Osaka Prefecture University at TRECVID 2011

Tomohiro Sakata† Nobuaki Matozaki‡ Koichi Kise† Masakazu Iwamura†
†Dept. of CSIS, Graduate School of Engineering
‡Dept. of CSIS, 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 Content-based Copy Detection task. Our purpose for participation is to examine the performance of our image recognition method in the video retrieval. Our image recognition method utilizes local features directly, and achieves the high-speed processing, and high accuracy. We submitted 4 runs as follows.

- *IMP.m.balanced.Uvote : uniform vote*
- *IMP.m.balanced.Wvote : weighted vote*
- *IMP.m.nofa.Uvote : uniform vote*
- *IMP.m.nofa.Wvote : weighted vote*

The deference between Uvote and Wvote is whether votes are weighted or not. As a result, the image recognition works better than we expected even only with a slight modification for video retrieval. However further improvements are required for achieving satisfactory results as compared to the state-of-the-art methods.

1. Introduction

We have studied large-scale image recognition and have proposed an efficient method based on cascaded recognizes with approximate nearest neighbor matching. Our purpose of participation in TRECVID 2011 is not to create a new framework for the video retrieval but to examine the performance of our image recognition method in the field of video retrieval. In this year, we participated in the Content-based Copy Detection (CCD) task and INstance Search (INS). However we unfortunately could not submit the INS result, since we have a mistake in the database of INS. Therefore, we explain only CCD.

Our method of image recognition does not utilize the Bag-of-feature representation[1] which is generally used for data reduction. It employs a hash table for reducing the processing time. This method achieved high accuracy in the specific object recognition[2][3]. We call it the hash-based method in this paper.

The hash-based method utilizes SIFT feature[4] with PCA for dimensionality reduction. Since the reference data is huge, two frames per second were sampled and features were extracted from them. The retrieval is divided into two steps. At the first step, the reference video which corresponds to the query video is retrieved. The hash-based method is utilized in this step. At the second step, the corresponding part between the reference video and query video is retrieved. ANNS[5] is utilized in this step. As a result, we have achieved NDCR close to the median, though F1 values are lower than their median. The low F1 values were mainly caused by the second. On the other hand, the moderate values of NDCR are due to the performance of the hash-based method. In total, the overall performance of our method is not satisfactory, thus further improvements are required.

This paper is organized as follows. In Section 2, we describe the hash-based method. Its application to the copy detection and evaluation results of CCD are explained in Section 3. Finally, the conclusion is in Section 4.

2. Hash-based method

First, we explain the object recognition method based on a hash table, or the hash-based method, which is the base of our video retrieval method. In the hash-based method, memory reduction and fast recognition are achieved by scalar quantization and cascaded recognizers. Let us start with the explanation of storage of feature vectors in the database.

2.1. Storage of image data in the database

Let $\boldsymbol{x} = (x_1, x_2, \cdots, x_{d'})$ be a d'-dimensional vector extracted from reference images by applying SIFT and PCA. A binary vector $\boldsymbol{u} = (u_1, u_2, \cdots, u_d)$ is defined using the first $d(< d^{'})$ dimensions of $\boldsymbol{x}$ as follows:

$$u_j = \begin{cases} 1 \text{ if } x_j - \theta_j \geqq 0, \\ 0 \text{ otherwise}, \end{cases} \quad (1)$$

