PKU@TRECVID2010: Copy Detection with Visual-Audio Feature Fusion and Sequential Pyramid Matching[*]

Yuanning Li, Luntian Mou, Chi Su, Xiaoyu Fang, Mengren Qian, Menglin Jiang
Yaowei Wang, Yonghong Tian[+], Tiejun Huang, Wen Gao

National Engineering Laboratory for Video Technology, Peking University
[+] Corresponding author: Phn: +86-10-62758116, E-mail: yhtian@pku.edu.cn

Abstract

Content-based copy detection (CBCD) over large corpus with different types of complex transformation is important but challenging for video content analysis. In practice, some features are robust against certain types of transformations but vulnerable to other types of transformations; other features may be the other way around. To accomplish the TRECVID'2010 CBCD task, we have implemented a cascade detection system, XSearch, to fuse different types of features. In XSearch, several basic detectors over individual features (e.g. audio features, global and local visual features) are firstly employed to maintain candidate pools of matching clips from reference corpus for each query clip. Secondly, a sequential pyramid matching (SPM) is utilized to align two video clips and filter out the mismatched pairs in the candidate pool preliminarily. In SPM, two sequences of sampled video frames are matched in coarse-to-fine multiple resolutions. Finally, all matching results from different basic detectors that still remain in the candidate pool are further filtered and fused with rank-based fusion. We have submitted four runs, i.e., "PKU-IDM.m.balanced.kraken", "PKU-IDM.m.nofa.kraken", "PKU-IDM.m.balanced.perseus" and "PKU-IDM.m.nofa.perseus". Results show that with the cascade detection system, we have achieved excellent performances in NDCR along with a competitive F1 measure.

1. Introduction

Content-based copy detection (CBCD) over large corpus with different types of complex transformation becomes important but challenging for many video applications (e.g., copyright control, advertise tracking and legal/illegal content monitoring). Given a source video, a duplicate video can be produced by different kinds of transformations (e.g., pattern insertion, cam-recoding and cropping in visual content, bandwidth limit, single-band compacting, and mixture with speech in audio signal). However, some features are robust against certain types of transformations but vulnerable to other types of transformations; other features may be the other way around.

To accomplish the TRECVID'2010 CBCD task, we have implemented a cascade detection system, called XSearch (as shown in Fig. 1), to fuse different types of features. During the pre-processing, reference/query videos are firstly split into two components of video and audio. Accordingly, key-frames of video/ audio are sampled with uniform interval. Then several basic detectors over individual feature are firstly employed to maintain candidate pools of matching clips from reference corpus for each query clip. Different types of feature (e.g. audio features, global and local visual features) are firstly extracted from video/audio key-frames. Then two indexes (i.e., LSH and inverted index) are utilized for efficient matching over feature vectors and local features respectively. One candidate pool is built upon the matching results from each basic detector. For global feature and audio features, local sensitive hashing (LSH) [1] is employed for efficient matching of feature vector. Inverted index of bag-of-features [5] is built upon local features (i.e., SIFT [3] and SURF [4]). In the testing phase, a sequential pyramid matching (SPM) is utilized to align query video clips over the matching results of individual features and some mismatched pairs are filtered out from the corresponding candidate pool. Finally, all matching results from different basic detectors that still remain in the candidate pools are further filtered and fused with rank-based method.

The remainder of this paper is organized as follows. Sec. 2 describes the preprocess step. In Sec. 3, three basic detectors are presented. Sequential pyramid matching is introduced in Sec. 4. Fusion method over the results of different basic detectors is presented in Sec. 5. Experimental results are given in Sec.6. Sec.7 concludes this paper.

[*] This work is partially supported by grants from the Chinese National Natural Science Foundation under contract No. 90820003 and No. 60973055, and the CADAL project.

