Bit Rate-based H.264 Video Copy Detection

Jian Li, Yan Kong

Jiangsu Engineering Center of Network Monitoring, School of Computer and Software, Nanjing University of Information Science & Technology, China
lijian_cs@nuist.edu.cn, kongyan4282@163.com

Abstract

In this paper, we propose a bit rate-based copy detection scheme for H.264 compressed video. The video descriptor is extracted from the compressed video domain thus without the need of decoding the stream. We firstly segment the video stream into a series of shots, then organize the bit rate of the P-frames within each shot in time sequencing, and perform low-pass filtering on the bit rate sequence to reduce the noise from video coding, and finally we obtain the video descriptor from the filtered bit rate sequences shot by shot. The duplicated video is identified by measuring the edit distance between the video descriptors. The most obvious advantage of our proposed CBCD scheme is of low computational load under high detection accuracy. Besides, experimental results demonstrate that it is also able to resist most common video operations, such as re-encoding, cropping etc.

Keywords: Video copy detection, H.264, Bit rate-based, Edit distance

1 Introduction

The rapid growth of Internet video applications like YouTube raises two urgent issues, video content management and video copyright protection. One fundamental question about these issues is to identify the videos from the same source while possibly undergoing various different transformations such as re-encoding, scaling etc. There are two kinds of method promising to carry out this task, i.e., digital watermarking and content based copy detection (CBCD). We focus on the latter in this study.

One CBCD scheme normally consists of the following two processes: first extracting the descriptor from the video content, and then identifying the duplication by comparing the query's descriptor to the target's one. In the literature, most proposed video descriptors were extracted from the decoded video frames (referred to as frame-based descriptors) [1-5]. It has been demonstrated that the frame-based descriptors are robust against common video transformations, including resolution change, luminance shift, display format conversions and etc. These methods are mostly based on image processing and pattern recognition techniques [8, 17-18, 20]. Besides frame-based descriptors, Wu *et al.* [6] proposed a descriptor capturing the camera transitional behaviors (CTB), such as shot boundaries and camera panning/tilting. This CTB-based descriptor was rather efficient in terms of descriptor extraction and matching.

Digital videos are commonly stored and distributed in compression format. A video decoding procedure is hence required before extracting the descriptor, which will consume some time and, especially, large memory resources. However, the efficiency of extracting video descriptor is not taken as a serious issue by most existing CBCD schemes, because they consider the time of query descriptor extraction to be insignificant [7]. But it is not the truth in reality. For instance, YouTube has 300 hours of new video uploaded per minute, and this number is still increasing now. So without a compact yet efficient descriptor it will be incredibly time-consuming to find video duplication given such a large number of queries, not to mention constructing descriptor database for the already stored videos. Intuitively, it should be much more efficient to extract the descriptor directly from a compressed video. Although the idea emerged as early as MPEG-2 era, these schemes [9-10, 12] cannot be directly applied to the new video coding standard, i.e. H.264, which is widely used now.

In this paper, a novel video copy detection scheme for H.264 video is proposed. We firstly segment the video into a series of shots by analyzing the video stream. Then the video descriptor is extracted from all of these shots. A frame recording the complex or fast moving scene usually needs more bits for coding. That is the frame bit rate roughly reflects the information quantity of one frame. And the bit rate sequence of the frames in one shot is closely related to the video content. However, the obtained bit rate is also dependent on the coding setup. We take the influence from video coding as noise added to the information quantity. Therefore we apply low-pass filtering on the

[*]Corresponding Author: Jian Li; E-mail: lijian_cs@nuist.edu.cn

DOI: 10.3966/160792642018011901009

bit rate sequences to remove the noise from video coding, and obtain the video descriptor from the filtered bit rate sequences. In the descriptor matching process, editor distance [4, 13] is employed to estimate the similarity between the query and the target. The experimental results demonstrate that the proposed video descriptor is resistant to common video transformations, such as cropping, scaling and re-encoding.

The rest of this paper is organized as follows. Section 2 provides a framework of our proposed video copy detection scheme. The descriptor extraction and matching algorithms are introduced in Section 3. We present the experimental results in Section 4, followed by the conclusion in Section 5.

