0 = Y$. α was a parameter in the range (0, 1), and Y was the final ranking list of key frames from 1000 top-ranked shots, and we set the score of each key frame as the same with its original shot.

4 Interactive Search

This year we used similar strategy as what we used in the interactive search task of INS 2016. For the sake of efficiency, we performed interactive search based on RUN2_E without result re-ranking stage. First, the user labeled the top-ranked results in the ranking lists of automatic search as positive or negative samples for each topic. In the final interactive run, the positive samples would rank at the top of the list, while the negative ones would be discarded. Next, the positive samples were used as expanded queries to conduct the location and person search. The number of expanded queries for each topic was up to 10 to guarantee the efficiency of interactive search. After getting the scores of expanded queries, the scores of expanded and original queries were merged together to effectively boost the search accuracy.

5 Conclusion

By participating in the instance search task in TRECVID 2017, we have the following conclusions: (1) Video examples are still helpful for accuracy improvement, which can provide more information to get higher results (see Table 1). (2) The automatic removal of “bad” faces is important to achieve accurate face recognition results, which causes severe confusion. (3) The fusion of location and person similarity is a key factor of the instance search. Because we have various cues for instance search, we should carefully balance and make full use of them to achieve ideal search results.

Acknowledgements

This work was supported by National Natural Science Foundation of China under Grants 61771025, 61371128 and 61532005. For the using of BBC EastEnders video or images snapshots, we thank for the programme material copyrighted by BBC.

References

[1] G. Awad, A. Butt, J. Fiscus, *et al.*, “TRECVID 2017: Evaluating Ad-hoc and Instance Video Search, Events Detection, Video Captioning and Hyperlinking”. Proceedings of TRECVID 2017, 2017.

[2] G. Awad, W. Kraaij, P. Over, *et al.*, “Instance search retrospective with focus on TRECVID”, *International Journal of Multimedia Information Retrieval,* vol. 6, no. 1, pp. 1-29, 2017.

[3] C.G.M. Snoek, K.E.A. van de Sande, O. de Rooij, *et al.*, “The MediaMill TRECVID 2008 Semantic Video Search Engine”. *TRECVID*, Maryland USA, November 17-18, 2008.

[4] K. Mikolajczyk, and C. Schmid, “Scale and Affine Invariant Interest Point Detectors”. *International Journal of Computer Vision (IJCV)*, vol. 60, no. 1, pp. 63-86, 2004.

[5] J. Matas, O. Chum, M. Urban, and T. Pajdla, “Robust wide baseline stereo from maximally stable extremal regions”. *British Machine Vision Conference (BMVC)*, pp. 384-393, 2002.

[6] D. G. Lowe, “Distinctive Image Features from Scale-invariant Keypoints”. *International Journal of Computer Vision (IJCV)*, vol. 60, no.2, pp. 91-110, 2004.

[7] K. Mikolajczyk and C. Schmid, “A Performance Evaluation of Local Descriptors”. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 27, no.10, pp. 1615-1630, 2004.

[8] K. Simonyan, A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition”, *International Conference on Learning Representations (ICLR)*, 2015.

[9] C. Szegedy, W. Liu, Y. Jia, *et al.*, “Going deeper with convolutions”, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1-9, 2015.

[10] K. He, X. Zhang, S. Ren, *et al.*, “Deep residual learning for image recognition”, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770-778, 2016.

[11] B. Zhou, A. Khosla, A. Lapedriza, *et al.*, “Places: An image database for deep scene understanding”, arXiv preprint arXiv:1610.02055, 2016.

[12] K. Zhang and Z. Zhang and Z. Li, *et al.*, “Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks”. *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499-1503, 2016.

[13] O. M. Parkhi, A. Vedaldi, and A. Zisserman. “Deep Face Recognition”. *British Machine Vision Conference (BMVC)*, pp. 41.1-41.12, 2015.

[14] C.-C. Chang and C.-J. Lin, “LIBSVM : a library for support vector machines”. *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, pp. 1-27, 2011.

[15] Y. Peng, X. Huang, J. Qi, *et al.*, "PKU-ICST at TRECVID 2016: Instance Search Task", *TRECVID*, Gaithersburg, MD, USA, Nov. 14-16, 2016.