IMP at TRECVID 2012

Tomohiro Sakata Nobuaki Matozaki Koichi Kise Masakazu Iwamura
Intelligent Media Processing Group,
Dept. of CSIS, Graduate School of Engineering,
Osaka Prefecture University
1-1 Gakuencho, Naka, Sakai, Osaka 599-8531, Japan
{sakata, matozaki}@m.cs.osakafu-u.ac.jp {kise, masa}@cs.osakafu-u.ac.jp

Abstract

We have participated in the Instance Search (INS) task. In order to improve the accuracy of retrieval, we have tried not the common approach based on the Bag-of-Features (BOF) representation but voting based on direct matching of local features, which can be thought of as the extreme of BOF representation taking all local features as individual visual words. The most serious problem of this simple approach is the computational cost. We have solved this problem by using a hash-based approximate nearest neighbor search. To improve further the accuracy, we have utilized the following two methods: a pseudo relevance feedback method (feedback), and enlargement of original query images. The former is to take additional local features from top ranked shots to expand the original query. The latter is to enlarge the original query image to obtain more local features. The runs we submitted are as follows:

- *F_X_NO_IMP.h_f_e1_4 : with feedback , original query*
- *F_X_NO_IMP.h_f_e2_2 : with feedback , original and 2 times enlarged queries*
- *F_X_NO_IMP.h_e2_1 : without feedback , original and 2 times enlarged queries*
- *F_X_NO_IMP.h_e3_3 : without feedback , original, 2, and 3 times enlarged queries*

These simple methods have allowed us surprisingly good results for some queries. In this note, we explain the proposed method as well as to disclose reasons of good results and remaining future work for improving the results.

1. Introduction

INS is the task of specific object recognition for the video data. In the specific object recognition, it is known that a better recognition rate can be obtained by using a larger number of visual words [1]. An extreme is to take all local features obtained from video frames in the database (DB) as different visual words. However this causes a serious problem of matching local features from the query image to those in the DB whose number is about 1.5 billion.

To solve this problem a de facto standard way is to employ the Bag-of-Features (BOF) representation [2] which limits the number of visual words by using the vector quantization. The accuracy of recognition, which is lowered by the BOF, is recovered by using some sophisticated classifiers and learning process.

In our approach, on the other hand, we come back to the simplest way — taking all local features as different visual words and represent each shot by using a vector with huge dimensions. This approach is equivalent to a simple voting method by matching local features.

Needless to say, this approach poses a problem of computational load. It is too time consuming to apply the exact nearest neighbor search for matching local features. In our approach, we solve this problem by using a hash-based approximate nearest neighbor search [3].

To further improve the accuracy, we employed the following two methods in addition to the local feature matching. One is a method called pseudo relevance feedback, which takes top ranked images by the first round retrieval to extract local features from them and employ these local features in addition to the ones from the original query image for the second round retrieval. With the expectation that the top ranked images often contain correct instance objects to be retrieved, additional shots can be retrieved by using their local features. The other is to enlarge the original query image to obtain more local features. This is to cope with some instance objects whose size is too small to be described by local features.

Even with the above quite simple methods, we have achieved the mean average precision of 16.9%. For some queries, our results are the best among others. This means that for these queries direct matching of local features has advantages as compared to commonly used methods based on the BOF representation. In this note, we discuss pros and cons of our simple method with analysis of retrieval results.

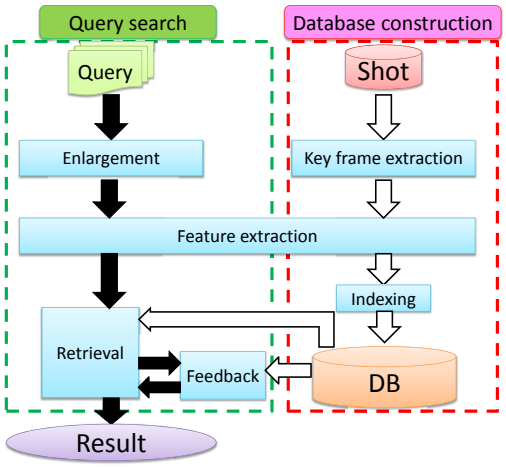


Figure 1. Processing flow of the proposed method.