2. Preprocess

Pre-processing consists of two parts (i.e., visual and audio pre-processing). Visual pre-processing includes key-frame extraction, black frame detection and picture-in-picture detection. Key-frames are extracted by uniform sampling with the rate of 3 frames per second. Key-frames where luminance of each pixel is below a predefined threshold are dropped as black frames. Hough transform that detects two pair of parallel line is employed for picture-in-picture detection. Accordingly, foreground picture and picture-in-picture are served as query frames to match with the reference frames. In the pre-process step, audio signal is divided into short time frame which is 60 ms length with overlap rate 2/3 among frames. We brief the preprocess step for different transformations as follows:

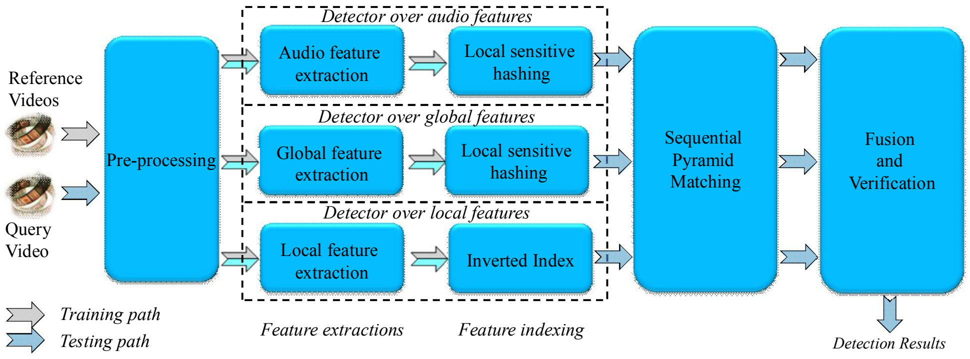

Fig. 1. Paradigm of our system, XSearch

Picture-In-Picture (T2, some T8 and some T10): Our system uses a Hough Transform based method to detect and locate the inserted foreground pictures. For those queries detected with picture-in-picture transformation, the system will process the foreground and background pictures and the original key-frame respectively.

Frame dropping (some T6 and some T10): It has been observed that the dropped frames are replaced with black frames. Hence we employ the black frame detection so that those black frames will not be processed anymore.

Flip (some T8 and some T10): to improve the robustness of our system to the flip transformation, key-frames will be flipped and matched as original key-frames.

3. Basic detectors over different features

Three basic detectors are built upon different types of features such as audio features, global visual and local features. Details about the implementation are summarized as follows:

3.1 Detector over audio features

Weighted ASF (WASF) [6] which is mainly based on the MPEG-7 descriptor - Audio Spectrum Flatness (ASF) and Human Auditory System (HAS) is utilized as audio feature. This feature is proven to be robust to several audio transformations: sampling rate change, noise addition, and speed change. In particular, a stereo waveform is converted into a mono waveform at the pre-process phase. A pre-emphasis of 0.97 is applied and then multiplied by a Hamming window before computing the DCT transform. In order to efficiently depress the noise distortion, a smooth filter bank is used. After that, the Outer ear and Middle ear functions in the Human Audio System (HAS) are applied to weight audio data. Then, the weighted ASF descriptor is used as audio feature. To improve the robust feature, audio feature whose frequency ranges between 250 Hz and 3000 Hz are extracted. Details of WASF can be found at [6].

To improve the matching efficiency of the high-dimensional vectors of audio features and global features (c.f. Sec.3.2), locality-sensitive hashing (LSH) [1] is employed to approximate the near neighbor search in the high dimensional space. The LSH Algorithm is summarized as follows:

1) Given two feature vectors (i.e., v_1 and v_2) of *n* dimension and a distance function $D(v_1, v_2)$, LSH need to find a group of hashing functions $H = \{h_1, h_2, h_3, ..., h_n\}$, where *h* satisfies:

$$\begin{aligned}(D(v_1,v_2) < r_1) &\Rightarrow p[h(v_1) = h(v_2)] > p_1 \\ (D(v_1,v_2) > r_2) &\Rightarrow p[h(v_1) = h(v_2)] < p_2\end{aligned} \quad (1)$$

In Eq.1, r_1 and r_2 are two predefined distance thresholds while p_1 and p_2 are two constant probabilities.

2) Randomly choose k hashing functions from H to construct a hashing function set $g = \{h_1, h_2, ..., h_m\}$, which projects the *n* dimensional vector to *m* dimensional space.

3) Repeat step2 *L* times to obtain *L* different hashing function sets.

4) Give a query vector *q*, LSH maps *q* to *m* dimensional space by hashing function sets obtained in step3. Then *k* nearest neighbors of q are found by searching the *L* hashing tables and comparing the distance function between q and the feature vectors that have the same hashing values of *q*.