2 Framework of The CBCD Scheme in Compressed Domain

Figure 1 illustrates a prototype framework of the CBCD system for compressed video. Because the input query could be compressed by any video compression standard, we need to identify the query's compression format first, and then select the algorithms of descriptor extraction and matching accordingly. The descriptors of the database videos are extracted previously for all the considered compression standards, and are then stored in separate descriptor databases. Please note that the construction of different kinds of descriptor databases for various compression standards only increase the offline descriptor extraction time, while the online efficiency regarding extraction of descriptor of query video and descriptor matching process is not influenced. Therefore, by cooperating with the schemes designed for other compression standards, our scheme can be efficiently used in an online video repository system for copy detection.

3 Descriptor Extractions and Matching Algorithm

In this section we will introduce the video descriptor extraction and matching algorithms. From Subsection 3.1 to 3.3 we will detail the three components involved descriptor extraction, namely "Segmentation", "Alignment", 'Filtering and Normalization' in Figure 1. The descriptor matching algorithm will be given in Subsection 3.4.

3.1 Video Stream Segmentation

Nowadays the H.264 codecs are capable of detecting scene change in a video stream for timely inserting the key frame there (aka instantaneous decoder refresh (IDR) frame). Hence, we can directly make use of this detection result from encoder. However, there also exists another kind of key frame being automatically inserted at a certain time interval, say, ten seconds 2. The second kind of key frame serves as the random access (or seeking) points of video stream. These two kinds of key frames are illustrated in Figure 2. We can identify the second kind of key frame easily since they are of a same distance apart from their immediately prior key frames, in particular, the distance is the largest interval between two adjacent key frames. After removing the second kind of key frame, the video stream is segmented into a sequence of shot by the key frames associated with scene change. For example, the first, the second and the fourth key frames in Figure 2 define two shots. One may concern that video encoder is unable to detect the scene change consistently. Figure 3 presents a shot change detection result from x264, one popular H.264 encoder, with different setting. We can observe that the shots detected by encoder are mostly the same to each other regardless of encoding setup. Furthermore, the experimental results presented in Section 4 also demonstrate the robustness of the scene change result of the H.264 encoder.

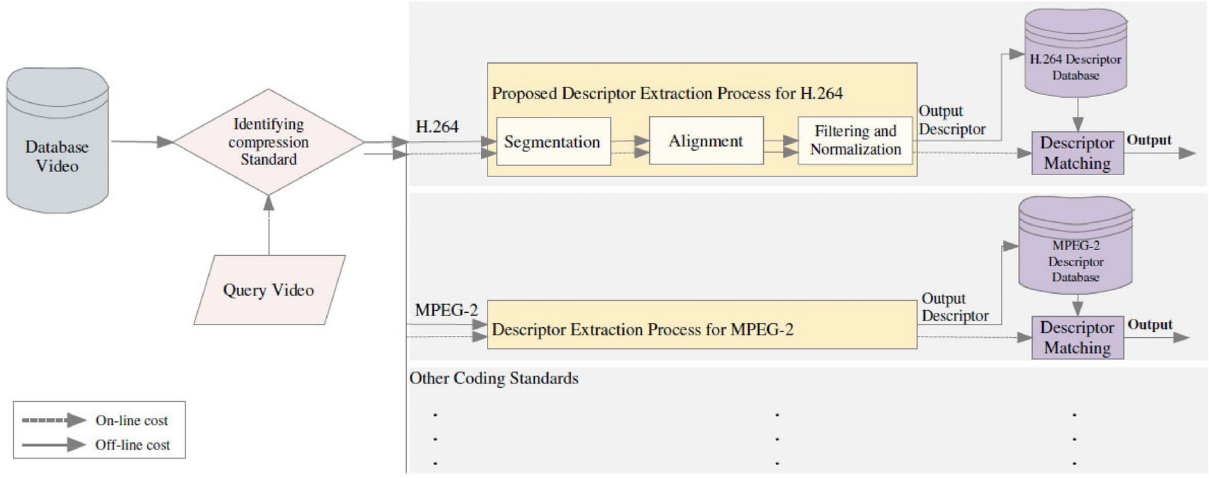
Database Video
Identifying compression Standard
Query Video
H.264
Proposed Descriptor Extraction Process for H.264
Segmentation
Alignment
Filtering and Normalization
Output Descriptor
H.264 Descriptor Database
Descriptor Matching
Output
MPEG-2
Descriptor Extraction Process for MPEG-2
Output Descriptor
MPEG-2 Descriptor Database
Descriptor Matching
Output
Other Coding Standards
On-line cost
Off-line cost