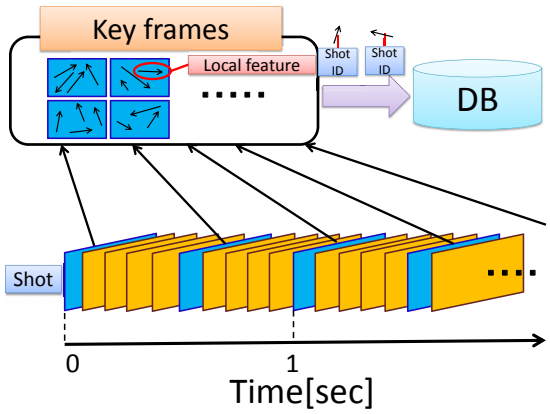


Figure 2. Key frame and feature extraction.

2. Proposed Method

Figure 1 shows the processing flow of our method. For the database construction (the right flow of Fig. 1) we first select key frames from which local features are extracted. Then these local features are stored into the database with their shot ID. For the query processing (the left flow of Fig. 1), we first enlarge the input query image if necessary. Then local features are extracted from the original and enlarged query images. The retrieval is done by voting for shot IDs based upon matching of local features from the query to those in the DB.

In the following each step of processing is described.

2.1. Database construction

2.1.1 Key frame extraction

The process of extracting key frames as well as local features from them is illustrated in Fig. 2. Since many frames are quite similar with one another, it is not necessary for us to extract local features from each of them. To reduce the number of images for feature extraction, we simply take a frame image in every 0.5[s].

2.1.2 Feature extraction

As local features, we employ the Opponent SIFT feature [4]. We utilize the Harris Laplace detector [5] which enables us to obtain local regions that are robust to scale changes. From the extracted local regions, opponent SIFT features which contain color information are extracted. First, the image of a local region is converted from the RGB color space to the opponent color space as follows:

$$\begin{pmatrix} O_1 \\ O_2 \\ O_3 \end{pmatrix} = \begin{pmatrix} \frac{R-G}{\sqrt{2}} \\ \frac{R+G-2B}{\sqrt{6}} \\ \frac{R+G+B}{\sqrt{3}} \end{pmatrix} \quad (1)$$

The channel O_3 is the same as the intensity of HSV color space, the channel O_1 has red and green information, and the channel O_2 has yellow and blue information. Then, features are extracted from each channel by the SIFT descriptor, resulting in 384 (= 128 dimensions × 3 channels) dimensional features. We reduce their dimension from 384 to 60 by using the principal component analysis.

2.1.3 Indexing [3]

Let $\boldsymbol{x} = (x_1, x_2, \cdots, x_{d'})$ be a d'-dimensional local feature extracted from key frames. A binary vector $\boldsymbol{u} = (u_1, u_2, \cdots, u_d)$ is defined using a truncated feature vector, i.e., the feature vector with the first $d(< d')$ dimensions of $\boldsymbol{x}$ as follows:

$$u_i = \begin{cases} 1 \text{ if } x_i - \theta_i \geqq 0, (1 \leq i \leq d) \\ 0 \text{ otherwise,} \end{cases} \quad (2)$$

where θ_i is the median of the value x_i of feature vectors in the database. The hash value of a feature vector $\boldsymbol{x}$ is calculated as

$$H_{\text{index}} = \left(\sum_{i=1}^{d} u_i 2^{i-1} \right) \bmod H_{\text{size}} \quad (3)$$

where H_{size} is the size of the hash table. A scalar quantized version of $\boldsymbol{x}$ and its shot ID are recorded in the hash table.

Collisions of hashing are resolved by using the chain method. Too many collisions reduce the performance of retrieval, since feature vectors that cause the collisions are similar and thus with less discriminative. For solving this problem, we employ the upper limit of the number of shot IDs in each chain. If they exceed the limit, the chain is deleted from the hash table and no further record is allowed.

2.2. Query search

2.2.1 Enlargement of the query image

It is sometimes the case that we are not able to extract an enough number of local features from the original query

image. This is because of either (1) the lack of resolution, or (2) small object regions in the query image. To cope with this problem we enlarge the original query image two or three times as large as the original by using a method of interpolation based on Lanczos resampling. Then local features are likewise extracted from both the original and enlarged query images.

