Multimed Tools Appl
https://doi.org/10.1007/s11042-018-5862-3

Multiscale video sequence matching for near-duplicate detection and retrieval

Yuanyuan Yang[1] · Yonghong Tian[2] · Tiejun Huang[2]

Received: 27 July 2017 / Revised: 24 November 2017 / Accepted: 1 March 2018

Abstract As one of key technologies in content-based near-duplicate detection and video retrieval, video sequence matching can be used to judge whether two videos exist duplicate or near-duplicate segments or not. Despite a lot of research efforts devoted in recent years, how to precisely and efficiently perform sequence matching among videos (which may be subject to complex audio-visual transformations) from a large-scale database still remains a pretty challenging task. To address this problem, this paper proposes a multiscale video sequence matching (MS-VSM) method, which can gradually detect and locate the similar segments between videos from coarse to fine scales. At the coarse scale, it makes use of the Maximum Weight Matching (MWM) algorithm to rapidly select several candidate reference videos from the database for a given query. Then for each candidate video, its most similar segment with respect to the given query is obtained at the middle scale by the Constrained Longest Ascending Matching Subsequence (CLAMS) algorithm, and then can be used to judge whether that candidate exists near-duplicate or not. If so, the precise locations of the near-duplicate segments in both query and reference videos are determined at the fine scale by using bi-directional scanning to check the matching similarity at the segments' boundaries. As such, the MS-VSM method can achieve excellent near-duplicate detection accuracy and localization precision with a very high processing efficiency. Extensive experiments show that it outperforms several state-of-the-art methods remarkably on several benchmarks.

Keywords Near-duplicate video detection · Video sequence matching · Multiscale matching · Constrained longest ascending matching subsequence · Bi-directional scanning

✉ Yuanyuan Yang
yyyang0518@126.com

Yonghong Tian
yhtian@pku.edu.cn

[1] Beijing University of Posts and Telecommunications, Beijing, China

[2] Peking University, Beijing, China

Published online: 04 May 2018

Every day, a large number of videos are created and transmitted over the network. People can obtain the source files of the web videos handily and release the edited versions to the Internet. This gives rise to the existence of a noticeable amount of copies or near-duplicate videos on the Internet. Accordingly, it imposes urgent demands on video near-duplicate detection as a key role in many tasks and applications such as video search, content filtering, copyright protection, video usage monitoring, movie recommendation and video advertising [14, 25]. In content-based near-duplicate detection, one key issue is how to precisely and efficiently judge whether two videos exist duplicate or near-duplicate segments and where are their starting and ending timestamps [23]. This task, often referred to as *video sequence matching*, is very challenging since the duplicate or near-duplicate segments may be generated from the origins by means of various audio-visual transformations. To make things worse, the content of many copies may be significantly changed from their origins. To address this challenge, various video sequence matching methods and techniques have been proposed in recent years.

Basically, existing works on video sequence matching can be divided into three categories according to the different matching granularities: global-level matching, segment-level matching and frame-level matching. Global-level matching (e.g., [6, 16, 38]) considers the query video (denoted by Q) as an entirety and attempts to find the near-duplicate segments by iteratively scanning over all the possible subsequences in the reference videos (denoted by $\mathbb{R}$). Here an implicit assumption is that the whole query video Q is or is not a duplicate. For example, the work [6] transformed the video sequence to a normalized size, then the similarity can be calculated by computing Hamming distance between video hash values. The drawback of these global-level matching algorithms is that they cannot cope with the case that one segment of Q is a duplicate of a small segment in the reference video $R \in \mathbb{R}$. Instead, segment-level matching (e.g., [30, 51]) partitions the videos into several clips according to a fixed length or in terms of shots. Then the similar subsequences can be found according to the similarity between each query clip and each reference clip. However, this approach is still difficult to deal with the case when temporal transformations such as frames inserting or deleting are involved in generating the duplicate or copy. Thus a more flexible paradigm, frame-level matching (e.g., [23, 42]), is to partition Q and $\forall R \in \mathbb{R}$ into audio-visual frames and then make use of the matching relationship between the frames or some keyframes to determine the near-duplicate parts between two videos. Obviously, frame-level matching methods can achieve much better performance on near-duplicate detection and localization. However, because each frame or keyframe in Q and R has to be involved in the matching calculation, the frame similarity evaluation is computationally high. That is, suppose L_Q and L_R denote the average numbers of frames (or keyframes) in a query video and a reference video, $|\mathbb{R}|$ denotes the size of the reference database, then the computational complexity of frame-level matching is $\mathcal{O}(|\mathbb{R}|L_Q L_R)$ for a given query Q. This causes a very high overhead for a large, continuously expanding reference database.

To accelerate frame-level matching, various frame-fusion methods (e.g., [11, 23, 27, 41, 42, 46]) have been proposed. Often, these methods firstly search a list of similar reference frames for each query frame, and then determine video matches by assembling the frame matches with proper temporal fusing strategy. The main difference between them is how to utilize temporal consistency constraints on frame matches to identify several video matches. Typical models include 2D Hough Transform [27], spatio-temporal verification [11], Viterbi-based frame fusion [42], approximate string matching [46], temporal pyramid matching (TPM) [41], and frame matching-result graph [23]. Generally speaking, these methods show promising performance on different benchmark datasets. However, how to enable high-efficient matching in a very large reference database meanwhile maintaining a

high performance on near-duplicate detection and localization is still a difficult topic. This even becomes more challenging nowadays since Internet is actually such a continuously expanding database with the unprecedented amount of videos.

Toward this end, this paper proposes a multiscale video sequence matching (MS-VSM) method, which can gradually detect and locate the similar segments between videos from coarse to fine scales. Our MS-VSM method still follows the frame-fusion framework and performs three scales of matching: At the coarse scale, it makes use of the Maximum Weight Matching (MWM) algorithm to rapidly select the candidate reference videos from the video database for a given query; Then for each candidate video, its *most similar* segment with respect to the given query is obtained at the middle scale by the Constrained Longest Ascending Matching Subsequence (CLAMS) algorithm, and then can be used to judge whether that candidate video exists near-duplicate or not; Finally for the asserted case, the precise locations of the near-duplicate segments in both query and reference videos are determined at the fine scale by using bi-directional scanning to check the matching similarity at the segments' boundaries. In this manner, the MS-VSM method can achieve excellent near-duplicate detection accuracy and localization precision with a very high processing efficiency.