3.2 Detector over global features

DCT is utilized as the global feature in our system. It has been shown that DCT is robust to simple transformations such as T4 (Re-encoding) and T5 (Gamma Change). DCT also works well on several complex transformations such as T2 (Picture-In-Picture) and T3 (Pattern Insertion) by pre-processing. In particular, key-frames are firstly converted to color space of YUV, keeping the channel of Y only. Then the Y-channel images are normalized with the size of 64×64 pixels. The normalized image is further divided into 8×8 blocks with the size of 8×8 pixels. After that, a 2-D DCT transformation is conducted over each block to obtain a DCT transform matrix with the same size. The energy for the first four sub-bands (c.f. Fig. 2) is accumulated by summing up the absolute values of the DCT coefficients on each sub-band. Accordingly, a 256-Dimension DCT feature can be obtained by computing the relative magnitudes of the energy using following rules: arranging the energy values of each sub-band for 64 blocks respectively into a ring, comparing each pair of two neighboring energy values, assigning each block with one bit of '1' if the clock-wisely first is larger or equal to the second, or else with one bit of '0'. Hence, a 256-bit DCT feature can be obtained from an individual key-frame. With the efficient matching over LSH and carefully implemented code, extraction and matching of DCT features over the reference/ query corpus can be conducted efficiently within 12 hours on 4-core PC.

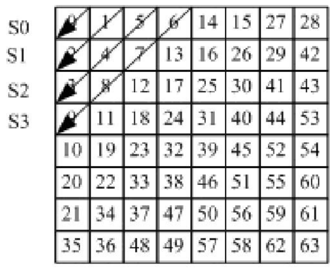

Fig. 2. Illustration of DCT subband

3.3 Detector over local features

Bag-of-words representation [5] is employed for two widely used local features, i.e., SIFT [3] and SURF [4]. All local features are quantized as visual words by *k*-means (*k=400*). A reverted index of visual word is implemented for efficient search and matching. To further improve the performance of local feature matching, spatial, scale and orientation information is also used. In particular, the space of key-frame is divided into 1×1, 2×2, and 3×3 cells. Then the spatial position of each local feature is quantized into three integers ranging from 0 to 20. Orientation and scale of each local feature are also quantized into 8 and 2 bins respectively. Accordingly, such quantized information is integrated within the inverted index. Fig.3 illustrates the inverted index using local features and their quantized spatial information.

As shown in Fig.3, given a query key-frame, SIFT and SURF features are firstly extracted and quantized as visual words by the leant codebooks. Then their spatial information is further quantized into integers. By searching the inverted index, reference key-frames that have similar appearance and spatial layout can be found efficiently. It is shown that such scheme works well on T1 (Camcording), T2 (Picture-In-Picture) and T3 (Pattern Insertion) over large video corpus.

4. Sequential pyramid matching

For each query video, we can get several reference candidates by frame-level voting over the basic detectors of individual features. Then sequential pyramid matching (SPM) is introduced to filter out the mismatch candidates and align the query video and the candidate reference video. Firstly, the subsequence of the candidate reference video is picked out by identifying its first and the last matched key-frames, and likewise the subsequence of the query video. Then the subsequence of the query video slides over the subsequence of the candidate reference video, evaluates their similarity of SPM and finds the most probable position to align two sequences.