2.2.2 Retrieval [3]

Let $\boldsymbol{q} = (q_1, ..., q_d)$ be the truncated query feature vector. Candidate feature vectors $\boldsymbol{X}$ for matching are obtained from the hash table using the hash index of $\boldsymbol{q}$ calculated by Eq. (3). In order to cope with variations of local features, we employ multi-probing as follows. For a dimension i of the query vector which satisfies $|q_i - \theta_i| \leq e$ where e is a tolerance, we employ not only the original bit vector but a flipped bit vector with the value $1 - u_i$. We apply this multi-probing process up to b bits. This means that at maximum 2^b bit vectors are employed for finding the candidates $\boldsymbol{X}$.

The feature vectors among the first k nearest neighbors (NNs) are obtained by calculating the Euclidean distance between the query feature vector $\boldsymbol{q}$ and each feature vector in $\boldsymbol{X}$. Then for each of these k-NNs, we cast a vote to its corresponding shot ID. We employ not 1-NN but k-NNs because we have multiple correct shots for one query image.

The weights of votes are not equal for all shots. In general, shots with more local features tend to have more votes. Another point we need to consider is that k-th NN is less important than the first NN; the importance could be a function of k. These points are reflected to a simple weighting of votes: $(0.95)^{k-1}/\sqrt{C_s}$ for k-th NN where C_s is the number of local features extracted from a shot s.

In addition, we give different weights for local features from foreground (the object region) and those from background. To be precise, local features from the foreground have a weight n times as large as those from the background.

We apply the above weighted voting for all features from the query and finally rank shots according to the sum of weights.

2.2.3 Feedback

One of the problems to be addressed for improving the accuracy of retrieval is that local features extracted from the query image is far different from those of shots including target objects. This happens because, for example, of different sizes in the image, different lighting conditions and camera angles.

A solution is to generate possible variations by simulating these effects and employ them as well for the retrieval. Another, a simpler way is to utilize shots in the DB. We expect that some shots include object images whose local features are different from those of the query image. If such shots also include local features that are similar enough from those of the query image, by matching to similar ones we are able to access different local features, which enable us to retrieve additional shots. This process is called "relevance feedback".

A serous drawback of this strategy is that there is no way to select local features of the target object. A simple solution for this problem is called "pseudo relevance feedback", which assumes all local features extracted from the top r-ranked shots are from the target object. This is obviously incorrect, since, even if a shot includes the target object, not all local features are from it. However, this simple strategy often works well because local features extracted from other objects match to irrelevant shots and thus incorrect votes are distributed to different incorrect shots. This process is simply called "feedback" in this note.

When we apply the feedback, we should consider that the local features obtained by the feedback (expanded local features) are extracted from the shots in the DB. This means that all expanded local features have exactly the same local features in the DB. To avoid this match, we cast a vote not to the first NN. In addition, expanded local features are less reliable as compared to the original. Thus for the second round retrieval, which employ both the original and expanded local features, we put a smaller weight $m(< 1)$ to expanded local features.

Table 1. Conditions of each run.

Run	feedback	original query	enlarged query	
			2 times	3 times
IMP.h_f_e1_4	✓	✓		
IMP.h_f_e2_2	✓	✓	✓	
IMP.h_e2_1		✓	✓	
IMP.h_e3_3		✓	✓	✓

3. Evaluation

3.1. Conditions

We submitted the following four runs.

- F_X_NO_IMP.h_f_e1_4
- F_X_NO_IMP.h_f_e2_2
- F_X_NO_IMP.h_e2_1
- F_X_NO_IMP.h_e3_3

Table 1 shows differences of runs in terms of feedback and enlargement. These runs share the same values of parameters as shown in Table 2.

3.2. Results

Table 3 summarizes the MAP(mean average precision) of each run and Figs. 8 and 9 show their details where the

Table 2. Shared parameter values.

No. of dimensions for hashing	d	32
No. of NNs for Voting	k	20
Voting weight for features from object regions	n	5
Upper limit of bit flips	b	10
No. of shots for feedback	r	10
Weight for expanded local features	m	0.1

Table 3. Mean average precision of each run.

