WHU-NERCMS at TRECVID2014:Instance Search Task

Mang Ye, Bingyue Huang, Lei Yao, Jian Qin, Jian Guan, Xiao Wang, Bo Luo, Zheng Wang, Dongjing Liu, Zhuosheng Zhang, Su Mao, Chao Liang*

National Engineering Research Center for Multimedia Software, School of Computer, Wuhan University, Wuhan, 430072, China
{ cliang@whu.edu.cn}

Abstract. This paper introduces our work at the automatic instance search task of TRECVID 2014. Our work is divided into two parts: First part is object retrieval based on BOW. Specially, we extract feature histogram of frames through general BoW. We adopt similarity measure method to compare the probe and gallery shots, then we obtain the initial ranking results; Second, several optimization strategies are adopted to improve the initial results.

1 Introduction

In TRECVID 2014 [1], we participate in the automatic instance search task(INS), results are submitted as shown in Table 1. MAP is the evaluation index [2]. And description shows the sequence numbers of candidate query images we adopted. For example, "image examples #1-3" represents that the 1st to 3rd images of topic are selected as query images in our work. The framework of our team is shown in Fig 1. Our INS task can be divided into two parts: object retrieval based on BOW and optimization phase. The first part of our INS work can be summarized as feature extraction, codebook training, object saliency and similarity measure. Specially, in feature extraction, we first extract the keyframes of videos, and then employ SparseSIFT [3] feature to represent the local geometric relationship of keypoints extracted from above keyframes. In the codebook training phrase, a subset consisting of 20 million features is selected randomly from the whole features as training data, and a 1M codebook is therefore obtained after clustering these datas. To compute object salience, considering the availability of contextual region, the "Stare Model" [4] is utilized to weight query images. Thereafter, we adopt the binaryzation and tf weighting methods to compute the similarity of two images. In the final optimization part, we use the face filter and color filter to optimize the initial results.

* Corresponding author.

Table 1: INS results and descriptions for each run. As we know, each topic include four candidate query images, therefor description shows the sequence numbers of query images we adopted. For example, "image examples #1-3" represents that the 1st to 3rd images of topic are selected as query images in our work.

Method	MAP	Description
F_NO_NERCMS_1	0.059	image examples #1-3 only
F_NO_NERCMS_2	0.057	image examples #1-3 only
F_NO_NERCMS_3	0.055	image examples #1-2 only
F_NO_NERCMS_4	0.042	image examples #1 only

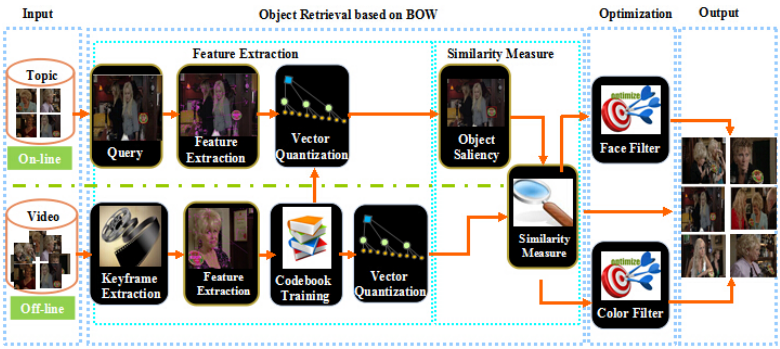

Fig. 1: The framework of our team

2 OBJECT RETRIEVAL BASED ON BAG OF WORDS

2.1 Feature Extraction

This section presents our feature extraction. Firstly, we extract video keyframes as the pretreatment. In our implementation, the middle frame of shot which includes frames less than 50 is extracted as the keyframe. When a shot includes 50 frames or more, we select 2 frames at the location of 1/3 and 2/3 of the keyframes. Then, in feature extraction, we employed SparseSIFT feature to express the picture, since the local feature based on scale-invariant key point has already been shown to be effective in object retrieval. As have been proved in many previous works, RootSIFT [5] is more efficient than the original SIFT descriptors. Therefore, RootSIFT is adopted in all of our steps. The result of SparseSIFT feature is shown in Fig 2. The left picture is the raw frame and the right picture shows the SparseSIFT feature, we can see many key points are extracted form region-of-interest(ROI). The dimension of resulting feature is 128. Considering the balance and calculation convenience, we take no more than 1000 SparseSIFT features per frame.

Fig. 2: The results of SparseSIFT

2.2 Codebook Training

As we know, codebook training is important to instance search. And many early research works show that the retrieval precision can benefit from larger size of visual codebook. We adopted Approximate K-Means (AKM) [6] algorithm to train the codebook. AKM uses randomized KD trees to perform approximate nearest neighbor search, which makes it possible to train large codebook in reasonable time.

We extracted about 1.2 billion features from about 0.7 million key frames. Considering the hardware configuration and the requirement of time complexity, a subset consisting of 20 million features is selected randomly from the 1.2 billion features as training data. After clustering these datas, we finally get a 1M codebook.

