A SUFFIX ARRAY APPROACH TO VIDEO COPY DETECTION IN VIDEO SHARING SOCIAL NETWORKS

Ping-Hao Wu, Tanaphol Thaipanich and C.-C. Jay Kuo

Ming Hsieh Department of Electrical Engineering and
Signal and Image Processing Institute
University of Southern California, Los Angeles, CA 90089-2564

ABSTRACT

To address the multiplicity and copyright issues on file sharing social networks, we propose a fast video copy detection algorithm using the suffix array data structure in this work. The proposed algorithm consists of two steps. In the first step, we extract robust features which are discriminative yet insensitive to various attacks. Specifically, we develop a compact one-dimensional signature based on the shot change position of video files. Unlike images and audio, the size of a video file is usually large, which makes it computationally expensive to match two long signature sequences. Thus, in the second step, we adopt an efficient matching technique based on the suffix array data structure. The proposed system can perform the sequence matching in linear time while the complexity of conventional duplicate video detection algorithms grows at least quadratically with the video length.

Index Terms— Video sharing, video copy detection, suffix array.

1. INTRODUCTION

Digital video technologies have been growing rapidly thanks to the advances in video capturing, coding and the broadband network infrastructure. These developments allow users to edit, re-produce, and re-distribute video contents of various formats easily nowadays. Fueled by the need of video sharing platforms, many websites such as YouTube and Google Video provide space for users to store and share their video contents over the Internet. One issue critical to file sharing social networks is the control of copyright of video uploaded by users. The amount of video contents available over the Internet has been growing exponentially. As of March 2008, a YouTube search returns about 77.3 million video titles. There exist quite a few duplicates among all video files spread over the Internet because of the absence of central monitoring and management. Thus, another important techniques demanded by file sharing networks are efficient video copy clustering and elimination.

As an alternative to watermarking, content-based video copy detection has drawn a lot of attention recently since it can be applied to all video before or after distribution. The main idea is to use the unique characteristics inherent in video to achieve copyright protection. The unique characteristics, called features or signatures, of a query video file are extracted and compared with those of video contents in the database. An appropriate matching scheme is then applied to see if the given video is a duplicate of certain video in the database. Existing duplicate video detection systems have some shortcomings. The signatures used in most algorithms are frame-based, and direct comparison of signature vectors is required. They are not only sensitive to various attacks but also computationally expensive [1]–[3].

Here, we propose a novel video copy detection algorithm to address these shortcomings. First, instead of extracting signatures from frames and comparing frame-based signatures directly, the video signature is extracted based on the temporal variation of the underlying video. The one-dimensional video signature is very compact. Besides, since it is not related to frame characteristics directly, it is less sensitive to various attacks applied. Second, we adopt an extremely fast matching technique using the suffix array data structure. The proposed system can perform sequence matching in linear time while the complexity of conventional video copy detection algorithms grows at least quadratically with the video length.

The rest of the paper is organized as follows. We present a compact video signature by examining the temporal variation of a video in Sec. 2. An efficient signature matching algorithm using the suffix array data structure is described in Sec. 3. Experimental results are shown in Sec. 4 with discussion. Finally, concluding remarks are given in Sec. 5.

2. SIGNATURE EXTRACTION

Most previous work on video copy detection was based on finding one or more low-level features such as color, texture, or motion. However, it is apparent that these features could be easily altered even with minor changes. In addition, if features are extracted on the frame basis, the resultant signature would demand a large storage space and a high computational complexity in video copy search. Noticing these drawbacks, the proposed video signature for duplicate detection is based on a higher level feature; namely, the temporal structure of video.

2.1. Video Temporal Structure

Although a video title can be seen as a long sequence of still images, it actually contains more information than merely a series of frames. Specifically, in contrast with still images, video has a series of events evolving with time. The temporal composition of these events uniquely defines the characteristics of the underlying video and, therefore, constitutes its *temporal structure*. The video temporal structure can be represented by a set of *anchor frames*, which mark important events in video sequences. Shot boundaries can be seen as part of anchor frames, considering that they are inherently linked to the way how video being produced. Although it was reported that such a high-level signature alone tends to provide insufficient

This work was supported in part by National Science Council of Taiwan under contract NSC-095-SAF-I-564-045-TMS.

information, we will show that an extremely efficient and compact signature can be generated using video shot boundaries alone, which maintains high query precision.

