A Computationally Efficient Algorithm for Large Scale Near-Duplicate Video Detection

Dawei Liu[1] and Zhihua Yu[1,2]

[1] Instititue of Network Technology, Institute of Computing Technology(Yantai), CAS, Shandong, P.R. China
[2] Instititue of Computing Technology, CAS, Beijing, P.R. China
liudw@int-yt.com, yzh@ict.ac.cn

Abstract. Large scale near-duplicate video detection is very desirable for web video processing, especially the computational efficiency is essential for practical applications. In this paper, we present a computationally efficient algorithm based on multi-layer video content analysis. Local features are extracted from key frames of videos and indexed by an novel adaptive locality sensitive hashing scheme. By learning several parameters, fast retrieval in the new hashing structure is performed without high dimensional distance computations and achieves better real-time retrieving performance compared with other state-of-the-art approaches. Then a descriptor filtering method and a two-level matching scheme is performed to generate a relevance score for detection. Experiments on near-duplicate video detection tasks including various transformed videos demonstrate the efficiency gains of the proposed algorithm.

Keywords: Near-duplicate Detection, Locality sensitive hashing, SURF, Multimedia content analysis.

1 Introduction

With the rapid growth of digital video content production on the web, near-duplicate video detection for protecting and managing video content has received growing attention over the last decade. There are two general techniques for near-duplicate video detection: digital watermarking and content-based copy detection. Compared with digital watermarking, which embeds hidden data information called watermarking in an image or video, the content-based techniques, which employs video content analysis and detects video copies by video signatures or key frame features, leads to better efficiency and effectiveness.

Recently, most approaches focus on the content-based near-duplicate video detection. The general procedure of existing work can be summarized as three stages. First, using shot detection methods, videos are segmented into clips, which then represented by one or more key frames. Second, a set of high dimensional feature vectors are extracted by feature detector and descriptor. Finally, the similarity between videos is computed from the feature vectors under spatial and/or temporal sequence matching schemes [7][11][14]. While the feature representation stage can be further classified

X. He et al. (Eds.): MMM 2015, Part II, LNCS 8936, pp. 481–490, 2015.

into two categories: global feature and local feature, each has different design of video content representations and similarity metrics between feature sequences. Yeh et al. proposed a global frame-level descriptor [6], which is a compact 16-dimensional feature vector based on computing the spectral properties of a graph built from partitioned blocks of a frame, and a fast sequence matching scheme: dot plot[5]. Chiu et al. [13] combines both global and local feature descriptors and integrates min-hashing and spatiotemporal matching to detect video copies. Shang et al. [8] introduced a binary spatiotemporal feature which is global and fast to compute using an indexing structure based on inverted file. Liu et al. [9] described a framework which used SIFT as local feature and employed locality sensitive hashing (LSH) [3] and random sample consensus (RANSAC) techniques to index and detect. Avrithis et al. [12] quantized local features to visual word and used a RANSAC-like matching method. Comparative study of state-of-the-art techniques [10] concluded that local feature-based methods are more robust but more computational expensive.

In this paper, we consider the content-based near-duplicate video detection based on local features. We employ SURF [2] as local visual features and we design an LSH-based indexing structure with parameterizations to reduce the computational cost, while maintaining the scalability and robustness. After the feature vectors retrieval, a descriptor filtering method is applied to further reduce the number of candidate feature sets and then a two-level matching scheme to generate a relevance score for detection. To the best of our knowledge, few framework has yet been present to merge LSH with SURF to design video content indexing and retrieval. The only work similar to ours is [1] in which Zhang et al. used SURF features and LSH indexing separately without optimization. In contrast, our algorithm leverages the characteristics of both feature vectors and indexing structures to reduce computational cost and boost retrieval performance.

The rest of this paper is organized as follows. In Section 2, we present our adaptive locality sensitive hashing for SURF indexing. Section 3 introduces the descriptor filtering and two-level matching scheme. Section 4 gives the experimental results and performance analysis of our proposed algorithm. Finally we conclude this paper and give some future work in Section 5.