Figure 1. Our proposed video copy detection scheme, which can be cooperated with other kinds compressed domain scheme [8] for detecting different format of videos

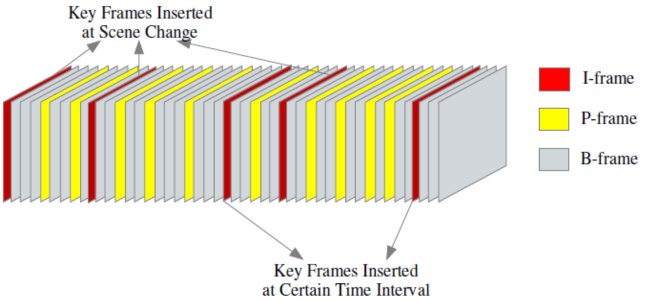

Figure 2. The key frames in one video clip (referred to as I-frame in the figure). Please note that the distance between the second and the third key frames is the same as the distance between the fourth and the fifth key frames. It is because the third and the fifth key frames are inserted at a same time interval

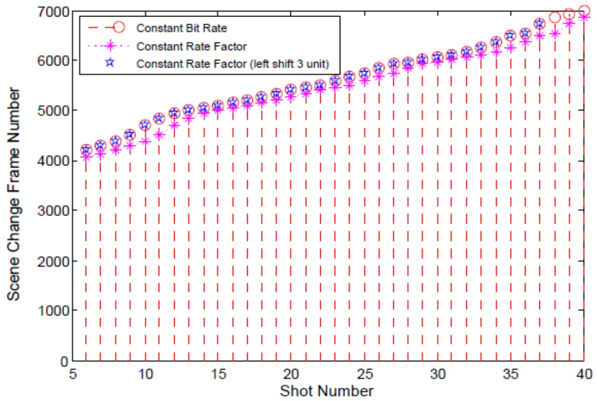

Figure 3. Comparison of shot change detection results of x264 with different coding setups, namely CBR (constant bit rate) mode and CRF (constant rate factor) mode. The video is clipped from a long movie. The symbols "*" and "∘" are used to indicate the occurrence of shot change. Specifically, the x-coordinate of the symbol is the number of the shot, while the y-coordinate is the number of the frame where shot change occurs. Note that the detection results associated with CRF appears not aligned with that of CBR because of a few detection error in the preceding shots. However, after making a left shift for the CRF curve the two sets are mostly matched

3.2 P-Frame Alignment

We propose to use the bit rate evolution of the P-frame as the feature of a video shot. It is based on the following two reasons why we adopt P-frame. First, unlike the B-frame that is not included in the H.264 Baseline videos, the P-frames definitely exist in every single H.264 video; Second, unlike the IDR-frame that infrequently exists in video stream, the P-frames spread throughout a video shot and construct a skeleton of the video. However, we need to align the P-frames to solve the following two problems. One is that the quantization parameter (QP) of the P-frames may be different to each other. For instance, to code the video with CBR mode, the QP is varied with frames, i.e., the frame with more information is encoded with larger QP. So we need to get rid of the influence from QP, in order that the bit rate can approximately reflect the information entropy of the frame. As mentioned in [15], an increase of 1 in quantization parameter means the increase of quantization step by approximately 12%. Moreover, a change of the quantization step by 12% means an reduction of video bit rate by approximately 12%. Based on this rule, we are able to obtain the bit rate of all P-frames with respect to a same QP (benchmark). In particular, both the average bit rate of all the P-frames within a shot and the bit rate of the first P-frame can be used as the benchmark.