Extensive experiments were conducted on two benchmark datasets, including MUSCLE-VCD-2007 [20] and CC_WEB_VIDEO [43]. Several state-of-the-art methods were used for comparison. Experimental results show that the proposed MS-VSM method outperforms these state-of-the-art methods remarkably on several benchmarks.

The remainder of this paper is organized as follows: We first briefly review the related work for video sequence matching in Section 2. Section 3 describes the overall framework and the algorithmic details of the proposed MS-VSM method. Extensive experiments are presented in Section 4, and finally we conclude the paper in Section 5.

1 Related work

In this section, we briefly review the related work of frame-level video sequence matching in recent years. Basically, the frame-level matching framework is to partition each query video and all reference videos into audio-visual frames and then make use of the matching relationship between the frames or some keyframes to determine the near-duplicate parts between two videos. According to the different problem formulations and the used solving techniques, the existing frame-level matching methods can be roughly divided into five categories: temporal voting, probabilistic model, graph model, dynamic programming, and multiscale model.

Temporal voting is the simplest approach to assemble similar frame pairs between each query and reference video. Its basic idea is to extract a small number of pertinent features from frames (or key frames) in a video and then match them with the database according to a dedicated voting function [19]. In [10, 11], a spatio-temporal verification model was introudced to characterize the matching relationship between the query video and the potentially corresponding video segments. This spatio-temporal model first determines the temporal shift based on 1-D Hough voting, and then determines the spatial component by estimating a 2-D affine transformation between the matching video sequences. Similarly, Liu et al. [27] applied 2D Hough transform to aggregate similarity scores on audio frames from the matched audio segments, and then identified the segments that were fallen into the peak bin as the copy segments in reference video. In [4], a spatio-temporal matching scheme was proposed, by compiling the spatial and temporal information of all the frame pairs from two video sequences as a 2D intensity map and then applying the Hough transform to

search the map for the specific copy patterns. In [21],the largest matching score among all keyframe-level similarities is treated as the video-level similarity.

Instead of the simple temporal voting, the probabilistic matching methods make use of various probabilistic models to represent the matching relationship between videos. Nicolas et al. [12] proposed a probabilistic Markovian framework to fuse a set of similarity searches based on keyframes for identifying near-duplicate segments. In [22], a probabilistic model was proposed for video copy location which was formulated as a likelihood maximization problem. A simplified approach was also presented to reduce the maximization problem into a set of 0-1 value problems based on the indexing structure. Huang et al. [15] represented a video stream as a sequence of compact signatures called linear smoothing functions (LSFs) and then adopted compound probability to combine three independent video factors to effectively measure segment similarity in near-duplicate detection. A joint spatial-temporal sequence alignment framework was formulated in [8] as a maximum a posteriori Bayesian inference problem that would simultaneously satisfy a frame-correspondence and frame-alignment similarity. This problem could be solved by iteratively alternating a min-sum algorithm and a gradient descent algorithm.

Alternatively, the basic idea of graph-based matching methods is to formulate the problem of finding the similar parts between video sequences as a graph model and then utilize different graph theory methods (e.g., longest path, bipartite matching, or dense subgraph) to find the near-duplicate subsequences. In [23], a graph-based method was presented to convert the video sequence matching into finding the longest path in the frame matching-result graph with time constraint. Two bipartite graph matching algorithms, namely maximum matching (MM) and optimal matching (OM), were proposed in [32] for the matching of shots in clips, where MM was used to select candidate videos, while OM was used to calculate the similarity between query and each candidate video. Similarly, in [17], the similarity of two video clips could be measured by means of graph-based similarity measures such as maximal cardinality bipartite matching, which then were used to determine whether a query video and source video are near-duplicates. A graph transformation and matching approach was proposed in [37], in which the mapping relationship between the query and the database video was first represented by a bipartite graph, and then Maximum Size Matching (MSM) was deployed for each subgraph to obtain a smaller set of candidates and Sub-Maximum Similarity Matching (SMSM) was devised to identify the similar subsequences. By exploiting temporal consistency of the matched keyframes between two video clips to construct a graph, the work [3] formulated the task of detecting near-duplicate sub-clips as the problem of finding all the dense subgraphs, which would be solved by the optimization method of graph shift. Similarly, in [50], a weighted keyframe matching graph was constructed to model the matching relationship between the query keyframes and those of video data, and then potential video subsequence candidates could be obtained by extracting a number of maximal matching subgraphs from the matching graph.

Instead of utilizing the probabilistic or graph models, there are also many studies that solve the video sequence matching problem by utilizing dynamic programming algorithms such as Viterbi, dynamic time warping (DTW), Smith-Waterman and string matching. In [42], the frame fusion problem was formulated as the decoding problem of a hidden Markov model and a Viterbi-like dynamic programming algorithm was utilized to perform copy determination and temporal localization, which comprised an online back-tracking strategy with three relaxed constraints (i.e., emission constraint, transition constraint and gap constraint). By representing a video sequence using multiple features, a sliding-window-based DTW algorithm was employed in [35] to compute temporal frame alignments between two sequences. In [45], video copy detection was treated as a local alignment problem

between two frame sequences and then the optimal local alignment could be computed by the Smith-Waterman algorithm. In [46], a video was represented by an ordered list of feature descriptors, and similarities between such representations were then measured by the approximate string matching technique. Liu et al. [24] adopted path matrix to aggregate scores for measuring video similarity localizing the similar parts.