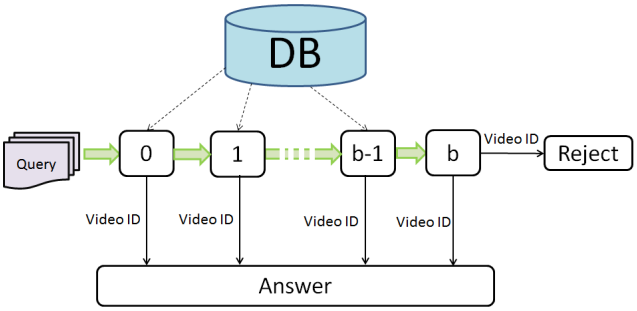

Figure 1. Cascaded recognizers

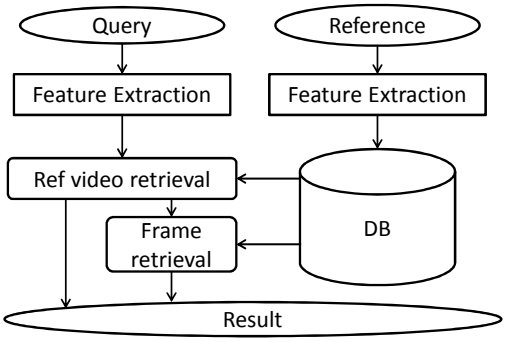

Figure 2. Overview of CCD

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

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

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

Collisions 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 length of chains. If they exceed the limit, the chain is deleted from the hash table and no further record is allowed.

We apply the above processing to the feature vectors extracted from all reference images, for the storage of image data.

2.2. Retrieval by the cascaded recognizers

The architecture of our hash-based recognition method is shown in Fig. 1. In this figure, the numbers $0, \cdots, b$ in squares correspond to a recognizer with a different level of approximation. The smaller the number is, the more the recognition is approximated. At each step, the recognizer matches each feature vector extracted from the query with those stored in the database by using approximate nearest neighbor search. The recognizer casts a vote to the image from which the matched feature vector is extracted. If the conditions described below are satisfied, the application of cascaded recognizers is terminated at this point and the image which has the maximum votes is regarded as the recognition result. Otherwise, the processing proceeds to the next step by using the next recognizer which has less approximation for matching. Thanks to the properties of "monotonicity" and "difference accessibility"[3] we can guarantee that the efficiency of processing are not deteriorated even in the worst case, in which the recognition result is obtained at the final step. Normally we can obtain about 10 times speed-up for recognition.

The conditions of terminating the recognition process are as follows. Let V_1 and V_2 be the numbers of votes for the first and second best images. If $V_1 > t$ and $rV_1 > V_2$ are satisfied, the recognition process is terminated and V_1 is outputted, where t and r are parameters as a result.

2.3. Recognizer

The recognizer at each step of the cascade is defined as follows. Let $\boldsymbol{q} = (q_1, ..., q_d)$ be a query feature vector. Candidate feature vectors to be matched are retrieved from the hash table using its hash index calculated by Eq.(2). However, since the values of bit vectors are not stable due to different imaging conditions, the query may miss its nearest feature vector in the database. In order to cope with this problem, we apply a kind of query expansion as follows. if $|q_j - \theta_j| \leq e$ is satisfied where e is a tolerance, we utilize not only the bit vector with u_j but also $1 - u_j$. In other words, we search both possibilities by flipping the bit if the value q_j is close enough to the threshold θ_j for making bit vectors.

Note that unlimited application of this process deteriorates the efficiency due to the combinatorial explosion. Thus we limit its application to the maximum number b, which corresponds to the number of steps of the cascade. At the first step, no bit flip is applied to obtain the recognition result. At the next step, one bit flip is applied, and its value is increased up to b at the final step. The dimension to which the bit flip is applied is selected by the value of difference $|q_j - \theta_j|$. The flip is applied to the dimensions with smaller difference up to the current limit.

3. Content-based Copy Detection

In this section, we explain the CCD method and evaluation results.

3.1. Method for CCD

An overview of our method for CCD is shown in Fig. 2. The method for CCD is divided into three steps as follows.

1. Feature extraction and DB construction

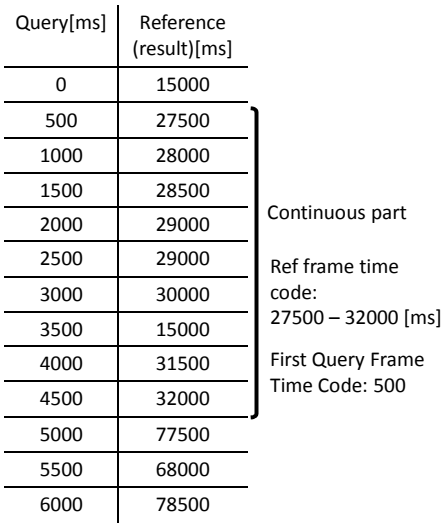

Figure 3. Example of frame retrieval

2. Reference video retrieval
3. Frame retrieval

We explain these steps briefly in the following.