2 Adaptive Locality Sensitive Hashing for SURF Indexing

Fig.1. illustrates the near-duplicate video detection framework of our algorithm. The processing consists of two parts: Indexing and Retrieval. Indexing videos are processed by shot detection, key-frame and SURF extraction to generate a set of 64-dimensional feature vectors. Then a video database is built using an indexing structure. In the retrieval parts, the same local features extraction is performed and by retrieving in the database a candidate result set is generated and filtered, then two-level matching methods are applied to get the final near-duplicate video results. We focus on the adaptive LSH indexing, feature filtering, frame-level and video-level matching, as indicated by shades.

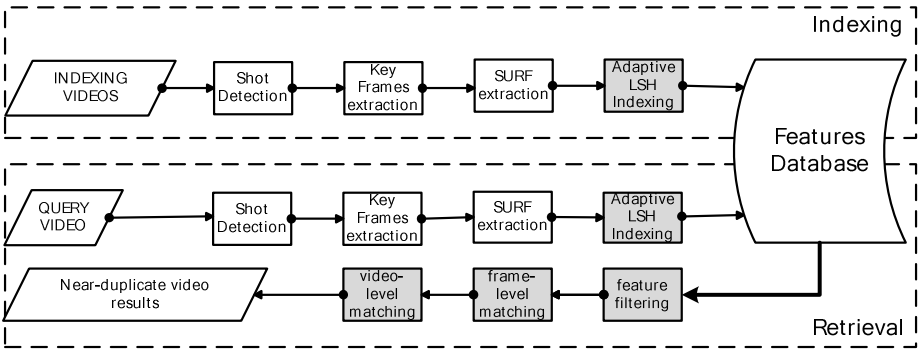

Fig. 1. Overview of Near-Duplicate Video Detection Framework

2.1 Insights into SURF and LSH

SURF (Speeded-Up Robust Features) detector and descriptor [2] is based on calculating approximate Hessian response for image points and is efficiently implemented on the basis of integral images. SURF is proved to be equal or superior to performance and significantly better computational efficiency in comparison with other local feature methods, such as SIFT, PCA-SIFT. Moreover, some intermediate results, such as the sign of the Laplacian (i.e. the trace of the Hessian matrix) and the position of each interest points, can be used to accelerate the feature vector matching process with no extra computational cost.

LSH (Locality Sensitive Hashing) is introduced in [3] for approximate nearest neighbors search in high dimensions. LSH function families have the property that objects that are close to each other have a higher probability of colliding than objects that are far apart. For different distance measures, different LSH families have been proposed. We consider the LSH for p - norms based on p -stable distributions [4] since SURF descriptors are designed to measured by the Euclidean distance. In the basic LSH scheme, a query point is hashed into several buckets in different hash tables to retrieve all points in these buckets, then the distances to each point is computed. We argue that the Euclidean distance computing in high dimensions (i.e.64-dimensions for SURF) has a high computational complexity and becomes the performance bottleneck for existing LSH-based methods [9][13]. While simply without distance computing, retrieving all points in several buckets to candidate sets also results in high computational cost in the next multi-level matching process due to the large number of points contains in these buckets. We introduce the Adaptive LSH with parameterizations to solve the problem. The key idea of our adaptive LSH scheme is that the accuracy of near-duplicate video detection based on video content analysis depends on multiple factors and the fine granularity of SURF is enough for frame-level matching. In our scenario, speed is more important than "local accuracy", more effective filtering and less unnecessary computing is demanded.

In comparison with global feature based methods, which use one compact feature vector to describe one key frame, local feature based methods characterize one key

frame using hundreds of points (for SURF about 100 interest points generated without filtering) and are more robust yet more computational cost. Therefore, boosting the retrieval efficiency is an essential issue for local feature based methods.