Generally speaking, the above four kinds of frame-level video matching methods have shown promising detection and localization performance on different benchmark datasets. However, with the continuously growing amount of videos on the Internet, the matching efficiency has increasingly become the bottleneck problem. To facilitate high-efficient video sequence matching, multiscale matching methods are attracting more and more attention in recent years. Tian et al. [31, 41] designed a multiscale sequence matching method to assemble frame similarity search results into video-level matches by 2D Hough transform and multi-granularities similarity evaluation. As such, video matching is performed in a pyramid structure, making it computationally efficient. Similarly, Wu et al. [44] presented a coarse-to-fine near-duplicate video localization scheme. In the coarse scale, a fast histogram matching algorithm was used to select the candidate segments; while in the fine scale, the Intensity Mark (IMark) similarity between the query and each candidate segment was calculated to refine the localization result. In [5], an alternative coarse-to-fine video sequence matching strategy was presented, in which the coarse-grain level matching was used to select candidate videos according to the measure of time-decay hit frequency, and then the approximate Hough transform algorithm was applied to identify the near-duplicate subsequences at the fine level. All these results show that, multiscale matching is a possible solution to enable high-efficient matching in a very large reference database. Still following such a framework, this paper proposes a novel multiscale video matching method, which can achieve excellent copy detection accuracy and localization precision with a very high processing efficiency.

2 The Proposed method

This section is to present our multiscale video sequence matching (MS-VSM) method. Toward this end, we first formulate the sequence matching problem from multiple scales, and then describe the algorithmic details of the MS-VSM respectively from the coarse, middle and fine scales. For more readability, Table 1 shows some main notations used in this section.

2.1 Problem formulation

Generally speaking, the task of near-duplicate detection and localization can be stated as follows: given a query video Q and a reference video dataset $\mathbb{R}$, the task is to examine whether there exists a reference subsequence $\{S_R \subseteq R \mid R \in \mathbb{R}\}$ that is the near-duplicate to a query subsequence $S_Q \subseteq Q$, namely, whether $\mathcal{D}(Q, R)$ or more precisely $\mathcal{D}(S_Q, S_R)$ holds, where $S_Q = [t^{(B)}(Q), t^{(E)}(Q)]$ and $S_R = [t^{(B)}(R), t^{(E)}(R)]$, $t^{(B)}(X)$ and $t^{(E)}(X)$ denote the starting and ending positions of a segment in a video X, $\mathcal{D}(X, Y) = 1$ stands for X being a copy of Y by the system (otherwise, $\mathcal{D}(X, Y) = 0$). According to our discussion in Sec. I, if we follow the frame-level video matching paradigm, the computational complexity of frame similarity evaluation should be $\mathcal{O}(\rho|\mathbb{R}|L_Q L_R)$ for a given query Q, where ρ denotes the similarity evaluation time between two frames, L_Q and L_R denote the average numbers of frames (or keyframes) in Q and $R \in \mathbb{R}$, and $|\mathbb{R}|$ is the size of $\mathbb{R}$. Therefore, it is

Table 1 Main notations used in this section

Notation	Meaning
Q, with $\left\{Q^{(\mathrm{C})}, Q^{(\mathrm{M})}, Q^{(\mathrm{F})}\right\}$	A query video, with its coarse, middle and fine-scale representations
$R \in \mathbb{R}$, with $\left\{R^{(\mathrm{C})}, R^{(\mathrm{M})}, R^{(\mathrm{F})}\right\}$	A reference video, with its coarse, middle and fine-scale representations
(S_{Q_j}, S_{R_k})	The matching pair between the j^{th} subsequence of Q and the k^{th} subsequence of R
(T_Q, T_R)	The most similar segments between Q and R, with $T_Q \subseteq S_{Q_j}$ and $T_R \subseteq S_{R_k}$
$\left(\Delta S_Q^{(\mathrm{F})\mp}, \Delta S_R^{(\mathrm{F})\mp}\right)$	The boundary near-duplicate segments before/after $\left(T_Q^{(\mathrm{F})}, T_R^{(\mathrm{F})}\right)$ between Q and R
$Sim\left(X^{(i)}, Y^{(i)}\right)$	The i^{th}-scale segment similarity between $X^{(i)}$ and $Y^{(i)}$, where $i \in \{\mathrm{C}, \mathrm{M}, \mathrm{F}\}$
$\omega(q_s, r_t)$	The frame similarity between $q_s \in Q$ and $r_t \in R$ calculated using frame features
$\mathsf{m}(q_s, r_t)$	The boolean variable to indicate whether $q_s \in Q$ and $r_t \in R$ are a matching pair
$\mathbb{C}_Q = \left\{\left(S_Q^{(\mathrm{C})}, S_R^{(\mathrm{C})}\right)\right\}$	The candidate reference videos for Q constructed at the coarse scale
$\mathcal{D}(X, Y)$	The value of 1 stands that X has a near-duplicate of a segment of Y
$\mathcal{L}(X, Y)$	The matching length between X and Y
$\{\theta_0, \theta_1, \theta_2\}$	The pre-defined similarity thresholds at the coarse, middle and fine scales

desirable to develop a highly efficient and effective matching method for a large reference database.

To begin with our discussion, let's first review one motivated example: Anyone who used eletronic maps or vehicle navigation systems probably has the following experience. If we want to search a route from Stanford University to Google, it is desirable that the

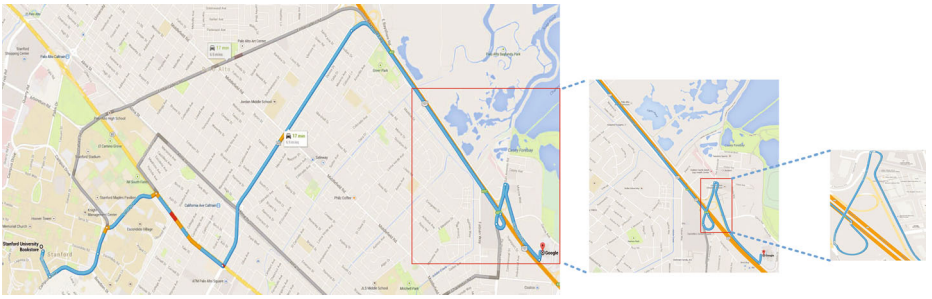

Fig. 1 A multiscale exploration of a route in an electronic map. **a** shows the entire route from Stanford University to Google, **b** is the enlarged view of a specific street in the route, while **c** presents a detailed view of a specific road intersection

navigation system would allow an exploration from an overview of the route, to a local view of a specific street in the route, and even to more details about the left/right turn lane at a road intersection. Figure 1 illustrates such a paradigm. The background represents a macroscopic view of the route in an entire region. This is analogous to viewing the route 10,000m above the ground and one can only watch the general direction and the overall traffic flow in the route. As one zooms in to a local area of the route (e.g., a specific street in the route), a mesoscopic view is obtained. Similar to watching the route 1,000m above the ground, one can view the neighboring blocks around the street (e.g., the neighboring streets, the landmark buildings). Finally, if one focuses on a detailed view of a specific road intersection, a picoscopic view is achieved as if one were operating a vehicle. As such, one can interact with the driving environment (e.g., signs, signals, etc.), make control decisions (e.g., turn right), and manage the vehicle to dynamically respond to travel safely. Overall speaking, such a multiscale representation allows one to explore the different levels of details about the route, e.g., to decompose a whole route down to one or more local segments.