Besides, another problem is that the P-frames in video stream may vary with the number of B-frame. As shown in Figure 4, the number of B-frame inserted between P-frames is dependent on the video content and coding setup. To solvethis problem, we need to arrange the P-frames in time order. POC (picture order count) is a parameter written into video stream to record the time order of each frame. With the help of POC, we average out the bit rate of P-frames within a certain frame interval in order that the obtained bit rate sequence is arranged in time order. The frame interval must be larger than the maximum number of inserted B-frame. In real applications it is seldom to insert more than there B-frames between I- or P-frames. Hence, five is big enough to be as our frame interval. Figure 5 presents an example of normalizing the P-frames.

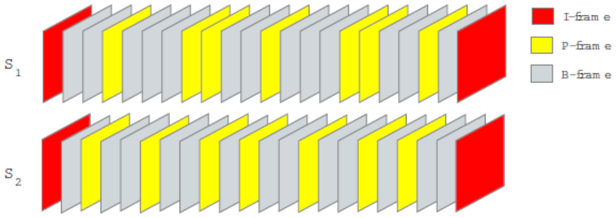

Figure 4. A same video shot with different B-frame setting. There are at most three B-frames inserted between I-/P-frames in the S₁ sequence, while at most two B-frames inserted between I-/P-frames in the S₂ sequence. The largest number of B-frame that can be inserted is determined by encoder

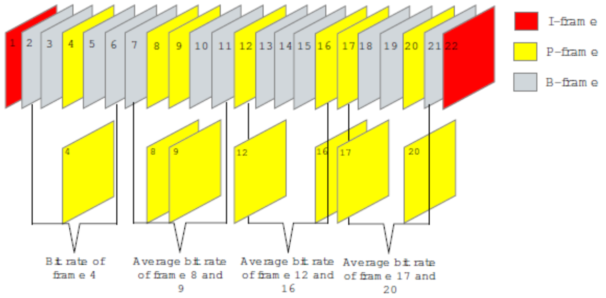
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
I-fram e
P-fram e
B-fram e
4
8
9
12
16
17
20
Bit rate of fram e 4
Average bit rate of fram e 8 and 9
Average bit rate of fram e 12 and 16
Average bit rate of fram e 17 and 20

Figure 5. Example of P-frame alignment. We obtain a value that is the average bit rate of the P-frames in each 5-frame interval

3.3 Low-pass Filtering and Normalizing

It is noted that the sequence of bit rate of the P-frames also contains noise resulted from video coding. The noise is introduced by various reasons, e.g., different inter-prediction algorithms or different coding setup. We need to smooth away this noise in order that the obtained video descriptor is mostly dependent on the video content. A direct solution is low-pass filtering given that the noise is usually located in the high frequency band. In our implementation, a Gaussian-kernel filter is employed to reduce such a kind of noise. Finally, for the sake of the subsequent matching process, the filtered bit rate sequence of the i^{th} shot is sub-sampled to a L dimensional vector s_i. We suggest setting the L at 32 considering the trade-off between descriptor distinctiveness and robustness. In order to achieve robustness against some attacks like re-encoding, the s_i is normalized as follows,

$$\vec{d}_i = \frac{\vec{s}_i}{sum(\vec{s}_i)} \tag{1}$$

where $\vec{d}$ is the descriptor of a video shot. The video escriptor is obtained by concatenating the vectors $\vec{d}$ of all the shots.