2.2 Adaptive LSH (ADLSH) with Parameterizations

We leverage the characteristics of SURF to design a sign function for each point p as:

$$sign(p) = \begin{cases} 1 & \text{if the trace of the Hessian metrix is 1} \\ -1 & \text{others} \end{cases}$$

Merging the sign function with the basic LSH function, each of our adaptive LSH function is represent as:

$$h_{a,b}(p) = \left\lfloor \frac{a \cdot p + b}{W} \right\rfloor \cdot sign(p) \tag{1}$$

where p is a 64-dimensional SURF feature descriptor in our algorithm. Accordingly a is a 64-dimensional random vector with entries chosen independently from a 2-stable distribution (i.e. here is the Gaussian distribution for the Euclidean distance) and b is a real number chosen uniformly from the range $[0, W]$. Each hash function $h_{a,b}(p)$ maps a 64-dimensional vector v onto the set of signed integers. To implement the indexing structure, each point p is hashed into L buckets in L hash tables: $g_j(p)$, for $j = 1, ..., L$, where $g_j(p) = \left(\left|h_{1,j}(p)\right|, ..., \left|h_{k,j}(p)\right|\right)$. We use the absolute value $\left|h_{a,b}(p)\right|$ of each function to generate the k-dimensional label of each bucket and then split each bucket into two parts considering the sign of point. The number of buckets in our scheme is about twice the basic LSH, accordingly the number of points in each buckets is averagely reduced by 50%..

One challenge with LSH-based methods is that there are several parameters: W, k, L to be tuned, while existing approaches determine parameters by experimental results, which are uncertain and impractical. We analyze parameter restrictions and propose a sample learning approach to set parameters automatically during indexing and at the same time to further reduce the average amount of points in each buckets. We design our adaptive LSH for R-near neighbor search in 64-dimensional SURF descriptor space. To process a query point q, we retrieve all points in buckets $g_1(q), ..., g_L(q)$ without distance computing while guarantee that the resulting candidate set contains all R-near neighbors v of q (i.e. $\left\|q - v\right\|_2 \le R$). As proved in [4], basic LSH solve the R-near neighbor problem with a probability at least $1 - \delta$, where δ is the fail probability (0.1% in our implement) and for two

points p_1, p_2, let $c = \|p_1 - p_2\|_2$, the probability of the two points collide under a hash function chosen uniformly at random from LSH family is:

$$p(c) = \Pr_{a,b}\left[h_{a,b}(p_1) = h_{a,b}(p_2)\right] = \int_0^W \frac{1}{c} f_2(\frac{t}{c})(1-\frac{t}{W})dt \tag{2}$$

where $f_2(t)$ denote the probability density function of the absolute value of the Gaussian distribution ($p = 2$):

$$f_2(t) = \begin{cases} \frac{2}{\sqrt{2\pi}} e^{-t^2/2} & \text{if } t \geq 0 \\ 0 & \text{if } t < 0 \end{cases}$$

For the R-near neighbor problem in our scenario (without loss of generality we assume that $R = 1$), to retrieve all points with the Euclidean distance less than 1, the following condition is necessary:

$$p(c) \geq p(1) = 1 - 2\Phi(-W) - \frac{2}{\sqrt{2\pi}W}(1 - e^{-(W^2/2)}) \tag{3}$$

Note that each bucket is labeled by a k -dimensional vector, the label collision probability is:

$$\Pr_g\left[g(q) = g(p)\right] = \prod_{i=1}^{k} p_i(c) \geq p(1)^k \tag{4}$$

For all L functions, a query point q finds a 1-nearest neighbor with the following probability:

$$\Pr_{NN}\left[\|q - p\|_2 \leq 1\right] = 1 - (1 - p(1)^k)^L \geq 1 - \delta \tag{5}$$

For a fixed $p(c)$, the optimal value of W is a function of c and decreasing W decreases the $p(c)$ for any two points. Moreover, increasing k or decreasing L decreases the probability to find nearest neighbors. We design the following three steps to determine the parameters:

1) sample learning:

We randomly choose m pairs of total n SURF feature points extracted from the indexing videos to be samples, we estimate $p(c)$ with:

$$c = \frac{1}{m}\sum\nolimits_m \|p_i - p_j\|_2$$

and the average number of collisions in each bucket to be:

$$N_{bucket} = \sum\nolimits_n p(c_n)^k \approx \sum\nolimits_n p(c)^k .$$

2) W :

We adopt the minimum power of 2 under restriction (3) to be the proper value for parameter W .