Run	MAP[%]
F_X_NO_IMP.h_f_e1_4	16.9
F_X_NO_IMP.h_f_e2_2	16.9
F_X_NO_IMP.h_e2_1	16.5
F_X_NO_IMP.h_e3_3	15.7

former is with feedback and the latter is without it. From these results, it can be said that the feedback allows us to improve the MAP. On the other hand, the enlargement is effective only for the case without feedback [1]. As shown in Figs. 8 and 9 our method achieved the best results for some queries.

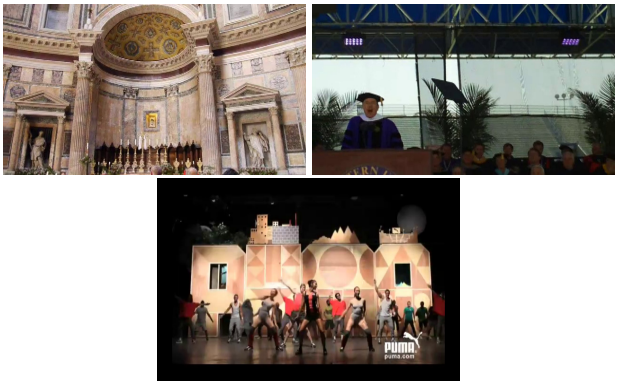
Figure 3 shows example queries with higher and lower average precisions (APs).

As shown in Fig. 3(a), queries with rich textures and/or less background were with high APs. Even if the object is small like the query of "PUMA logo" at the bottom of Fig. 3(a) we can also obtain a high AP if the similar background is always with the object.

For queries in Fig. 3(b) we were not able to obtain high APs. First, for queries with small objects such as the Mercedes star, there are many correct shots which include different backgrounds. Local features extracted from these backgrounds disturbed the retrieval. Second, for queries such as the Stonehenge, local features are from general parts such as rock and grass and thus less discriminative. This easily led incorrect matches to shots with general parts. In order to cope with this problem, it is required to consider more global structure among local regions, by, for example, taking into account the relative positions of local regions. Finally, for queries with different patterns and colors such as the MacDonald's arches, local features from these different parts are quite dissimilar and thus prevent us from retrieving the correct shots. In order to solve this problem, it is necessary to put a focus on a specific aspect of the object: in this case the outer shape of the object.

In order to know more details about the results, we investigated the accuracy of matching for local features from (1) the object, (2) the background, and (3) by the feedback, with the case of F_X_NO_IMP.h_f_e1_4. As Fig. 4 shows, the feedback is generally effective to improve the accuracy

[1]As compared to the run without feedback and enlargement, we have confirmed the effectiveness of enlargement.



(a)Queries with high APs

(b)Queries with low APs

Figure 3. Examples of Queries.

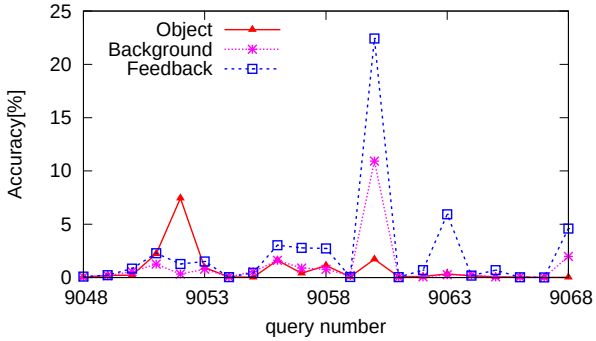


Figure 4. Accuracy of matching.

of matching. Except for some cases, queries with higher accuracy for local features from either the object or the background can achieve a higher accuracy for the feedback as well. On the other hand, we were not able to improve the results if the accuracies of local features for the object and/or the background were too low. In this case irrelevant local features were mostly employed for the feedback.