The similar multiscale representation can also be applied to sequence matching. Let X denote a video sequence, then we can derive its multiscale representation with different sampling methods, in which the representer at each scale characterizes a different level of abstraction of X. Figure 2 illustrates such a representation from three scales, denoted by $X^{(\mathrm{F})}$, $X^{(\mathrm{M})}$ and $X^{(\mathrm{C})}$ from fine to coarse, respectively. To derive such a representation, the simplest sampling method is uniform sampling (e.g., at the sampling rates of 10:1, 5:1 and 3:1, respectively). That is, for an one-minute video sequence X with the frame rate of 30 fps, the frame numbers of $X^{(\mathrm{F})}$, $X^{(\mathrm{M})}$ and $X^{(\mathrm{C})}$ are 180, 36, and 12, respectively. A more reasonable method is shot-based sampling. That is, $X^{(\mathrm{F})}$ can be obtained by uniformly sampling X still at the sampling rate of 10:1, but $X^{(\mathrm{M})}$ is constructed by selecting several keyframes for each shot (e.g., 3∼5 frames extracted from each shot, representing its center and boundaries), while $X^{(\mathrm{C})}$ can be derived from $X^{(\mathrm{M})}$ by selecting one representative keyframe for each shot. In this case, we should perform the shot boundary detection over X. For simplicity, this study adopts the uniform sampling method. Our experimental results show that this simple sampling method performs very well.

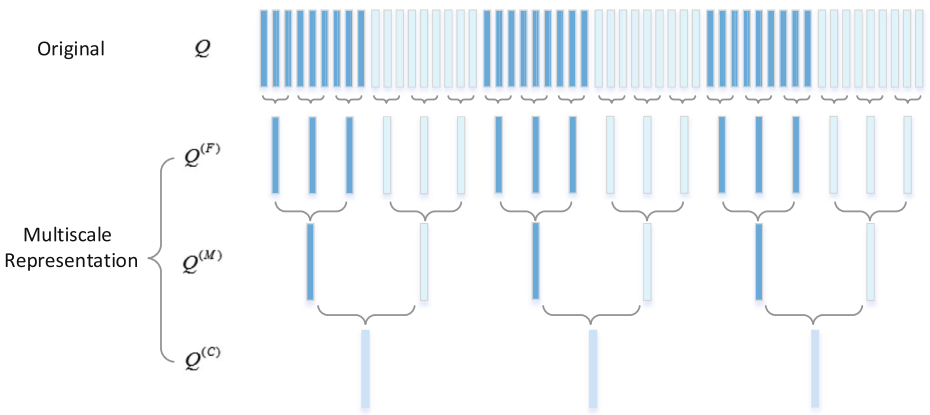

Fig. 2 A multiscale representation of a video for video sequence matching

In practice, such a multiscale representation can offer a significant advantage for video sequence matching. Typically, the coarse representation can enable high near-duplicate detection efficiency but cannot guarantee the accuracy; the situation is quite the contrary for the fine representation. Thus a natural solution is to combine the representations at multiple scales to enable both high effective and efficient video sequence matching. Toward this end, this study is to develop a multiscale video sequence matching (MS-VSM) approach.

One key issue is whether this approach is theoretically feasible, or equivalently, whether the matching results from multiple scales are inherently consistent. The term "consistency" here concerns the coupling among the matching results at different scales. Fortunately, the quotient space theory [47] provides the theoretical foundation for multiscale matching. In this theory, two most important properties are "falsity-preserving" and "truth-preserving". The so-called "falsity-preserving" property says that if a proposition is rejected in the coarse space, it must be false (i.e., without solution) in the finer space; while the "truth-preserving" property indicates that if a proposition is true in coarser-grained space, then under certain conditions, the corresponding problem in the fine space is also true. When applied to our problem, the two properties can guarantee the feasibility of the MS-VSM approach. That is, for a large-scale video matching problem, we can transform it into several equivalent prolems in the coarser-grained spaces so as to reduce the complexity of problem solving. More formally, given a query video Q and a reference video $R \in \mathbb{R}$, if $\mathcal{D}(Q^{(i)}, R^{(i)})$ in two coarse scales hold where $i \in \{\mathrm{C}, \mathrm{M}, \mathrm{F}\}$ (e.g., $\mathcal{D}(Q^{(\mathrm{C})}, R^{(\mathrm{C})})$ and $\mathcal{D}(Q^{(\mathrm{M})}, R^{(\mathrm{M})})$), then we can deduce that $\mathcal{D}(Q, R)$ must be true according to the "truth-preserving" property. On the contrary, if $\mathcal{D}(Q^{(\mathrm{C})}, R^{(\mathrm{C})})$ does not hold, we know immediately that $\mathcal{D}(Q, R)$ is not true according to the "falsity-preserving" property.

Then, the remaining issue is how to perform video sequence matching from multiple scales. In this study, the MS-VSM approach adopts the following steps to solve the video near-duplicate detection and localization task:

(1) At the coarse scale, the MS-VSM rapidly selects the candidate reference videos from the database $\mathbb{R}$ for a given query Q. Formally, let $S_Q^{(\mathrm{C})}$ (or $S_R^{(\mathrm{C})}$) denote a segment of Q (or $R \in \mathbb{R}$) at the coarse scale, the objective here is to construct the following candidate set $\mathbb{C}_Q$:

$$\mathbb{C}_Q = \left\{ \left(S_Q^{(\mathrm{C})}, S_R^{(\mathrm{C})} \right) \mid Sim\left(S_Q^{(\mathrm{C})}, S_R^{(\mathrm{C})} \right) > \theta_0, \; S_Q^{(\mathrm{C})} \subseteq Q^{(\mathrm{C})}, S_R^{(\mathrm{C})} \subseteq R^{(\mathrm{C})}, \forall R \in \mathbb{R} \right\} \tag{1}$$

where $Sim\left(S_Q^{(\mathrm{C})}, S_R^{(\mathrm{C})} \right)$ denotes the similarity between the coarse-scale representations of two segments S_Q and S_R (called *coarse-level similarity*), and θ_0 is a similarity threshold at the coarse scale.