3.4 Descriptor Matching

We firstly discuss how to measure the similarity between two shot vectors, based on which we can calculate the edit distance between two video descriptors. As our shot vector is a histogram-like feature, it is suitable to apply χ^2 test statistic to similarity measure as in [4]. Furthermore, it is observed that the relationship between two adjacent vector elements remains unchanged after video transformation. So we propose to integrate this characteristic into the χ^2 test statistic. Suppose $\vec{x}$ and $\vec{y}$ are two shot vectors, the modified χ^2 test statistic $C(\vec{x},\vec{y})$ is given by

$$C(\vec{x},\vec{y}) = \frac{1}{2}\sum_{i=1}^{32}\frac{w_i \bullet (x_i - y_i)^2}{x_i + y_i} \tag{2}$$

where parameter wi represents the relationship between the elements of two vectors, that is

$$w_i = \begin{cases} \text{-1} & \text{if} \quad (x_i - x_{i-1})\cdot(y_i - y_{i-1}) < 0, \\ 1 & \text{if} \quad (x_i - x_{i-1})\cdot(y_i - y_{i-1}) \geq 0. \end{cases} \tag{3}$$

Eq. (3) indicates that we do not consider the difference between xi and yi if they are both bigger (or smaller) than their preceding ones. A threshold is needed to define two different shots. In our implementation, the threshold is set at 0.008. That is two shots are regarded as from a same source if their modified χ^2 test statistic (2) is smaller than 0.008.

The similarity between two video descriptors can be measured by editor distance which has been widely used for string matching and video copy detection [4, 13]. As defined in [13], the edit distance between two strings, s1 and s2, is the minimal cost of a sequence of operations that transfer s1 into s2. The operations are generally restricted to insertion, deletion and substitution. It is natural to apply such an idea to match two video descriptors consisting of equal sized vectors (these vectors are like the characters in the string). Assume v1 and v2 are two arbitrary video descriptors and the length of v1 is smaller than v2. The matching process is to calculate the smallest edit distance between v1 and v2 (including the subset of v2). Considering that the resulted edit distance is no bigger than the length of v1, we normalize the edit distance by dividing it by the length of v1.

4 Experimental Results

4.1 Database and Query Video Setup

The database used in experimental works consists of a number of videos with different duration (from 5min to 1.5h, totally about 40 hours) and resolution (from 320×240 to 1280×720). All the database videos are compressed by H.264 encoder x264 with Baseline Profile, CRF 23. We have 8 hours of query video, half of which are randomly selected from the database, namely positive queries. The positive queries are also transformed with various operations that will be detailed below. The other query videos are negative queries.

4.2 Robustness and Distinctiveness Test

We first test the descriptor's robustness against some common video operations, namely re-encoding (H.264 Baseline profile, three-fold reduction in bit rate; H.264 High profile, two-fold reduction in bit rate), cropping (25% in both horizontal and vertical resolution), scaling (four-fold reduction in size), flipping (horizontally and vertically), sharping and blurring (ffmpeg video filter, luma amount=-2, 1). A normal user can perform these operations easily by a video tool like famous FFMPEG3. In Figure 6, we show the edit distance between each transformed positive query and its corresponding ground truth. Besides, in order to verify the distinctiveness of the descriptor the edit distances between all negative query and the database videos are also plotted. From the exper-imental results it is observed that the edit distances corresponding to the negative queries are mostly larger than 0.4. While the edit distances corresponding to the positive queries are mostly smaller than 0.4. Hence, we can use 0.4 as the threshold to differentiate the negative and positive queries, and achieve a comparable result with [19] in terms of recall-precision rate. It is worth noting that we only employ one simple feature while seven features are involved in [19]. In addition to the aforementioned common video operations, we also test the descriptor's robustness against other transformations, like AWGN and camcording. This kind of transformation is able to greatly vary the video bit rate, and thus may influence the robustness of our proposed descriptor. Experimental results demonstrate that our scheme is able to resist AWGN to a certain extent (less than 15dB), but fails in camcording test. This performance is comparable to the traditional video copy detection schemes [14, 16], but is not as good as the most advanced ones [6-7]. We also speculate that all the features extracted from video compressed domain easily suffer from the attacks that introduce a large amounts of noise (such as camcording), considering the energy of noise could be much larger than that of the signals in compressed video. How to improve the scheme robustness against the attacks like AWGN and camcording is thus one of our future work.