For the case of query 9052, which includes the subway logo, the feedback was not effective. This is because the subway logo was with a variety of backgrounds as shown

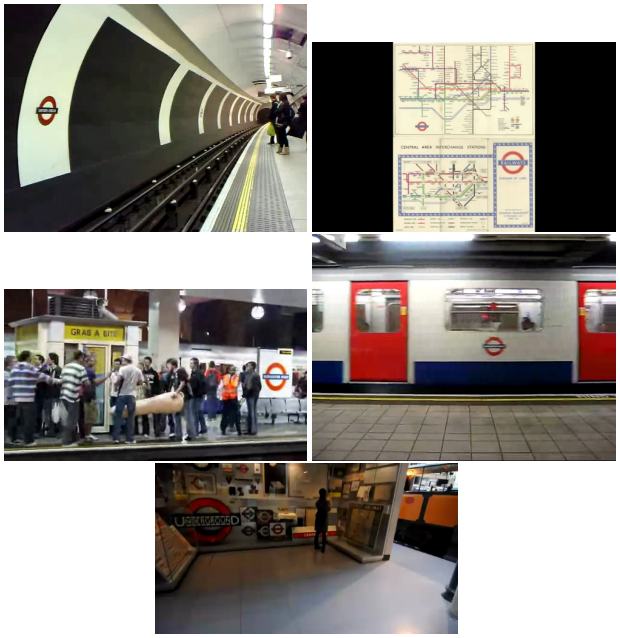


Figure 5. Correctly retrieved shots with the subway logo.

in Fig. 5. The shots shown in this figure were all correctly retrieved since the local features from the object were discriminative enough.

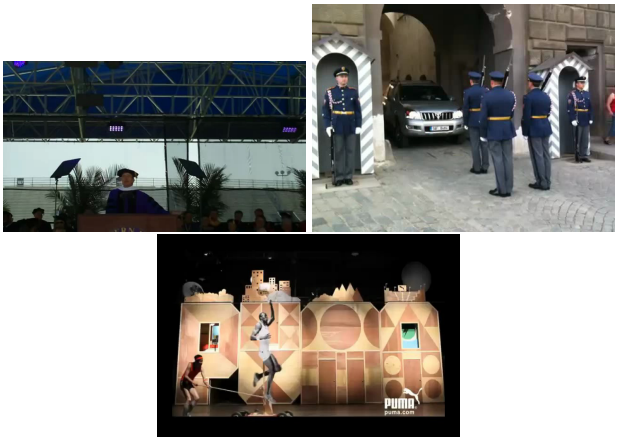
For the case of queries 9060 (person), 9063 (Prague castle), and 9068 (Puma logo), the feedback was effective. Some example shots that were and were not retrieved by methods with and without feedback are shown in Fig. 6.

Shots retrieved correctly without the feedback were often shared some parts of background with the query as shown in Fig. 3(a).

For the shots that were correctly retrieved by only using the feedback, frame images such as shown in Fig. 6(b) were quite different from the query image. However shots were often extracted from the same video sequence from which the query was also extracted. This means that at some point of a shot, frame images that contain similar local features are included. This allows us to retrieve such shots, from which different local features were further employed for the second round retrieval.

We also had some shots that were not retrieved even with the feedback. Figure 6 (c) shows such shots, which do not share any backgrounds with the query image. In order to cope with this problem, we need to have a similarity measure that is sensitive to the quality of matching for giving a higher weight to the part with the exact match.

Finally, in order to know the effect of weighting, we investigated the relationship between the number of local features extracted from a shot and its rank down to 50. Note that if the number of local features in a shot is small, we put a larger weight for each local feature. The results are shown in Fig. 7. As shown in this figure, incorrect shots with smaller number of local features tend to be ranked higher.



(a) Shots correctly retrieved by the methods with and without the feedback.

(b) Shots correctly retrieved only by the method with the feedback.

(c) Shots that were not retrieved by the methods with and without feedback.

Figure 6. Example shots in the database.

This is a negative effect of normalization using the number of local features. This problem can be solved by a different normalization scheme such as “pivoted normalization” [6]

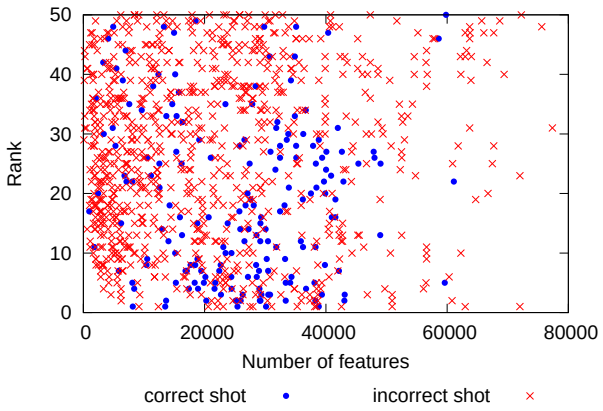


Figure 7. No. of local features and ranking of shots.

proposed in the field of IR.