2.2. Shot Boundary Detection for Video Signatures

Shot boundary detection has been extensively studied and several papers were published to give an overview and comparison of various shot boundary detection algorithms, *e.g.,* [4]. For the video copy detection problem, complicated shot boundary detection algorithms tend to be more sensitive to various changes. Thus, the algorithm used in this work is a simple one, which is based on the luminance histogram difference along with an adaptive threshold after temporal subsampling. The procedure is described below.

Frame Rate Re-sampling The input video is temporally sampled at an interval of 0.2 seconds. Although some information is lost during the subsampling process, it makes sense since we intend to capture the structure of the underlying video instead of representing it by frame-based features. It is observed that the duration of 0.2 seconds provides a good trade-off between key visual information preservation and computational complexity reduction. Furthermore, the slight variation in the shot duration caused by the difference in the frame rate can be avoided. By temporal subsampling, gradual transitions that have approximately the same length as the subsampling factor can be detected.

Distance Measurement The distance measure between frame f_t and frame f_{t+1} (after temporal subsampling) is calculated using the L_1 norm as

$$d(t) = \frac{1}{N} \sum_{i=0}^{K} \left| \hat{h}_i(t) - \hat{h}_i(t+1) \right|, \tag{1}$$

where $\{\hat{h}_i(t)\}$ and $\{\hat{h}_i(t+1)\}$ denote the cumulative luminance histogram for f_t and f_{t+1}, respectively, K is the number of bins for the histogram, and N is the number of pixels in a frame. If K is too large, a small variation would place similar pixel values at different bins, thus increasing the distance between the two frames that are similar. On the other hand, a bin that is too coarse would lose information so that it may not give the desired discriminating power. To seek a good balance between the above two extremes, we use a relatively fine bin number ($K = 128$) and the cumulative histogram in our system, which can take the cross-bin effect into account. After computing the distance measure, it is further filtered via

$$\tilde{d}(t) = \left| d(t) - \mathrm{MED}\left(d(t-1), d(t), d(t+1)\right) \right|, \tag{2}$$

where $\mathrm{MED}(\cdot)$ denotes the median filtering operation. Note that histogram-based distance measures have one serious drawback in detecting shot boundaries. That is, a short and abrupt illumination change such as the flash of light could introduce artificial shot boundaries. However, as long as such artificial boundaries can be detected in both sequences, they have no effect on our video copy detection system. In fact, an abrupt illumination change is exactly an unique event that we want to detect.

Adaptive Thresholding If the distance measure given in Eq. (2) exceeds a certain threshold, there might be an abrupt change in the video content. It means that the current frame represents an unique event and should be marked as an anchor frame. In our system, the threshold, T, is determined automatically according to the statistics within a short time window W of length L around the frame of concern:

$$T(t) = \mu_d(t) + \alpha \cdot \sigma_d(t), \tag{3}$$

where α is a scale factor, $\mu(t)$ and $\sigma_d^2(t)$ are the sample mean and the sample variance of the filtered distance measure in window W, respectively. Mathematically, $\mu(t)$ and $\sigma_d^2(t)$ can be calculated as

$$\mu_d(t) = \frac{1}{L} \sum_{i \in W} \tilde{d}(i), \tag{4}$$

$$\sigma_d^2(t) = \frac{1}{L} \sum_{i \in W} \left(\tilde{d}(i) - \mu_d(t) \right)^2. \tag{5}$$

Signature Generation The misalignment problem could happen in an attack. That is, the video clip could be padded or cropped temporally and the time instances of anchor frames are then shifted by an offset. Thus, the actual timing information of the event occurrence is not important. Instead, the lengths between each adjacent pair of anchor frames would stay the same even after content modifications. Thus, for a given video clip, after extracting the set of anchor frames, we compute the length between the current anchor frame and its previous one, and store all the length information into a one-dimensional sequence. This sequence is the signature for the video content, which is named as the *shot length sequence*. It serves as a good signature since it is highly unlikely for two unrelated video clips to have a long set of consecutive anchor frames with the same shot lengths. It is also worthwhile to point out that, since the set of shot boundaries is only a subset of anchor frames due to downsampling and median filtering, this sequence may not be necessarily the same as the actual shot length sequence obtained by traditional shot change detection algorithms.