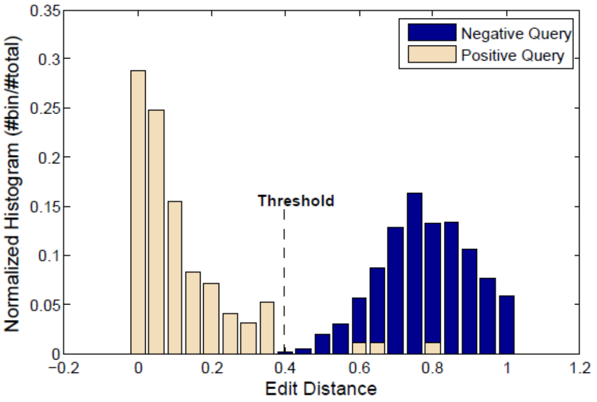

Figure 6. The statistical results of the edit distance between a negative query and a database video are shown by blue bins, while the buff bins indicate the statistics of the edit distance between the positive query and the corresponding ground truth in database. The threshold can be simply set at 0.4 to differentiate them

5 Conclusion and Discussion

In this paper, a CBCD scheme for H.264 compressed video has been proposed. It may be used in the domains of forensics [11], copyright protection etc. We first segment the video stream into a series of shots. Then we construct a bit rate sequence for each shot using its contained P-frames. Finally, the descriptor is extracted from the low-pass filtered bit rate sequence. The descriptor matching algorithm is based on the edit distance with proper modification. Apparently, our proposed CBCD scheme requires rather limited computational power owing to the efficient descriptor extraction algorithm. The bit rate of one P-frame is not robust, while with the help of low-pass filtering, the bit rate variation of the P-frames approximately reflects the content of the video, and thus is able to achieve robustness against common video operations. Experimental results also have confirmed our claim.

We also note that our proposed CBCD scheme cannot completely be independent of the codec setting. For instance, an adversary may disable the scene change detection when coding the video, which will possibly vitiate the proposed scheme. However, in real applications, the query video uploaded to a repository website will be subsequently transcoded by a codec in order to fit the display requirements. We can perform the descriptor extraction process after transcoding the video, or more economically, using the bit rate information from the encoding process. As long as website uses a same video codec as the one associated with descriptor extraction, the influence from cdoec setting can be removed.

Acknowledgement

This work was supported in part by the Natural Science Foundation of Jiangsu Province, China under Grant BK20141006, and in part by the Nat-ural Science Foundation of the Universities in Jiangsu Province under Grant 14KJB520024, and in part by the Natural Science Foundation of China under Grants 61502241, and in part by by the Priority Academic Program Develop-ment of Jiangsu Higer Education Institutionsi, and in part by the Nanjing Uni-versity of Information Science & Technology research foundation 2015r050.

References

[1] L. Chen, F. W. M. Stentiford, Video Sequence Matching Based on Temporal Ordinal Measurement, *Pattern Recognation Letters*, Vol. 29, No. 13, pp. 1824-1831, October, 2008.

[2] C.-Y. Chiu, H.-M. Wang, C.-S. Chen, Fast min-hashing Indexing and Robust Spatio-temporal Matching for Detecting Video Copies, *ACM Transactions on Multimedia Computing, Communications, and Applications*, Vol. 6, No. 2, pp. 1-23, March, 2010.

[3] J. Law-To, L. Chen, A. Joly, I. Laptev, O. Buisson, V. Gouet-Brunet, N. Boujemaa, F. Stentiford, Video Copy Detection: A Comparative Study, *ACM International Conference on Image and Video Retrieval*, Amsterdam, The Netherlands, 2007, pp. 371-378.

[4] M.-C. Yeh, K.-T. Cheng, A Compact, Effective Descriptor for Video Copy Detection, *17th ACM International Conference on Multimedia*, Beijing, China, 2009, pp. 633-636.

[5] C. Kim, B. Vasudev, Spatiotemporal Sequence Matching for Efficient Video Copy Detection, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 15, No. 1, pp. 127 -132, January, 2005.

[6] P. H. Wu, T. Thaipanich, C. C. J. Kuo, Detecting Duplicate Video Based on Camera Transitional Behavior, *2009 16th IEEE International Conference on Image Processing (ICIP)*,

Cairo, Egypt, 2009, pp. 237-240.