4. Conclusion

We have participated in the Instance Search Task by using the simple direct matching of local features. The pseudo relevance feedback has proven to be effective to boost the accuracy of retrieval. We have analyzed the results and found that the proposed method is effective to find instance objects under the condition that textures on the object and/or the accompanying background are discriminative and stable enough. The feedback helps us to find shots that do not contain local features similar to those from the query image on condition that these shots contain some local features similar to the shots retrieved at the first round. This often holds since query images and correct shots tend to be extracted from the same sequence of video data.

The future work is to employ different features and strategies of matching for addressing the problems we encountered in our runs.

References

[1] D. Nistér and H. Stewénius, "Scalable Recognition with a Vocabulary Tree," pp. 2161–2168, Jun. 2006.

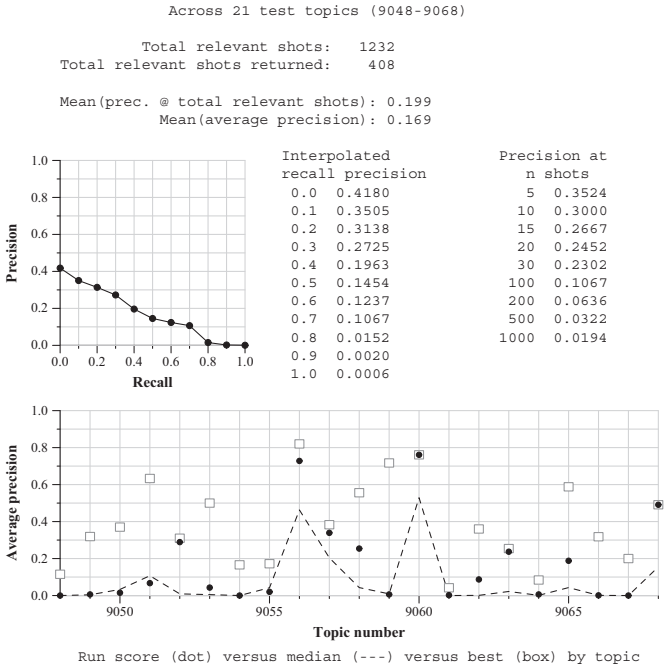
[2] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *In Workshop on Statistical Learning in Computer Vision, ECCV*, 2004, pp. 1–22.

[3] K. Kise, K. Noguchi, and M. Iwamura, "Robust and efficient recognition of low-quality images by cascaded recognizers with massive local features," *Proceedings of the 1st International Workshop on Emergent Issues in Large Amount of Visual Data (WS-LAVD2009)*, pp. 2125–2132, Oct. 2009.

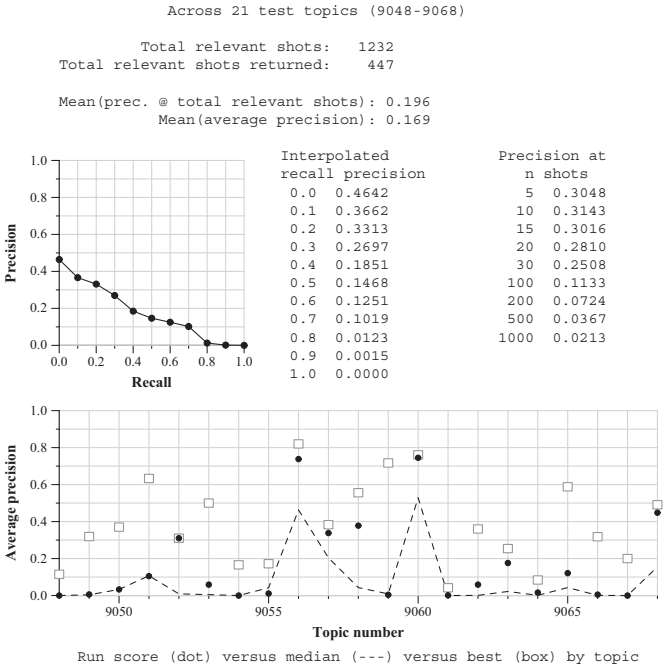
[4] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek, "Color Descriptors for Object Category Recognition," in *European Conference on Color in Graphics, Imaging and Vision*, 2008, pp. 378–381. [Online]. Available: http://www.science.uva.nl/research/publications/2008/vandeSandeECCGIV2008