3. SIGNATURE MATCHING WITH SUFFIX ARRAY DATA STRUCTURE

After obtaining the shot length sequence as the signature, the next task is to match the signatures of any two video sequences efficiently. In this work, the video copy detection problem is converted into a sequence alignment problem, where the sequence to be aligned is the shot length sequence. Traditional sequence alignment algorithms rely on dynamic programming. The computational complexity of a naïve version of dynamic programming is $O(mn)$, where m and n are lengths of two sequences under consideration. As sequences get longer, the time needed for alignment grows rapidly, which makes dynamic programming impractical for longer video sequences such as movies and/or TV programs. For the video copy detection application, the amount of video can be huge and their shot length sequences could be long. A suitable data structure and an effective algorithm that allow efficient matching are needed.

Suffix Array The suffix array [5] and its variants provide an ideal data structure for string processing and sequence alignment. One such application in bioinformatics is to identify identical subsequences from two DNA sequences [6]. Being motivated by this application, we adopt the suffix array data structure in the signature matching process. The suffix array is basically a lexicographically sorted list of all suffixes of the input string. The suffix array records the starting position (the suffix number) of each suffix only. The suffix number array along with the sorted suffix list provides a compact representation of the original string. The construction of suffix array can be achieved in linear time. Once a suffix array being constructed, it can be used to efficiently solve numerous string processing problems with the assistance of one or more additional arrays [6].

Matching using Suffix Array Depending on the attack types, some of anchor frames could only be detected in one video clip but not in the duplicate one, which means that the shot length sequences

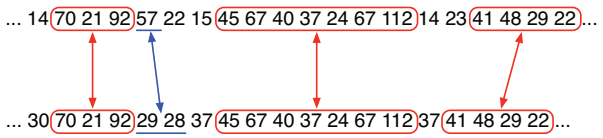

Fig. 1. An example of matching between two sequences, where red circles indicate the maximal unique matches.

may not be exactly the same. Thus, instead of matching the shot length sequence as a whole, we find all matching subsequences in them, which is the problem of identifying maximal unique matches. Given two sequences S_1 and S_2, a maximal unique match is defined as a subsequence that occurs exactly once in both sequences but not contained in any longer subsequence. Fig. 1 shows an example, where maximal unique matches are labeld by red circles. The underlined 57 in one sequence is the sum of $\{29, 28\}$, which occurs because an anchor frame is missed in the top sequence but still detected in the bottom one. Finding the set of all maximally unique matches is not computationally trivial. A naïve algorithm would compare $O(n^2)$ subsequences in one sequence with those in another sequence, and each comparison requires a complexity of $O(n)$.

It is possible to find all maximal unique matches using the enhanced suffix array in $O(n)$ time [6], which is much faster than the naïve algorithm. Besides the suffix array, two additional arrays are needed. They are the longest common prefix (LCP) array and the Burows-Wheeler transformation (BWT) array. The i^{th} entry of the LCP array stores the length of the longest common prefix between the i^{th} and the $(i-1)^{th}$ suffix in the suffix array, while the i^{th} entry of the BWT array stores the character right before the i^{th} suffix in the suffix array. These two arrays can be computed in $O(n)$ time. The algorithm is described below.

Fast shot length sequence matching algorithm

Let S_1 and S_2 be the signatures of two video clips, *i.e.,* the shot length sequences of the query video and a video clip from the database, respectively. We can perform the following to achieve fast matching.

1. Concatenate sequences S_1 and S_2 to form a single long sequence $S = S_1 \# S_2$, where $\#$ is a separator symbol that does not occur in S_1 or S_2.
2. Construct the enhanced suffix array representation for S. This step is done in $O(n)$ time, where $n = |S|$, is the length of S. By exploiting this compact representation, identical segments can be identified efficiently.
3. The maximal unique matches can be identified by finding all local maxima in the LCP array. Specifically, if the i^{th} entry of the LCP array exceeds a certain threshold, it is viewed as a local maximum. In addition, the i^{th} and the $(i-1)^{th}$ entry of the BWT array are compared to ensure that the match is not contained by a longer match.