2.3 Object Saliency

This section presents a "Stare Model" to weight every frames. Traditionally, features raised from region-of-interest (ROI) of query image are reserved for searching while features raised from background considered as noise. However, the background area may provide important information in some cases, especially when the target is a tiny object, as shown in Fig 3. In this case, background can provide relative information of target, which is important for our searching. The function used by "Stare Model" is

$$w(x) = \begin{cases} 1 & \text{if x} \in \text{mask} \\ \frac{2}{e^{kx/diag}+1} & \text{otherwise} \end{cases} \tag{1}$$

where $w(x)$ is the weight of a pixel, $diag$ indicates the length of diagonal axis of the query image, x is the minimum distance between the point and the mask region, k is a parameter of weight adjustment. In our experiments we choose $k = 15$. If x is belong to mask region, its weight is 1. Otherwise is damping according to the rule indicated by Eq.(1). With the "stare model", we are able to use the context when the instance is small to improve the recall rate.

Fig. 3: The tiny decoration in the red box is the target of topic 9108. When searching the tiny decoration, the feature of the wall and louver can provide important assistant information.

2.4 Similarity Measure

With the trained 1M codebook, we can quantize each 128 dimensional SparseSIFT descriptor of keyframes into one of the codes ranging from 1 to 1000000. For the query frames, we adopt soft matching method to quantize SparseSIFT descriptors of query frames. The parameter of soft matching is 3, which means one SIFT point can be quantized into 3 different codes in codebook. In similarity measure, we firstly convert the feature vectors to binaryzation vectors, which can reduce the computational complexity of similarity measure. Furthermore we implemented term frequency(tf) weighting to the binaryzation vectors. The similarity measure function is

$$Sim(q,i) = Coord(q,i) \times Norm(q) \times Norm(i) \times \sum_{x \in i} tf(x) \tag{2}$$

where q is a query image, and i denote image of gallery. $Coord(q,i)$ is a score factor based on how many query terms are matched in the specified image, and the value of $Coord(q,i)$ is the quantity of matching key features between query and gallery image. $Norm(q)$ is a normalizing factor to balance the inequality caused by non-uniform quantity of the feature points of query image, and $Norm(i)$ is a normalizing factor to balance the inequality caused by non-uniform quantity of the feature points of gallery image.

3 Optimization

After above steps, we get the initial results without any optimization. However, a lot of prior knowledge can be beneficial to optimize the initial results in practice. For example, when the instance we search often appears with people, like the topic 9099 which include the police hat as target, we can use person detection method to filter the initial results. In the case of the topic include vehicles,

such as the topic 9118, the vehicle detect method can be adopted to optimize. Furthermore, some topics have bright colors, so we can use it to improve the precision of our search. For example, the target of topic 9114 is the red mailbox, so we can exclude the images which do not have red color of inial results. Fig 4 shows three examples of topic need to optimize. In the following subsections, we will discuss it specially.

(a) Topic 9099 (b) Topic 9114 (c) Topic 9118

Fig. 4: Three examples of topic need to optimize. (a) Topic 9099 includes the police hat as target which often appears with people. (b) The target vehicle logo of topic 9118 is a part of vehicle. (c) The target of topic 9114 is the red mailbox, we can use the red color to optimize initial results.

3.1 Face Filter

In this subsection, we introduce the face filter we adopted for optimization. When the goal of topic is a person or always appears with persons, this method is adopted to filter the images which do not include persons. The face filter use the Viola-Jones face detect algorithm [7], which extract the integral images to calculate the Haar-like features efficiently. The Viola-Jones algorithm uses Adabosst leaning algorithm to select features and train the classifiers. And the cascade classifier is applied to promote efficiency.

In our implementation, firstly, we use the Viola-Jones face detect algorithm to detect all the keyframes. When face is detected, label 1 to the keyframe. Otherwise, label 0 to it. Then we get a vector valued by 0 and 1. Secondly, we select the topic of which target is person or always appears with persons manually. At last, we can optimize the results of person relevant topics using the vector achieved at first step.

3.2 Color Filter

Color filter is discussed in this subsection. The goal of color filter is to eliminate the images do not contain the target instance color obviously. The basic idea of our color filter method can be summarized as follows: firstly, extracting the h

component of image in the HSV color model, and get the main color histogram of target Ht. Then, getting the area of image within the scope of color by doing reverse projection for each image through Ht; Thirdly, calculating the scope area $Area$. At last, if the $Area < threshold$, filter out the image.

In our implementation, considerate the efficiency, the above process is decomposed into offline and online processes. Offline part statistics the area of each color value for every pictures under query; Online part aims to determine the color range, and calculate the area for each color values of query images based on the offline part. When changing the target image or the parameters of threshold in the above final step, the advantage of these processes will be highlighted. The result of offline part process can be used repeatedly, we need only to do online part again. Furthermore, the offline part is time-consuming in entire process, while online part can be finished in a few minutes.