(2) At the middle scale, these candidates are further checked so as to determine whether they are the near-duplicates or not. That is, for $\left(S_Q^{(\mathrm{C})}, S_R^{(\mathrm{C})} \right) \in \mathbb{C}_Q$,

$$\mathcal{D}(Q, R) = \phi\left(S_Q^{(\mathrm{M})}, S_R^{(\mathrm{M})} \right) \tag{2}$$

where $\phi(\cdot, \cdot)$ is a classifier based on the *middle-level similarity* (namely, the similarity between the middle-scale representations of two segments), $\left(S_Q^{(\mathrm{M})}, S_R^{(\mathrm{M})} \right)$ are

the corresponding middle-scale representations of $\left(S_Q^{(\mathrm{C})}, S_R^{(\mathrm{C})}\right)$. The simplest form of $\phi(\cdot,\cdot)$ is

$$\begin{aligned} &\exists T_Q^{(\mathrm{M})} \subseteq S_Q^{(\mathrm{M})}, \exists T_R^{(\mathrm{M})} \subseteq S_R^{(\mathrm{M})}: \\ &Sim\left(T_Q^{(\mathrm{M})}, T_R^{(\mathrm{M})}\right) > \theta_1 \,\&\mathcal{L}\left(T_Q^{(\mathrm{M})}, T_R^{(\mathrm{M})}\right) > \zeta_1 \end{aligned} \tag{3}$$

where $T_Q^{(\mathrm{M})}$ (or $T_R^{(\mathrm{M})}$) is a subsequence of $S_Q^{(\mathrm{M})}$ (accordingly, $S_R^{(\mathrm{M})}$), $\mathcal{L}\left(T_Q^{(\mathrm{M})}, T_R^{(\mathrm{M})}\right)$ denotes the matching length between $T_Q^{(\mathrm{M})}$ and $T_R^{(\mathrm{M})}$, θ_1 is a similarity threshold at the middle scale, and ζ_1 is the pre-defined shortest length for a duplicate segment.[1] That is, if there is a matching subquence pair between $S_Q^{(\mathrm{M})}$ and $S_R^{(\mathrm{M})}$ whose length is more than a pre-defined value and whose similarity is larger than a pre-defined threshold, then $\mathcal{D}(S_Q^{(\mathrm{M})}, S_R^{(\mathrm{M})})$ holds; By combining the two facts that $\mathcal{D}\left(S_Q^{(\mathrm{C})}, S_R^{(\mathrm{C})}\right) = 1$ and $\mathcal{D}(S_Q^{(\mathrm{M})}, S_R^{(\mathrm{M})}) = 1$, $\mathcal{D}(Q, R)$ is naturally true.

(3) If $\mathcal{D}(Q, R) = 1$, the precise locations of the near-duplicate segments in Q and R will be determined at the fine scale.

$$\begin{aligned} &\underset{\substack{\Delta S_Q^{(\mathrm{F})\mp} \subseteq S_Q^{(\mathrm{F})}, \\ \Delta S_R^{(\mathrm{F})\mp} \subseteq S_R^{(\mathrm{F})}}}{\operatorname{argmax}} \mathcal{L}\left(\Delta S_Q^{(\mathrm{F})-}, \Delta S_R^{(\mathrm{F})-}\right) + \mathcal{L}\left(\Delta S_Q^{(\mathrm{F})+}, \Delta S_R^{(\mathrm{F})+}\right) \\ &s.t. Sim\left(\Delta S_Q^{(\mathrm{F})-}, \Delta S_R^{(\mathrm{F})-}\right) \geqslant \theta_2 \& \\ &\qquad Sim\left(\Delta S_Q^{(\mathrm{F})+}, \Delta S_R^{(\mathrm{F})+}\right) \geqslant \theta_2 \end{aligned} \tag{4}$$

where θ_2 is a similarity threshold at the fine scale, $\Delta S_Q^{(\mathrm{F})-}$ and $\Delta S_Q^{(\mathrm{F})+}$ denote the neighboring segments in Q that locate before and after $T_Q^{(\mathrm{F})}$ (i.e., the corresponding fine representation of $T_Q^{(\mathrm{M})}$), and similarly for $\Delta S_R^{(\mathrm{F})-}$ and $\Delta S_R^{(\mathrm{F})+}$. Intuitively, this localization process is to find the longest near-duplicate segments in Q and R that are with the asserted $\left(T_Q^{(\mathrm{F})}, T_R^{(\mathrm{F})}\right)$ as their center. As such, the final near-duplicate segments in Q and R are $\left\{\Delta S_Q^{(\mathrm{F})-} \bowtie T_Q^{(\mathrm{F})} \bowtie \Delta S_Q^{(\mathrm{F})+}\right\}$ and $\left\{\Delta S_R^{(\mathrm{F})-} \bowtie T_R^{(\mathrm{F})} \bowtie \Delta S_R^{(\mathrm{F})+}\right\}$, where the operator $\bowtie$ denotes the sequence concatenation operation.

We can see that given a very large reference database $\mathbb{R}$, only the coarse process involves a large amount of frame similarity calculation (namely, the coarse-level similarity between frames in $Q^{(\mathrm{C})} \subseteq Q$ and $R^{(\mathrm{C})} \subseteq R$ where $\forall R \in \mathbb{R}$). After that, we only need to calculate the middle-level similarity for all frame pairs in $\mathbb{C}_Q$, and if $\mathcal{D}(Q, R)$ holds, the fine-level similarity between frames in $\left(\Delta S_Q^{(\mathrm{F})-}, \Delta S_R^{(\mathrm{F})-}\right)$ and between frames in $\left(\Delta S_Q^{(\mathrm{F})+}, \Delta S_R^{(\mathrm{F})+}\right)$.