3.1.1 Feature Extraction and make DB

Since video data is huge, it is prohibitive to use all frames of videos. Therefore we use two frames per second, i.e., sampling frames in every 0.5 seconds. We extract the SIFT feature[4] from these frames. In spite of frame sampling, the number of feature vectors from reference data is over 1 billion, thus the 128 dimensional vector of SIFT is too large to use directly. Accordingly PCA is applied to these features to reduce the dimension from 128 to 36.

Two kinds of databases are prepared for processing. One is the database which contains all feature vectors with each video ID. This is used in reference video retrieval. Another is the database which contains feature vectors from a reference video with each frame ID. This is used in frame retrieval.

3.1.2 Reference video retrieval

In this part, the reference video corresponding to the query video is retrieved. We regard the video as the set of feature vectors from its frames. In this way, the hash-based method can be utilized in the video retrieval. The method does not use the information of the frame ID and sequence, and only the video IDs are used in this part. All features from one query video are utilized to vote and the reference video corresponding to the query is retrieved.

3.1.3 Frame retrieval

In this part, matched frames between reference and query videos are retrieved based on the result of video retrieval. For each frame sampled from the query, the corresponding frame of the reference video is retrieved by ANNS[5] which is an Approximate Nearest Neighbor Search method. The corresponding frames that keep the sequential order in the reference video have higher chance to be a correct match. Therefore we retrieve such parts with the constraints on the sequential order. In this paper, "sequential" means that corresponding frames satisfy the formula,

$$|f_{n+1} - f_n| \leq 1500, \tag{3}$$

where f_n is the position in time of the frame of the reference video which corresponds to the n-th query frame. The unit of time is millisecond. The tolerance of 1500 [ms] is set to cope with false alarms caused by similar consecutive frames.

Figure 3 shows the example of the sequential part. The sequential part is retrieved as follows. First, we focus on one frame i and the next frame $i+1$ in the query, and check whether their corresponding frames in the reference video are also sequential. If they are, we step forward to find the next frames $i+1$ and $i+2$ to extend the sequence. Otherwise, we terminate the process. The sequential part is retrieved by repeating this process. After the application of the above process, we further merge the extracted sequences by checking the time gaps among them; if the gap is small enough, two sequences are further merged. Finally the longest sequential part is outputted as the result.

3.2. Evaluation results

We submitted 4 runs as follows.

- IMP.m.balanced.Uvote : uniform vote
- IMP.m.balanced.Wvote : weighted vote
- IMP.m.nofa.Uvote : uniform vote
- IMP.m.nofa.Wvote : weighted vote

The deference between Uvote and Wvote is whether vote in the hash-based method is weighted or not. Since the evaluation results of Wvote were better than Uvote, we only show the result of Wvote here. Figures 4 and 5 show the evaluation results of our Wvote runs. We use the video only query, thus we use the information of the a+v creation script and duplicate the results.

The results show that F1 values are lower than the medians. This is because the frame retrieval is so simple and did not work very well. In addition, this may affect the result of NDCR. The NDCR values close to the medians, hence the hash-based method works well in video retrieval. However there is large room for the improvement. For example, in the framework of CCD, countermeasures for various video transformations can be used.

In NOFA profile, actual NDCR values are worse as compared to the minimal NDCR values. This is simply because

a presence of one false alarm raises the NDCR value by a step of 100. In order to further improve the actual NDCR values, we need to employ better condition for rejection.

Many misses occurred in T8 and T10. This is because these contain multiple transformations. In order to cope with this problem, we first need to handle each transformation more effectively.

The problem of processing time is raised by the cascaded recognizers, which in the case that query is video, the number of feature vectors is huge. Thus the cascaded recognizers required much longer time for rejecting the video at the last stage of the cascade. Queries terminated in mid-flow spent several milliseconds, but queries which is not terminated spent several tens of seconds. Therefore we need to make the rejection more efficient to improve the efficiency of the overall processing.