4 Results And Analysis

The four results of our INS system are shown in Fig 5. The average precision is 0.059, which is the best run in all submitted runs. And the Fig 6 shows the results compare with other teams. The dot represents our best run score, the line represents median score and the best score is represented by box. From the Fig 6, we found that there are 9 topics that have higher average precision, while the rest are not satisfactory.

Compared to our work at TRECVID 2013 [8], we have following conclusions:

- For the tiny target like logo searching, SparseSIFT is better than DenseSIFT. But for the person searching, DenseSIFT is more effective;
- Large scale codebook training brings a more precise searching results, but at the same time, the calculation is more complex;
- Our BoW model is still too rough, so the discrimination ability is not strong.

After analysing our results and comparing to other participants [9], we get some suggestions and experiences to guide future work:

- Adopt intra-shot clustering method to extract the key frames, it's more precise than our keyframe extraction method this time;
- Downsize the raw key frames before extracting features. For high resolution frames, the large amount of features put a heavy burden on computational complexity, and artifact on moving objects introduces a lot of noise too;
- Add the vehicle filter to optimize initial results, since many targets always appear with vehicles. So we can optimize the results by filter the frames which do not include vehicles.

Acknowledgement. Our work use programme material copyrighted by BBC. Thank for the great support to our work by professor Jun Chen of NERCMS, and thank to the following undergraduates who give some help in our work, they are Weicheng Zheng, Pei Xu, Rui Guo, Dian Chen, Mengmeng Xiao.

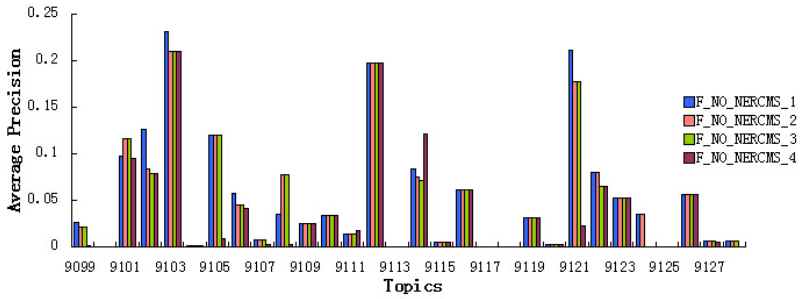

Fig. 5: Our NERCMS's results

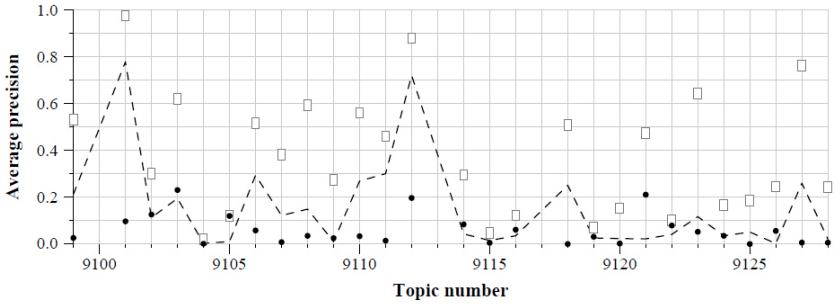

Fig. 6: The results compare with other teams

References

[1] Paul Over, George Awad, Martial Michel, et al.: TRECVID 2014 – An Overview of the Goals, Tasks, Data, Evaluation Mechanisms and Metrics. In: Proceedings of TRECVID 2014. (2014)

[2] Alan F. Smeaton, Paul Over, and Wessel Kraaij.: Evaluation campaigns and trecvid. In: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval(ACM). 321-330 (2006).

[3] Wang J G, Li J, Yau W Y, et al.: Boosting dense SIFT descriptors and shape contexts of face images for gender recognition. In: Computer Vision and Pattern Recognition Workshops (CVPRW). 96-102 (2010)

[4] Wei Zhang, Chun-Chet Tan, Shi-Ai Zhu, Ting Yao, Lei Pang, and Chong-Wah Ngo, Vireo@ trecvid 2012. (2012)

[5] Arandjelovic R, Zisserman A.: Three things everyone should know to improve object retrieval[C]. In: Computer Vision and Pattern Recognition (CVPR). 2911-2918 (2012)

[6] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman.: Object retrieval with large vocabularies and fast spatial matching. In: Computer Vision and Pattern Recognition (CVPR). 1-8 (2007)

[7] Paul Viola, Michael J. Jones.: Robust Real-Time Face Detection[J]. In: International Journal of Computer Vision(IJCV). 137-154. (2004)

[8] Yimin Wang, Mang Ye, Qingming Leng, Bingyue Huang,Zheng Wang, Yuanyuan Nan, Wenhua Fang, Chao Liang.: WHU-NERCMS at TRECVID2013:Instance Search Task. In: Participant Notebook Paper of TRECVID. (2013)

[9] Hongliang Baiy, Yuan Dongz, Shusheng Cenz, et al.: ORANGE LABS BEIJING(FTRDBJ) AT TRECVID 2013: INSTANCE SEARCH. In: Participant Notebook Paper of TRECVID. (2013)