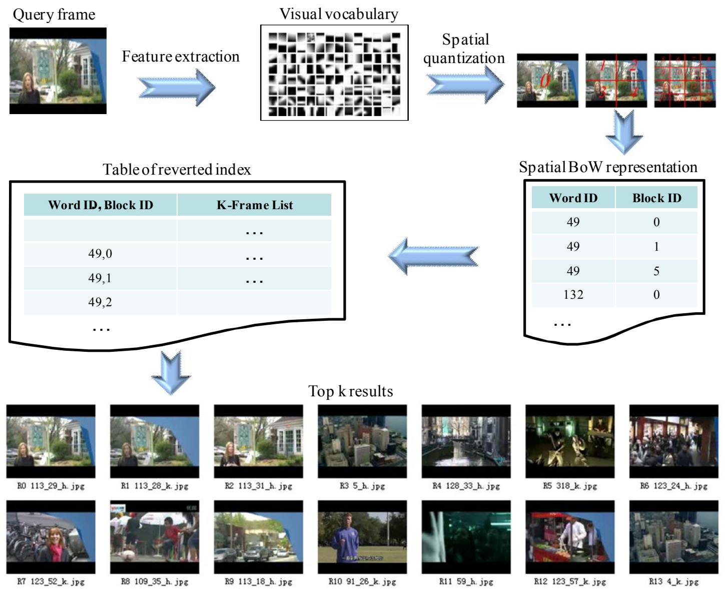

Fig.3. Illustration of key-frame retrieval over the inverted index of visual words and their spatial information.

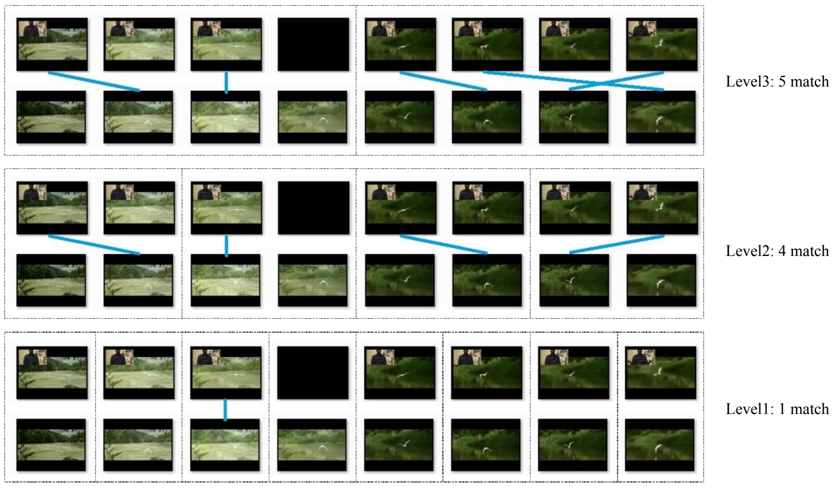

Fig.4. An example of sequential pyramid matching over two key-frame sequences. Note that key-frames within the same block can be matched, and the block number decreases with the level number.

Inspired by the spatial pyramid matching kernel [2], SPM adopts a multi-level sequential matching strategy, where the similarity of two key-frame sequence is evaluated over different granularities. As shown in Fig.4, in the first level, the similarity of sequence is evaluated over k (k = 8) block where key-frames within the corresponding block can be matched across two sequences; In the second level, sequence of key-frames is divided into k/2 blocks; In the third level, sequence goes on to be divided into k/4 blocks; Such procedure continues till there is only one block over each sequence. Finally, similarity of two key-frame sequences is calculated by accumulating the weighted similarities from multiple levels. SPM provides a matching mechanism robust to the mismatching and frame dropping.

5. Fusion and verification

Given the detection results from different basic detectors, a rank-based fusion strategy is utilized to obtain the final detection result. It is expected that the final result should be a subset of the union of all basic detection results. Firstly, items from a single basic detector are ranked separately. Then the intersection of detection results from any two basic detections are accepted. For items only reported by a single basic detector, the policy is summarized as follows: For BALANCED profile, such result items will be accepted only if they're in the top 60% of the basic detector results; for NOFA profile they should locate in the top 40%. Generally speaking, this fusion strategy tends to guarantee True Positive for BALANCED profile and limit False Alarm for NOFA profile. Experimental results show that it works very well.

Compared with "PKU-IDM.m.balanced.kraken" and "PKU-IDM.m.nofa.kraken", runs of "PKU-IDM.m.balanced.perseus" and "PKU-IDM.m.nofa.perseus" have an additional verification. This is because that a byproduct of acceleration the feature matching with bag-of-words representation is accompanied with the decrease in discriminativeness. Hence, we add a verification module where SIFT and SURF features are used to match items only reported by a single basic detector. Results show that such coarse-to-fine process works well.

6. Experimental results

NDCR: NDCR (Normalized Detection Cost Rate) measures the detection effectiveness of a CBCD system, i.e. how many queries it finds the reference data for or correctly tells users there is none to find.