3) k and L

Our aim is to reduce the number of points retrieved by one query point from L buckets to about one percent of the total number for small database or about a constant (we choose 10000 in experiments of this paper) for large database, that is: the parameter k and L are determined by:

$N_{bucket} \cdot L \approx \min(1\% \cdot n, 10000)$ while under restriction (5).

Fig. 2. An example of filtering false matching pairs. Left: a key frame extracted from a indexing video; Right: a key frame extracted from a query video with transformations of resizing and subtitles. All the lines denote matching pairs of original SURF descriptors. Red lines denote the false matching pairs which are filtered by adaptive LSH using bucket splitting; Blue lines denote the false matching pairs which are filtered by Descriptor Filtering method based on relative distances; Yellow lines denote the right matching pairs which compose the candidate sets after pruning in our algorithm.

2.3 Descriptor Filtering (DF)

After splitting buckets according to (1) and three steps parameterizations, for each point in each key frame, a predictable size of candidate set is generated without high dimensional distance computations. Note that the SURF descriptor matching by Euclidian distance is not robust to noisy transformation, some geometrical verification techniques, such as RANSAC, are used in most recent studies. Since our candidate set is well pruned, we simply introduce a distance-based filter to remove most false matched points. For each pair of descriptor, we leverage the position coordinates, which recorded during SURF feature extraction without additional computation, to calculate a relative distance in 2-dimensional space. Mean and standard deviation are computed for distances of all descriptors in each key frame (image) and those pairs whose distances have a large difference (exceeds standard deviation greatly) to mean are considered to be noisy points. Fig. 2. as an example illustrates the filtering effect of our algorithm.

3 Two-Level Matching Scheme

In Section 2, all matched SURF descriptors is detected using adaptive LSH indexing structure, a two-level matching scheme is proposed in this section. During indexing and retrieving the 64-dimensional SURF feature descriptors, corresponding video identifiers, key frame identifiers and the numbers of points in the same bucket are recorded with each descriptor. We hash each descriptor based on key frame identifiers and after filtering the candidate set, a linear scan is performed for each frame to select the frames that have a number of matching descriptors upon a threshold (for about 100 SURF descriptors per frame in our scenario, we choose the threshold to be 60). Therefore we got the near-duplicate results at the frame-level.

To get the near-duplicate video results of a query video, for each key frame f_i^q in the query video, the similarity between f_i^q and a key frame f_j^c of an indexing video is defined as:

$$sim(f_i^q, f_j^c) = \sum_{N_{descriptor}} \sum_{L} w_{i,j} \cdot N_{matching} \tag{6}$$

where $N_{descriptor}$ is the number of descriptors extracted from the query frame, $N_{matching}$ is the number of matching descriptors between the two frames in one bucket and $w_{i,j} = 1/N_{bucket}$ is the weight of the corresponding bucket. We use the weight $w_{i,j}$ to reduce the impact of large buckets, in which many descriptors of the same frame match to one query descriptor of the query frame.

The relevance score for one indexing video v_c to the query video v_q is defined as:

$$score_c = \frac{\sum_{N_{frame}} sim(f_i^q, f_j^c)}{N_{frame}} \tag{7}$$

where N_{frame} is the number of key frames of the query video. An indexing video v_c is considered to be a near-duplicate video of the query video if $score_c$ exceed a threshold S_t. The selection of S_t requires a trade-off between recall and precision during detection and is data dependent. Finally, the videos with first several highest scores are returned as the results of near-duplicate video detection.