[5] K. Mikolajczyk and C. Schmid, "Scale & Affine Invariant Interest Point Detectors," *Int. J. Comput. Vision*, vol. 60, pp. 63–86, October 2004. [Online]. Available: http://dl.acm.org/citation.cfm?id=990376.990402

[6] A. Singhal, C. Buckley, M. Mitra, and A. Mitra, "Pivoted document length normalization," in *Proc. SIGIR*. ACM Press, 1996, pp. 21–29.

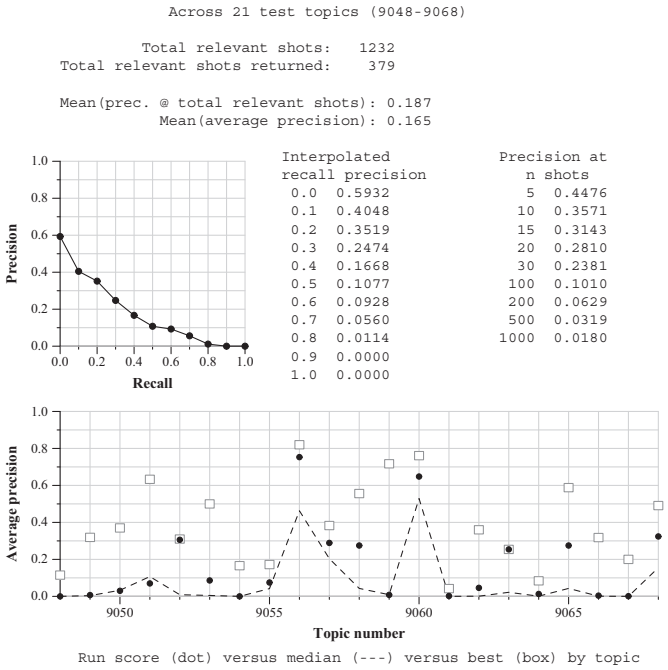


(a) IMP.h_f_e1

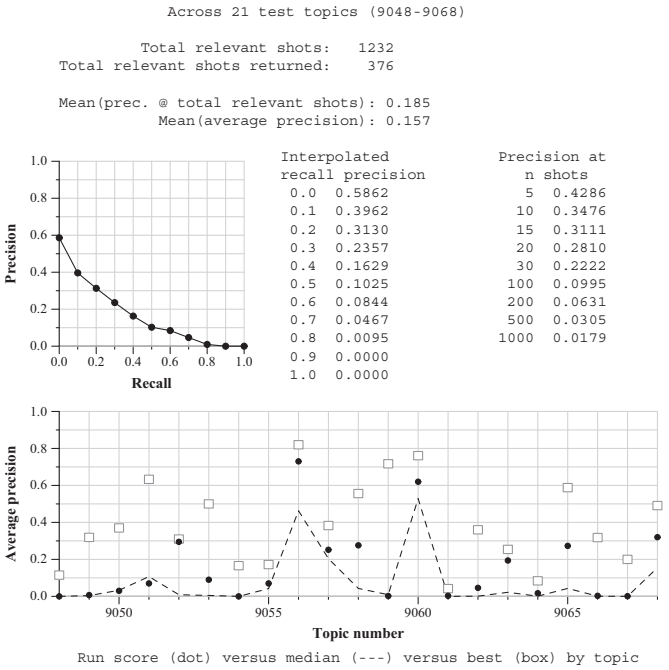


(b) IMP.h_f_e2

Figure 8. Results with the feedback.



(c) IMP.h_e2



(d) IMP.h_e3

Figure 9. Results without the feedback.