[1] Legally, only videos in which the length of identical or similar content is more than a pre-defined value (e.g., 10 seconds) can be treated as duplicates or near-duplicates. According to our sampling method, the frame number of a 10-seconds-video is 6, thus we can set $\zeta_1 = 6$ in our experiments.

Their computational complexity is much less than that in the coarse step and thus can be ignorable. Let $L_Q^{(\mathrm{C})}$ and $L_R^{(\mathrm{C})}$ denote the average numbers of frames in $Q^{(\mathrm{C})}$ and $R^{(\mathrm{C})}$, then the computational complexity of the MS-VSM is approximately $\mathcal{O}\left(\rho|\mathbb{R}|L_Q^{(\mathrm{C})}L_R^{(\mathrm{C})}\right)$. Considering that L_Q and L_R are hundreds of times more than $L_Q^{(\mathrm{C})}$ and $L_R^{(\mathrm{C})}$, the MS-VSM is computationally very efficient. Moreover, the computation can be further accelerated by using the indexing structure of the features (e.g., the inverted table for SIFT Bag-of-Words [28, 29]). In this manner, the MS-VSM is expected to achieve excellent near-duplicate detection accuracy and localization precision with a very high processing efficiency.

Figure 3 shows the flowchart of our MS-VSM. We can see that it consists of three main processes, namely, coarse-scale filtering, middle-scale refinement and fine-scale localization. In the coarse-scale filtering process, the Maximum Weight Matching (MWM) algorithm is used to rapidly filter the irrelevant videos from the reference database so as to obtain the candidates for a given query; in the middle-scale refinement process, the Constrained Longest Ascending Matching Subsequence (CLAMS) algorithm is then adopted to obtain the most similar segments of these candidates, which are then used to make a definite judgement about whether these candidates exist some near-duplicate segments of the given query; while in the fine-scale localization process, bi-directional scanning operations are used to determine the precise boundaries of the near-duplicate segments for those asserted candidates. Thus in the following subsections, we will present the algorithmic details about the three processes.

2.2 Coarse-scale filtering with maximum weight matching

The objective of the coarse-scale filtering process is to rapidly select the candidate reference videos from the database $\mathbb{R}$ for a given query Q. Note that the input data in this process are the coarse-scale representations of Q and $\forall R \in \mathbb{R}$. Here let $Q^{(\mathrm{C})} = \left\{q_1, q_2, \cdots, q_{L_Q^{(\mathrm{C})}}\right\}$ and $R^{(\mathrm{C})} = \left\{r_1, r_2, \cdots, r_{L_R^{(\mathrm{C})}}\right\}$ denote the coarse-scale representations of Q and R, where $L_Q^{(\mathrm{C})}$ and $L_R^{(\mathrm{C})}$ are the numbers of frames in $Q^{(\mathrm{C})}$ and $R^{(\mathrm{C})}$.

Following the frame-level video matching paradigm, the coarse-scale filtering process involves the following two steps:

(1) **Frame similarity evaluation**: This step is to calculate the similarity between every query frame $q_i \in Q^{(\mathrm{C})}$ and every reference frame $r_j \in R^{(\mathrm{C})}$ using some audio-visual features, and then return a list of top-T most similar reference frames (together with their similarity scores) for each q_i. Let $\mathbb{L}^{(\mathrm{C})} = \left\{\mathbb{L}_1, \mathbb{L}_2, \cdots, \mathbb{L}_{L_Q^{(\mathrm{C})}}\right\}$, where $\mathbb{L}_i =$

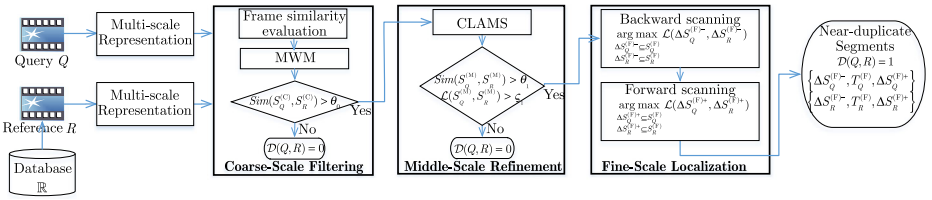

Fig. 3 The flowchart of our MS-VSM approach

$\{r_{i1}, r_{i2}, \cdots, r_{iT}\}$ is the list of similar reference frames linked to q_i ($T = 10$ in our experiments).

As mentioned above, the computational complexity is approximately $\mathcal{O}\left(\rho|\mathbb{R}|L_Q^{(\mathrm{C})}L_R^{(\mathrm{C})}\right)$, which can be further accelerated by using the indexing structure of the features. For example, if we use SIFT Bag-of-Words (BoWs) as the feature descriptor for each frame, SIFT BoWs for the coarse-scale frames of all reference videos are pre-calculated offline and stored into an inverted index table. For each $q_i \in Q^{(\mathrm{C})}$, we can use all its SIFT BoW features to search the inverted index table so as to obtain the matched features. Only two SIFT features mapped to the same word and satisfied with some other additional conditions (e.g., with similar position, scale and orientation) can be regarded as a match, while the similarity between two frames is defined as the average percentage of the matches. As such, the computational complexity of frame similarity evaluation can be remarkably reduced.

(2) **Maximum Weight Matching (MWM)**: This step is to construct the candidate set $\mathbb{C}_Q$ as described in (1). Toward this end, we first group the most similar reference frames in $\mathbb{L}^{(\mathrm{C})}$ according to the reference video ID, and then obtain a new list $\mathbb{M}^{(\mathrm{C})} = \{\mathbb{M}_1, \mathbb{M}_2, \cdots, \mathbb{M}_J\}$, where J is the amount of reference videos related to the query video $Q^{(\mathrm{C})}$, and $\mathbb{M}_j$ consists of all similar frame pairs $\left\{(q_s, r_t) \mid q_s \in Q^{(\mathrm{C})}, r_t \in R^{(\mathrm{C})}\right\}$ in which all the reference frames come from $R^{(\mathrm{C})} \in \mathbb{R}$.