Compared with other participants, our system achieves excellent NDCR performance: for BALANCED profile, our system gets 39 top 1 among 56 "Actual NDCR" and 51 top 1 among 56 "Optimal NDCR"; for NOFA profile, it gets 52 top 1 among 56 "Actual NDCR" and 50 top 1 among 56 "Optimal NDCR". The detailed analysis on Actual NDCR for NOFA profile is shown in table 1, and the analysis on the other three NDCRs are not listed due to space limitations.

As to our NDCR for each transformation, results indicate that NDCRs for "simple" transformations are relatively better (lower) than those for "complex" transformations, which accords with people's intuitive sense. For instance, our NDCRs for video transformation T5 merged with audio transformations T1~T4 are all below 0.01 while the NDCRs for video transformation T10 merged with audio transformation T5~T7 are all above 0.17, as is shown in table 1.

The NDCR measure also verifies our fusion strategy. Compared with BALANCED profile, submissions tuned for NOFA profile have fewer False Alarms at a cost of small decrease in True Positives, and both profiles have achieved good NDCRs. Besides, the "PKU-IDM.m.balanced.perseus" and "PKU-IDM.m.nofa.perseus" runs with additional verification module achieve a little better NDCR than "PKU-IDM.m.balanced.kraken", "PKU-IDM.m.nofa.kraken".

Mean F1: F1 measures the accuracy of localization, i.e. when a copy is detected, how accurately the system locates the reference data in the test data.

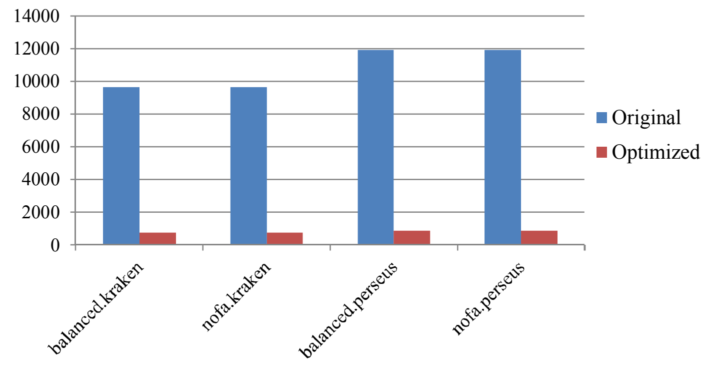

Fig. 5. Mean Processing Time of different runs over original and optimized system.

Our system achieves competitive F1 performance. For both BALANCED and NOFA profile and all the transformations, our F1 measures are all around 0.9 with a few percent of deviation. Besides, our F1 measures for different transformations are at the same level even though the NDCRs varies. This demonstrates that once the correct reference video is found, our Sequential Pyramid Matching strategy generally locates the copy position precisely.

Mean Processing Time: Processing Time measures the efficiency of a CBCD system, i.e. how much elapsed time is required to process a query.

When using all the detectors and strategies discussed above, our system require comparatively long processing time. However, it is worth to mention that our prototype system did not use any parallel programming techniques in the competition. **In fact, currently, processing time has decreased at least by an order of magnitude only by optimization with**

multi-threading and multi-processing (c.f. Fig5). Our system is also configurable. With fewer basic detectors, performance which is comparable to the best results with a small drop can be obtained using much less processing time.