4. EXPERIMENTAL RESULTS AND DISCUSSION

4.1. Experimental Setup

The video database used in our experiments is from MUSCLE-VCD-2007 [7]. The database contains 101 video clips of a total length equal to about 80 hours. These video clips come from several different sources such as the web video clips, TV programs and movies, and they cover a wide range of categories, including movies, commercials, sports games, cartoons, etc. They were transcoded into the MPEG-1 format at 1.15 Mbps.

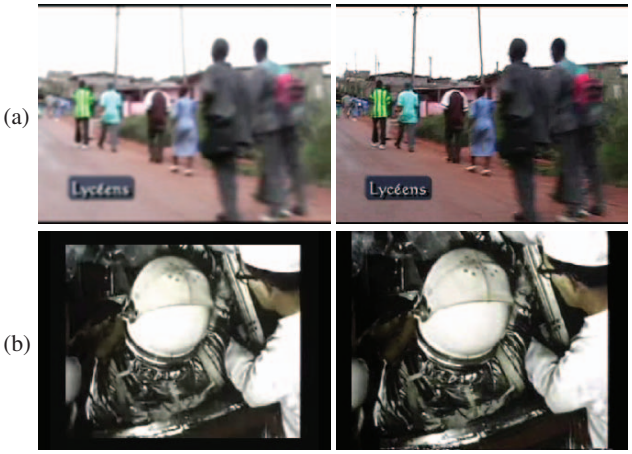

Fig. 2. Query video clips and their ground truths: (a) Color adjustment and blurring and (b) Reencoding, color adjustment, and cropping.

There are 25 query video clips with a total length a little more than 7 hours. Among them, 15 query video clips were attacked by one or more changes, while the remaining 10 video clips do not come from the database. The lengths of these queries vary from 6 minutes short video clip to 1 hour TV program. The attacks applied to these query video clips include different types of color adjustment, blurring, re-encoding with low bit-rate, cropping, subtitle insertion, camcording, zooming, and resizing. Fig. 2 shows two of the query video clips side-by-side with their corresponding ground truths.

In the process of finding matched shots, we only consider a segment of more than 3 consecutive shots that have the same shot lengths as a true positive. The reason to set this threshold is that subsampling video temporally has a similar effect as quantizing shot lengths. Some video clips could happen to have, say, 2 consecutive shots with the same lengths, but the content is actually different. The percentage of matched shots are computed. If it is high enough, the pair of video under consideration is marked as a duplicate one.

4.2. Experimental Results

Table 1 lists matching results of all query video clips with their corresponding ground truth video clips. The results are shown as the percentage of matched shots, which is calculated according to the signature length of the query video. That is, a 100% match means all anchor frames of the query video clip match those of the target video clip in the database in terms lengths between two adjacent anchor frames.

In our experiment, when the query video is compared with video clips in the database other than the ground truth, the reported percentage of match is always less than 1%. The query video has a high-percentage match only when it is compared against its corresponding ground truth. Table 1 shows the match percentages of the 15 queries. Eight of them have 70% to 100% matches, while four have only 53% to 62%. On the other hand, the 10 query video clips

Table 1. Results of matching percentages with the query video and its ground truth.

Query	Ground Truth	Max Match Percentages
Query1	movie27	53.2%
Query3	movie8	87.5%
Query5	movie44	80%
Query6	movie76	0%
Query9	movie9	81.35%
Query10	movie21	0%
Query11	movie37	0%
Query13	movie11	100%
Query14	movie17	69.68%
Query15	movie68	98%
Query16	movie13	74.67%
Query18	movie39	56.48%
Query19	movie52	61.84%
Query22	movie78	62.88%
Query25	movie83	81.81%

that do not have their ground truths in the database all have 0% or 1% to 3% matches. Thus, we can claim the detection of duplicates/non-duplicates is successful for the 22 query clips.