Thus the remaining problem is how to construct the candidate set $\mathbb{C}_Q$ from $\mathbb{M}^{(\mathrm{C})}$. From each $\mathbb{M}_j \in \mathbb{M}^{(\mathrm{C})}$, we can obtain two corresponding segments respectively in $Q^{(\mathrm{C})}$ and $R^{(\mathrm{C})}$ by checking their first and last frames in $\mathbb{M}_j$. Without loss of generality, they are denoted by $S_{Q_j}^{(\mathrm{C})}$ and $S_{R_k}^{(\mathrm{C})}$. Then the task becomes checking whether $S_{R_k}^{(\mathrm{C})}$ is potentially a near-duplicate of $S_{Q_j}^{(\mathrm{C})}$. To address this problem, a simple method is to measure the average frame similarity between the two segments. In this case, however, we cannot effectively distinguish the similarity of two segments with one-to-one frame matchings from that of two segments with one-to-many matchings. Thus a more reasonable solution is that the segment similarity is not only dependent on the frame similarity, but also on the interrelationship such as interference and granularity [32].

Toward this end, we apply the classical maximum weight matching (MWM) algorithm to calculate the coarse-level similarity score between $S_{Q_j}^{(\mathrm{C})}$ and $S_{R_k}^{(\mathrm{C})}$ under the one-to-one frame matching constraint. Given a weighted bipartite graph, the MWM problem is to find a set of vertex-disjoint edges with the maximum total weight. In general, the MWM algorithm, by optimizing the total weight of matchings, is able to rank relevant segments based on the similarity of visual and granularity. Let $\omega(q_s, r_t)$ denote the frame similarity between $q_s \in S_{Q_j}^{(\mathrm{C})}$ and $r_t \in S_{R_k}^{(\mathrm{C})}$, and $\mathsf{m}(q_s, r_t)$ denote whether q_s and r_t are a matching pair in $\mathbb{M}_j$, then

$$
\begin{aligned}
Sim(S_{Q_j}^{(\mathrm{C})}, S_{R_k}^{(\mathrm{C})}) &= \frac{1}{L_{j,k}} \max \sum_{(q_s, r_t) \in \mathbb{M}_j} \omega(q_s, r_t)\mathsf{m}(q_s, r_t), \\
s.t. \quad & \sum_{r_t} \mathsf{m}(q_s, r_t) = 1, \text{for} \forall q_s \in S_{Q_j}^{(\mathrm{C})}, \\
& \sum_{q_s} \mathsf{m}(q_s, r_t) = 1, \text{for} \forall r_t \in S_{R_k}^{(\mathrm{C})}, \\
& \mathsf{m}(q_s, r_t) \in \{0, 1\}, 0 \leqslant \omega(q_s, r_t) \leqslant 1.
\end{aligned} \tag{5}
$$

where $L_{j,k} = \min\{|S_{Q_j}^{(C)}|, |S_{R_k}^{(C)}|\}$. Figure 4 illustrates the process of applying the MWM algorithm to find a set of one-to-one matchings between $S_{Q_j}^{(C)}$ and $S_{R_k}^{(C)}$. Note that $Sim\left(S_{Q_j}^{(C)}, S_{R_k}^{(C)}\right)$ is just a result of that process.

In the process of obtaining of the $\mathbb{C}_Q$, we don't use the whole frame level matching information to filter but to use part of it to determine, this method improves the efficiency. The computation complexity can be reduced from $O((L_Q + L_R)(L + \log(L_Q + L_R)))$ to $O\left(\left(L_Q^{(C)} + L_R^{(C)}\right)\left(L + \log\left(L_Q^{(C)} + L_R^{(C)}\right)\right)\right)$.

After calculating the coarse-level similarity for all $\mathbb{M}_j \in \mathbb{M}^{(C)}$, we can apply (1) to filter the irrelevant reference segments and construct the candidate set $\mathbb{C}_Q$. The whole coarse-scale filtering process is described as Algorithm 1. Note that in the algorithm, we use the operator "=" to denote the set assignment, and use the operator "←" to denote adding an element to a set, and use Ø to denote an empty set.

Algorithm 1: Coarse-Scale Filtering with MWM

Input : $Q^{(C)} = \{q_i\}_{1,\cdots,L_Q^{(C)}}$ - The query video
$\mathbb{R} = \{R^{(C)}\}$ - The reference database
θ_0 - The pre-defined threshold
Output: $\mathbb{C}_Q$ - The candidate set
1. *Frame Similarity Evaluation*
for $\forall q_i \in Q^{(C)}$ **do**
  $\mathbb{L}_i$ = {top-T most similar reference frames for q_i};
end
2. *Filtering by MWM*
Construct $\mathbb{M}^{(C)} = \{\mathbb{M}_j\}_{1,\cdots,J}$ from $\{\mathbb{L}_i\}_{1,\cdots,L_Q^{(C)}}$;
$\mathbb{C}_Q = \emptyset$;
for $j = 1, \cdots, J$ **do**
  Extract $S_{Q_j}^{(C)}$ and $S_{R_k}^{(C)}$ from $\mathbb{M}_j$;
  Calculate $Sim(S_{Q_j}^{(C)}, S_{R_k}^{(C)})$ using (5);
  if $Sim(S_{Q_j}^{(C)}, S_{R_k}^{(C)}) > \theta_0$ **then**
    $\mathbb{C}_Q \leftarrow (S_{Q_j}^{(C)}, S_{R_k}^{(C)})$;
  end
end
return $\mathbb{C}_Q$.