4. Conclusion

We have employed the hash-based method and the SIFT feature in CCD task. As a result, although the performance was not satisfactory, we found that the hash-based method gives us a good starting point to implement the method of copy detection. Future work includes the improvement of the frame detection step.

References

[1] 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.

[2] K. Kise, K. Noguchi, and M. Iwamura, "Memory Efficient Recognition of Specific Objects with Local Features," *Proc. of the 19th International Conference of Pattern Recognition (ICPR2008) WeAT3.1*, 2008.

[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, 2009.

[4] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," in *International Journal of Computer Vision, Vol. 60, No. 2*, 2004, pp. 91–110.

[5] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu, "An optimal algorithm for approximate nearest neighbor searching in fixed dimensions," in *ACM-SIAM SYMPOSIUM ON DISCRETE ALGORITHMS*, 1994, pp. 573–582.

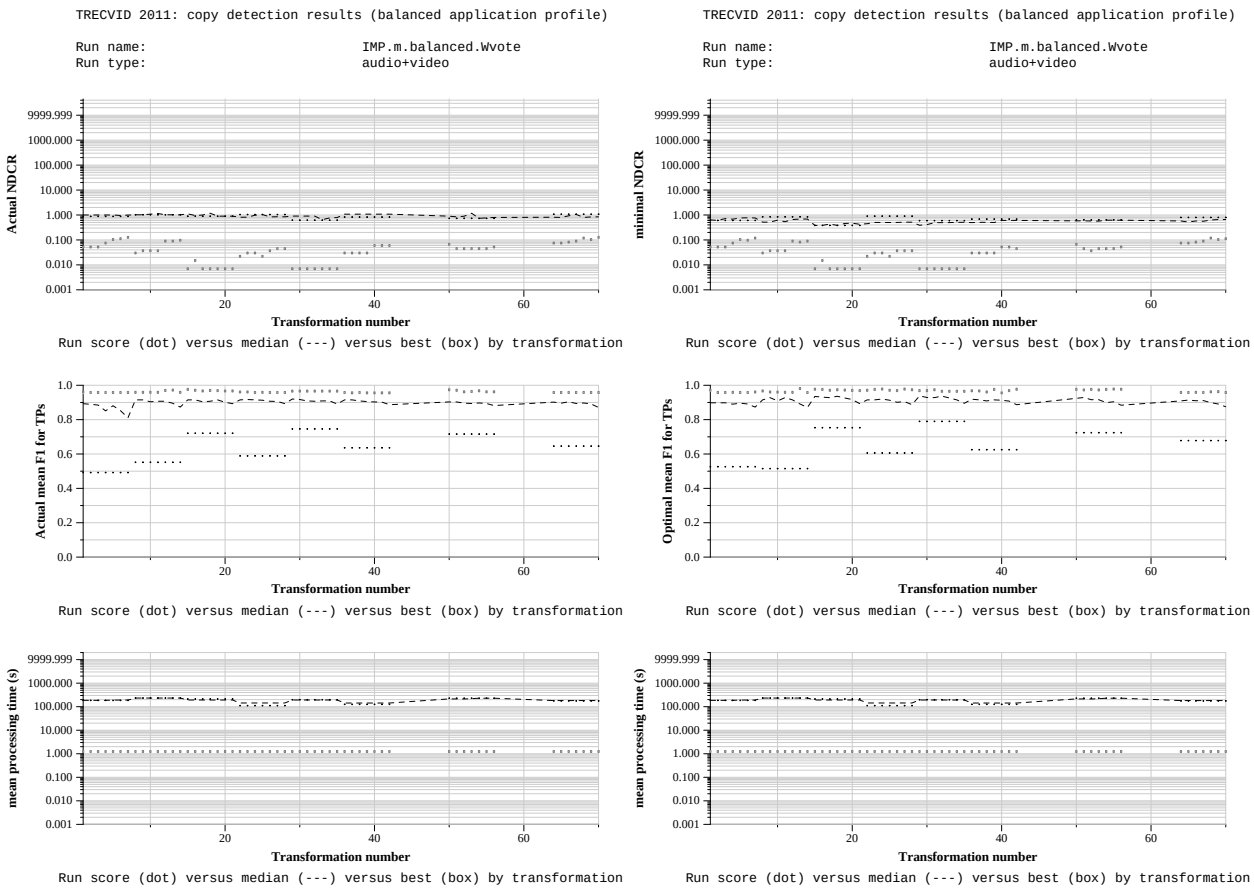

Figure 4. Actual(left) and optimal(right) Results of IMP.m.baranced.Wvote

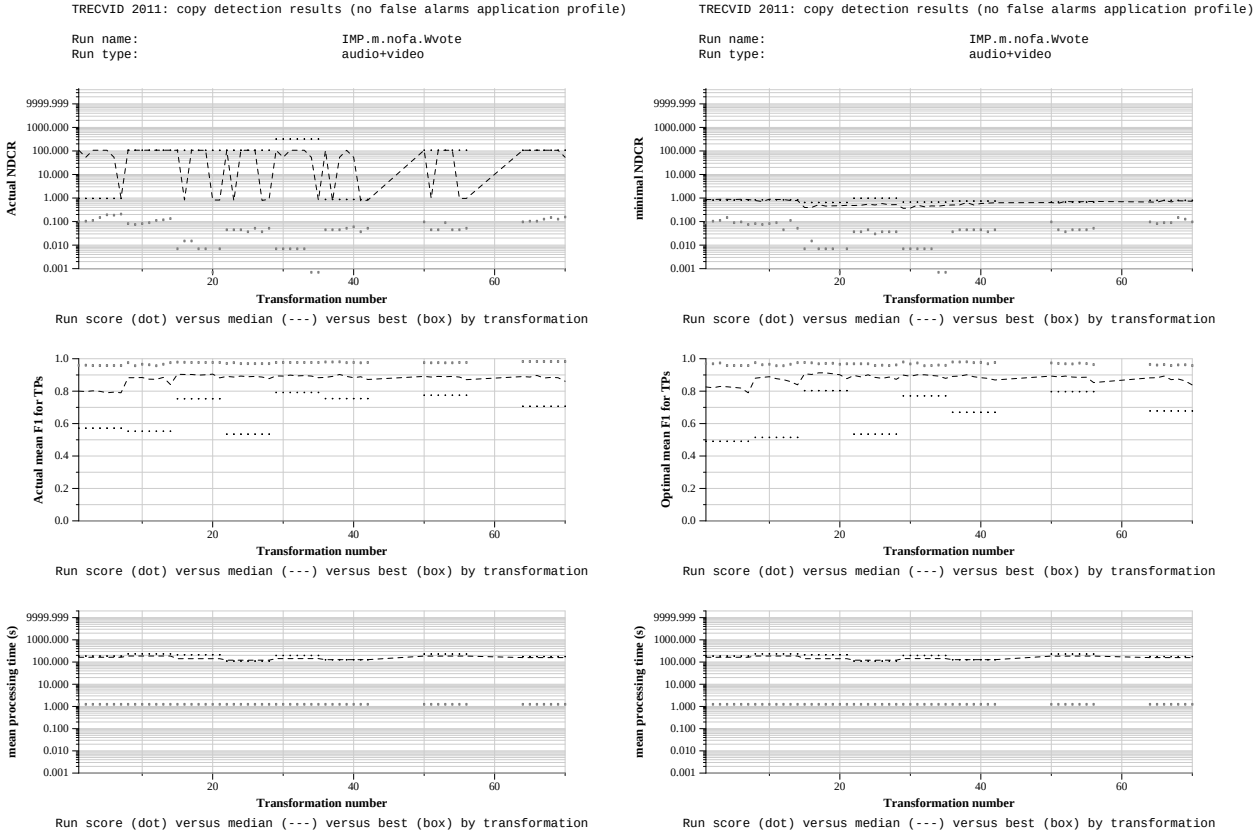

Figure 5. Actual(left) and optimal(right) Results of IMP.m.nofa.Wvote