4 Experiments

To evaluate the computational efficiency and robustness of our proposed algorithm, we conducted experiments using the MUSCLE-VCD benchmark [15] , which is an evaluation set of the TRECVID 2008 copy detection task. The dataset consists of 101 videos with a combined length of about 100 hours. In comparison with method proposed in [1], we tested precision, recall rates and computation time using the provided

15 query videos with different transformations such as blur, color adjustment, resizing, subtitles, encoding and crops. Let ADLSH, ADLSH+DF and BLSH represent the proposed adaptive locality sensitive hashing scheme, the adaptive LSH with descriptor filtering method and the basic LSH scheme used in [1], respectively.

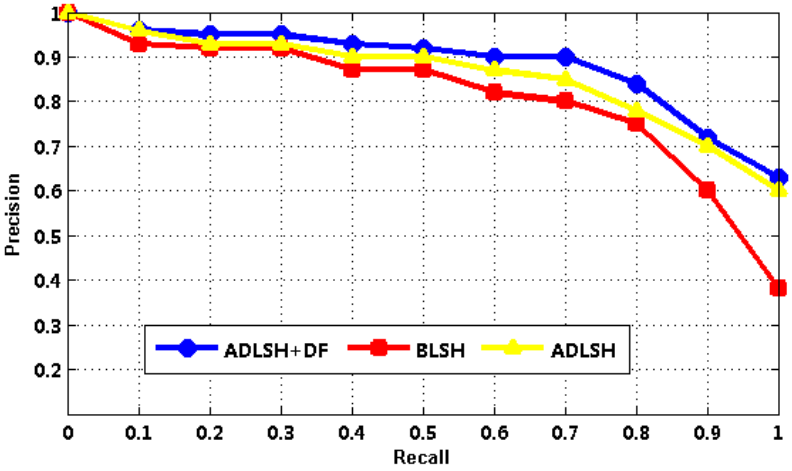

Fig. 3. Average Precision/Recall Curves across 15 query videos

Fig. 3. shows the comparison results of ADLSH+DF, ADLSH and BLSH evaluated by Precision-Recall curves. We can see that for recalls less than 0.8, all the three methods achieve high precisions larger than 0.8. At recalls larger than 0.8, our methods ADLSH+DF and ADLSH have higher precisions than BLSH. Moreover, ADLSH+DF slightly outperforms ADLSH. These results illuminate the impact of filtering. Matching SURF features under Euclidean distance without any filtering introduces noise in candidate sets and therefore cause the precision degrades. Filtering descriptors using the sign function and the relative distance contributes to robustness of detection.

Table 1. Efficiency of Indexing and Retrieval

Methods	Storage(MB)	Pruned Points	Time(s)
ADLSH+DF	19.6	6536	0.095
ADLSH	19.6	5198	0.068
LSH in [1]	18.5	-	0.347

To evaluate the computational efficiency gains of our algorithm, we focus on the index structure and retrieval speed. About 1.83×10^6 SURF descriptors (64-dimensional vectors) are extracted from 101 indexing videos and about 1.26×10^5 SURF descriptors are extracted from 15 query videos. We tested the average storage for each indexing video in the indexing structure, the number of pruned points during

process and the average retrieval time cost for each query point (Note that time cost for shot detection and key frame extraction in Fig. 1. are not considered here). Table 1 shows the results. Since our proposed algorithm (ADLSH+DF, ADLSH) maintain additional information for the use of filtering, about 5% more storage space is needed. While compared with BLSH, which applies no filtering method, ADLSH pruned 5198 points per query and ADLSH+DF pruned about 1400 more points. Both algorithms achieved 3.6-5.1 times faster than BLSH. (ADLSH+DF is slower due to the descriptor filtering method). The results also illustrate that the high dimensional distance computing in BLSH seriously degrades the time efficiency, while our algorithm builds indexing structure with proposed parameterizations to avoid the inefficient computation and therefore accelerates the retrieval process.