[7] M. Douze, H. Jegou, C. Schmid, P. Perez, Compact video Description with Precise Temporal Alignment, *11th European Conference on Computer Vision*, Crete, Greece, 2010, pp. 522-535.

[8] Z. Xia, X. Wang, X. Sun, Q. Wang, A Secure and Dynamic Multi-keyword Ranked Search Scheme over Encrypted Cloud Data, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 27, No. 2, pp. 340-352, February, 2016.

[9] V. Kobla, D. S. Doermann, K. Lin, C. Faloutsos, Compressed Domain Video Indexing Techniques Using Dct and Motion Vector Information in Mpeg Video, *SPIE Conference on Storage and Retrieval for Image and Video Databases V*, San Jose, CA, 1997, pp. 200-211.

[10] H. Yi, D. Rajan, L.T. Chia, A Motion-based Scene Tree for Browsing and Retrieval of Compressed Videos, *Information Systems*, Vol. 31, No. 7, pp. 638-658, November, 2006.

[11] J. Li, X. Li, B. Yang, X. Sun, Segmentation-based Image Copy-move Forgery Detection Scheme, *IEEE Transactions on Information Forensics and Security*, Vol. 10, No. 3, pp. 507-518, March, 2015.

[12] H. Wang, A. Divakaran, A. Vetro, S.-F. Chang, H. Sun, Survey of Compressed-domain Features Used in Audio-visual Indexing and Analysis, *Journal of Visual Communication and Image Representation*, Vol. 14, No. 2, pp. 150-183, June, 2003.

[13] G. Navarro, A Guided Tour to Approximate String Matching, *ACM Computing Surveys*, Vol. 33, No. 1, pp. 31-88, March, 2001.

[14] X.-S. Hua, X. Chen, H.-J. Zhang, Robust Video Signature Based on Ordinal Measure, *2004 IEEE International Conference on Image Processing (ICIP)*, Singapore, 2004, pp. 685-688.

[15] T. Wiegand, G. J. Sullivan, G. Bjontegaard, A. Luthra, Overview of the h.264/avc Video Coding Standard, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 13, No. 7, pp. 560 -576, July, 2003.

[16] L. Chen, F. W. M. Stentiford, Video Sequence Matching Based on Temporal Ordinal Measurement, *Pattern Recognition Letters*, Vol. 29, No. 13, pp. 1824-1831, October, 2008.

[17] B. Gu, V. S. Sheng, K. Y. Tay, W. Romano, S. Li, Incremental Support Vector Learning for Ordinal Regression, *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 26, No. 7, pp. 1403-1416, July, 2015.

[18] Z. Fu, X. Sun, Q. Liu, L. Zhou, J. Shu, Achieving Efficient Cloud Search Services: Multi-keyword Ranked Search over Encrypted Cloud Data Supporting Parallel Computing, *IEICE Transactions on Communications*, Vol. E98-B, No. 1, pp. 190-200, January, 2015.

[19] C. Kas, H. Nicolas, Compressed Domain Copy Detection of Scalable SVC Videos, *2009 Seventh International Workshop on Content-Based Multimedia Indexing*, Chania, Greece, 2009, pp. 89-94.

[20] T. Ma, J. Zhou, M. Tang, Y. Tian, A. Al-Dhelaan, M. Al-Rodhaan, S. Lee, Social Network and Tag Sources Based Augmenting Collaborative Recommender System, *IEICE Transactions on Information and Systems*, Vol. E98-D, No. 4, pp. 902-910, April, 2015.

Biographies

Jian Li is currently a Lecture in the School of Computer and Software at Nanjing University of Information Science & Technology, China. He received the B.S. and M.S. degrees from Shandong University, China, and the Ph.D. degree from Sun Yat-Sen University, China, all in computer science, in 2004, 2007 and 2011 respectively. His research interests are information hiding and forensics.

Yan Kong is currently a Lecture in the School of Computer and Software at Nanjing University of Information Science & Technology, China. She received the Ph.D degree from the University of Wollongong, Australia, in 2015. Her research interests include cloud computing, artificial intelligence, and big data.