V+A=M	perseus	kraken	best	median	V+A=M	perseus	kraken	best	median
1+1=T1	**0.046**	0.054	0.246	108.048	5+1=T29	**0.008**	0.046	0.046	535.411
1+2=T2	**0.046**	0.054	0.246	108.071	5+2=T30	**0.008**	0.046	0.038	535.657
1+3=T3	**0.046**	0.054	0.262	214.566	5+3=T31	**0.008**	0.046	0.054	535.634
1+4=T4	**0.046**	0.054	0.277	108.064	5+4=T32	**0.008**	0.046	0.054	535.611
1+5=T5	**0.123**	0.169	0.285	108.033	5+5=T33	**0.008**	0.062	0.054	321.537
1+6=T6	**0.138**	0.162	0.285	107.525	5+6=T34	**0.031**	0.092	0.054	321.222
1+7=T7	**0.108**	0.138	0.323	107.541	5+7=T35	**0.015**	0.069	0.069	321.222
2+1=T8	**0.023**	0.038	0.185	428.516	6+1=T36	**0.046**	0.054	0.1	535.403
2+2=T9	**0.023**	0.038	0.185	321.576	6+2=T37	**0.046**	0.054	0.092	535.657
2+3=T10	**0.023**	0.038	0.2	321.576	6+3=T38	**0.046**	0.054	0.108	535.634
2+4=T11	**0.023**	0.038	0.215	321.576	6+4=T39	**0.046**	0.054	0.123	535.611
2+5=T12	**0.062**	0.1	0.223	108.071	6+5=T40	**0.1**	0.2	0.123	214.851
2+6=T13	**0.046**	0.092	0.223	107.641	6+6=T41	0.123	0.185	0.1	214.512
2+7=T14	**0.062**	0.108	0.254	214.666	6+7=T42	0.115	0.185	0.077	214.489
3+1=T15	**0.023**	0.038	0.069	428.516	8+1=T50	**0.046**	0.054	0.138	321.737
3+2=T16	**0.023**	0.038	0.062	535.411	8+2=T51	**0.046**	0.054	0.131	535.411
3+3=T17	**0.023**	0.038	0.077	535.411	8+3=T52	**0.046**	0.054	0.146	535.411
3+4=T18	**0.023**	0.038	0.085	535.411	8+4=T53	**0.046**	0.054	0.162	321.737
3+5=T19	**0.031**	0.069	0.085	321.507	8+5=T54	**0.146**	0.169	0.169	321.514
3+6=T20	**0.031**	0.077	0.085	214.274	8+6=T55	**0.115**	0.138	0.162	215.089
3+7=T21	**0.031**	0.069	0.1	214.381	8+7=T56	**0.138**	0.162	0.185	215.02
4+1=T22	**0.054**	0.069	0.062	428.686	10+1=T64	**0.054**	**0.054**	0.123	428.516
4+2=T23	**0.054**	0.069	0.054	535.411	10+2=T65	**0.054**	**0.054**	0.123	535.411
4+3=T24	**0.054**	0.069	0.077	535.411	10+3=T66	**0.054**	**0.054**	0.138	322.168
4+4=T25	**0.054**	0.069	0.077	535.411	10+4=T67	**0.054**	**0.054**	0.154	322.176
4+5=T26	**0.077**	0.215	0.077	214.281	10+5=T68	0.192	0.215	0.162	108.048
4+6=T27	**0.085**	0.2	0.085	214.312	10+6=T69	0.185	0.223	0.154	214.697
4+7=T28	**0.062**	0.177	0.092	108.056	10+7=T70	**0.177**	0.2	0.185	108.018

Table 1. Actual NDCR Performance for NOFA profile. The "V+A=M" column identify Video Trans. ID, Audio Trans. ID and Video+Audio Trans. ID. The "perseus" and "kraken" columns correspond to the Act. NDCR of "PKU-IDM.m.nofa.perseus" and "PKU-IDM.m.nofa.kraken". Note that the items in bold mean these are the best (lowest) NDCRs among all the participants. The "best" column is the best NDCR obtained by all the other participants (excluding our results), and the "median" column indicates the median NDCR of all the participants (including our results).

7. Conclusion

Official evaluation results show that our system outperforms other systems at most tasks in terms of NDCR and F1. It demonstrates the effectiveness of the adopted strategies: multi-feature extraction, coarse-to-fine matching and fusion on the level of results. Although our system is effective, endeavors will be devoted to the improvements on the efficiency by parallelizing the algorithms and optimizing the implementation.

References

[1] A. Gionis, P. Indyk, R. Motwani. Similarity Search in High Dimensions via Hashing. VLDB, 1999.

[2] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, CVPR, 2006.

[3] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. Int'l J. Computer Vision, 60: pp. 91-110, 2004.

[4] H. Bay, T. Tuytelaars, and L. V. Gool. Surf: Speeded up robust features. ECCV, pp. 404-417, 2006.

[3] G Csurka, C Dance, L Fan, J Willamowski. Visual categorization with bags of keypoints, ECCV Workshop on Statistical Learning in Computer Vision, 2004.

[6] Jianping Chen, Tiejun Huang, "A Robust Feature Extraction Algorithm for Audio Fingerprinting", PCM 2008, pp.887~890.