However, there are three query videos that are duplicates of some videos in the database, but have 0% match when compared with their corresponding ground truths. They are Query6, Query10, and Query11. These 3 query video clips have several common properties. They either contain very few shot boundaries, only gradual transitional effects, or a lot of camera/object motions. There is no easily identifiable event that can be marked as anchor frames. This is the main reason why the detection failed on these three videos. This is the limitation of the proposed algorithm.

4.3. Storage Space and Computational Complexity

Since we use the enhanced suffix array to find all maximal unique matches, the computational complexity is $O(n)$, proportional to the length of the signature. Note that the length of the signature is the same as the number of anchor frames, and the set of all anchor frames is only a subset of the temporally subsampled frames. This means that the compact signature not only saves the storage space, but also reduces the computational complexity drastically. The size of the original database is about 35GB, while signatures for all video clips in the database only take 1.2MB, which is about 0.33% of the original database size.

To demonstrate the speed of the proposed video copy detection system, the run time information on this data set is obtained using the Linux command *time* on a computer with 2.16GHz CPU and 2GB memory. Table 2 summarizes the time needed for extracting anchor frames for the entire database and the query set, and the time for signature comparison. For a video database of length about 80 hours, roughly 1 hour is needed in extracting all anchor frames, and less than 7 minutes are needed in extracting anchor frames for all query video clips ($\sim$7 hours). Note that the power of the proposed suffix-array-based approach actually lies in efficient signature matching. Only 2 minutes are needed for signature comparison of all the 2,525 video pairs.

Table 2. Time needed for different tasks.

Task	# of Sub-task	Time
Anchor frame extraction (database)	101 videos ($\sim$80hr)	61m56.867s
Anchor frame extraction (query)	25 videos, ($\sim$7hr)	6m38.640s
Signature comparison	$101 \times 25 = 2525$	1m50.124s

5. CONCLUSION AND FUTURE WORK

In this paper, we presented a novel video copy detection algorithm that enables accurate and extremely efficient identification of video copies. The proposed algorithm utilizes the video temporal structure as the signature, which is compact, discriminative yet robust for a large class of video contents. A set of anchor frames that marks the unique events in video is extracted according to the cumulative luminance histogram difference and the statistics collected in a local window. To address the rapid growth of video contents, the proposed system uses an efficient data structure called the suffix array to achieve extremely fast matching of signatures. The matching algorithm can identify all the maximal unique matches in linear time and determine if the video under consideration is a duplicate video from the database. Both the speed and the discriminancy of the proposed methodology were demonstrated by experimental results.

Some weakness of the proposed methodology was also discussed. It does not work well for video contents with lots of camera/object movements and/or gradual transitional effects. To find a good signature for these video clips and develop a fast matching algorithm for them is under our current investigation.

6. REFERENCES

[1] Arun Hampapur, Kiho Hyun, and Ruud M. Bolle, "Comparison of sequence matching techniques for video copy detection," in *Storage and Retrieval for Media Databases 2002*, Minerva M. Yeung, Chung-Sheng Li, and Rainer W. Lienhart, Eds. 2001, vol. 4676, pp. 194–201, SPIE.

[2] S.-S. Cheung and A. Zakhor, "Efficient video similarity measurement with video signature," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, no. 1, pp. 59–74, Jan 2003.

[3] Justin Zobel and Timothy C. Hoad, "Detection of video sequences using compact signatures," *ACM Trans. Inf. Syst.*, vol. 24, no. 1, pp. 1–50, 2006.

[4] Irena Koprinska and Sergio Carrato, "Temporal video segmentation: A survey," *Signal Processing: Image Communication*, vol. 16, no. 5, pp. 477–500, January 2001.

[5] Udi Manber and Gene Myers, "Suffix arrays: a new method for on-line string searches," in *SODA '90: Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms*, Philadelphia, PA, USA, 1990, pp. 319–327, Society for Industrial and Applied Mathematics.

[6] Mohamed Ibrahim Abouelhoda, Stefan Kurtz, and Enno Ohlebusch, "Replacing suffix trees with enhanced suffix arrays," *J. of Discrete Algorithms*, vol. 2, no. 1, pp. 53–86, 2004.

[7] MUSCLE-VCD-2007, http://www-rocq.inria.fr/imedia/civr-bench/index.html.