5 Conclusions and Future Work

In this paper, we presented a computationally efficient algorithm for large scale near-duplicate video detection. An adaptive LSH scheme with parameterizations is proposed to index SURF feature vectors. Detection queries are retrieved in the proposed hashing indexing structure without high dimensional distance computations. The candidate set is then pruned by indexing buckets splitting and descriptor filtering methods. Relevance scores are generated using two-level matching methods and detection results are determined. The experimental results show the robustness preservation and computational efficiency gains of the proposed algorithm. Our future work will focus on: 1) the combination of our adaptive LSH indexing and different sequence matching algorithms, and 2) enhancing the parameterizations in an incremental way.

Acknowledgments. This work is supported by National Grand Fundamental Research 973 Program of China (No. 2013CB329602) and National Science Supported Planning (No. 2014AA015204).

References

1. Zhang, Z., Cao, C., Zhang, R., Zou, J.: Video Copy Detection Based on Speeded Up Robust Features and Locality Sensitive Hashing. In: Proc. IEEE Int. Conf. Automation and Logistics, pp. 13–18 (2010)
2. Bay, H., Tuytelaars, T., Van Gool, L.: Speeded-up Robust Features (SURF). Comput. Vis. Image Underst. 3(110), 404–417 (2008)
3. Gionis, A., Indyk, P., Motwani, R.: Similarity Search in High Dimensions via Hashing. In: Proc. Int. Conf. Very Large Data Bases, pp. 518–529 (1999)
4. Datar, M., Immorlica, N., Indyk, P., Mirrokni, V.: Locality-sensitive hashing Scheme Based on p-stable Distributions. In: Proc. ACM Symposium on Computational Geometry (2004)
5. Yeh, M.,, K.: T Cheng: Fast Visual Retrieval Using Accelerated Sequence Matching. IEEE Trans. Multimedia 13(2), 320–329 (2011)
6. Yeh, M., Cheng, K.T.: A Compact, Effective Descriptor for Video Copy Detection. In: Proc. ACM Int. Conf. Multimedia, pp. 633–636 (2009)

7. Caspi, Y., Irani, M.: Spatio-Temporal Alignment of Sequences. IEEE Trans. Pattern Anal. Mach. Intell. 24(11), 1409–1424 (2002)
8. Shang, L., Yang, L., Wang, F., Chan, K., Hua, X.: Real-time Large Scale Near-duplicate Web Video Retrieval. In: Proc. ACM Int. Conf. Multimedia, pp. 531–540 (2010)
9. Liu, X., Liu, T., Gibbon, D., Shahraray, B.: Effective and Scalable Video Copy Detection. In: Proc. ACM Int. Conf. Multimedia Information Retrieval, pp. 119–128 (2010)
10. Law-To, J., Chen, L., Joly, A., Laptev, I., Buisson, O., Gouet-Brunet, V., Boujemaa, N., Stentiford, F.: Video Copy Detection: A Comparative Study. In: Proc. ACM Int. Conf. Image and Video Retrieval (2007)
11. Kim, C., Vasudev, B.: Spatiotemporal Sequence Matching for Efficient Video Copy Detection. IEEE Trans. Circuits Syst. Video Technol. 15(1), 127–132 (2005)
12. Avrithis, Y., Tolias, G., Kalantidis, Y.: Feature Map Hashing: Sub-linear Indexing of Appearance and Global Geometry. In: Proc. ACM Int. Conf. Multimedia, pp. 231–240 (2010)
13. Chiu, C., Wang, H., Chen, C.: Fast Min-hashing Indexing and Robust Spatio-temporal Matching for Detection Video Copies. ACM Trans. Multimed. Comput. Comm. Appl. 6(2), Article 10 (2010)
14. Poullot, S., Buisson, O., Crucianu, M.: Scaling Content-based Video Copy Detection to Very Large Databases. Multimed. Tools Appl. 47, 279–306 (2010)
15. Law-To, J., Joly, A., Boujemaa, N.: Muscle-VCD-2007: A Live Benchmark for Video Copy Detection (2007)