2.3 Middle-scale refinement with constrained longest ascending matching subsequence

After obtaining the candidate set $\mathbb{C}_Q = \left\{\left(S_{Q_j}^{(C)}, S_{R_k}^{(C)}\right)\right\}$, the next step is to further check whether they are the near-duplicates or not, and if so, to roughly locate the positions of the near-duplicate segments in both videos. This is mainly because the coarse-scale representation of a video cannot provide enough descriptive information for accurate near-duplicate discrimination. Thus according to our assumption, this process will be performed on the

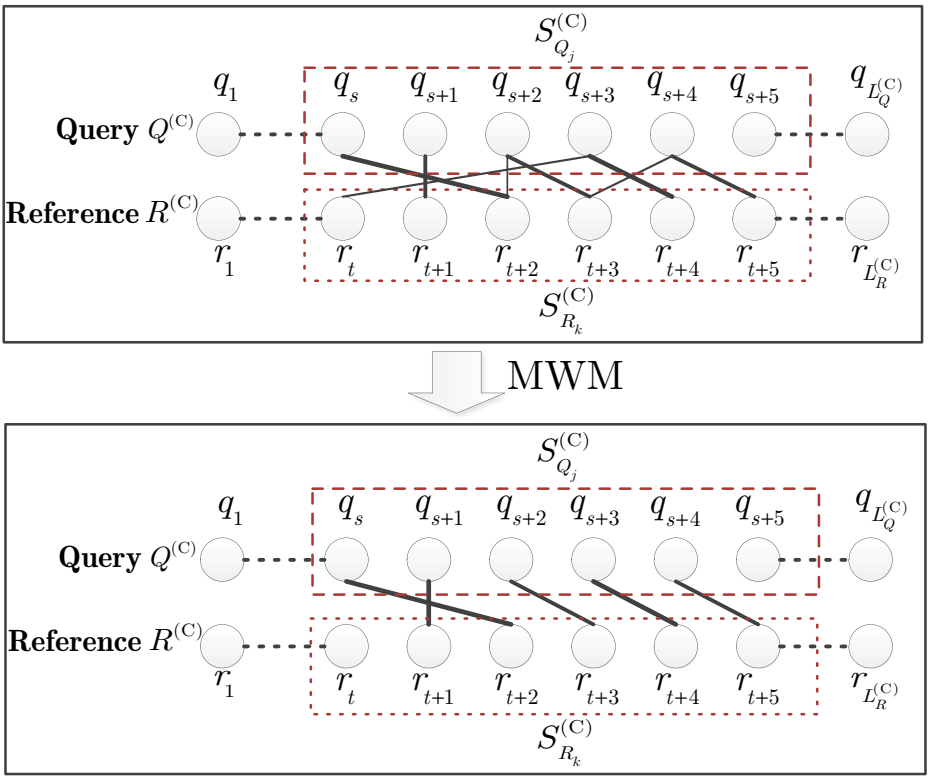

Fig. 4 Illustration of applying the MWM algorithm to find a set of one-to-one matchings between $S^{(C)}_{Q_j}$ and $S^{(C)}_{R_k}$. Note that the thicker a line between two frames, the larger the corresponding matching weight

middle-scale representations of Q and R, namely, $Q^{(M)}$ and $R^{(M)}$. Moreover, the mapping from the coarse-scale representation of a video (e.g., $Q^{(C)}$) to the middle-scale one (e.g., $Q^{(M)}$) is strictly one-to-one. So the candidate set $\mathbb{C}_Q$ can be directly represented as $\mathbb{C}_Q = \{(S^{(M)}_{Q_j}, S^{(M)}_{R_k})\}$.

According to (3), for $\forall(S^{(M)}_{Q_j}, S^{(M)}_{R_k}) \in \mathbb{C}_Q$, the condition to determine whether $\mathcal{D}(S^{(M)}_{Q_j}, S^{(M)}_{R_k})$ holds or not is that they have two subsequences whose lengths are both more than the pre-defined value ζ_1 and whose similarity is larger than the threshold θ_1. In practice, we only need to check their *most similar* subsequences. Toward this end, one possible solution is to still use the MWM algorithm on $S^{(M)}_{Q_j}$ and $S^{(M)}_{R_k}$ to find the subsequences with the optimal one-to-one matchings. However, it is enough to use such subsequences to judge whether two videos are *potentially* near-duplicates, but not to determine whether they are *definitely* near-duplicates or not. This is because the matchings between them may be very sparse (e.g., two matchings respectively in 1^{th} and 100^{th} frames). Moreover, the matchings in the two subsequences should preserve the temporal order (e.g., in ascending order). This is reasonable since two near-duplicate videos, even though they may suffer minor re-ordering edits in temporal domain, should be locally consistent. Thus by taking the maximum weight,

ascending and dense matching constraints in account, this study proposes a Constrained Longest Ascending Matching Subsequence (CLAMS) algorithm.

Traditionally, the longest common subsequence (LCS) problem is to find the longest subsequence common to all sequences (often just two sequences). Note that LCSs are not required to occupy consecutive positions within the original sequences. When applied in our problem settings, due to the probabilistic property of a matching between two frames, there are potentially a large number of LCSs between two video sequences. Thus we can also introduce the maximum weight matching constraint to the LCS problem, and meanwhile also apply the ascending matching constraint to further preserve the temporal order of matchings. In this study, we called the LCS with Maximum Weight and Ascending Matchings as the *Longest Ascending Matching Subsequence* (LAMS).

Definition (**Longest Ascending Matching Subsequence, LAMS**). Given two videos $X=\{x_1, x_2, \ldots, x_m\}$ and $Y = \{y_1, y_2, \ldots, y_n\}$, let $\omega(x_i, y_j)$ denote the frame similarity between $x_i \in X$ and $y_j \in Y$ where $0 \leqslant \omega(x_i, y_j) \leqslant 1$, and $\mathsf{m}(x_i, y_j)$ be a boolean value to indicate whether x_i and y_j are a matching (i.e., $\mathsf{m}(x_i, y_j) \in \{0, 1\}$), then the LAMS of X and Y, denoted by $\Gamma(X, Y) = \{(x_k^*, y_l^*), \ldots, (x_s^*, y_t^*)\}$, is defined as follows:

$$\Gamma(X, Y) = \underset{\forall (T_X^*, T_Y^*) \in \Omega}{\operatorname{argmax}} \sum_{x_i^* \in T_X^*, y_j^* \in T_Y^*} [\omega(x_i^*, y_j^*)\mathsf{m}(x_i^*, y_j^*)], \tag{6}$$

where

$$\begin{aligned} \Omega = \{(T_X^*, T_Y^*)\} = & \underset{\forall T_X \subseteq X, \forall T_Y \subseteq Y}{\operatorname{argmax}} \mathcal{L}(T_X, T_Y), \\ s.t. a) & 1 \leqslant i' < o' \ldots \leqslant m \text{ and } 1 \leqslant j' < p' \ldots \leqslant n \\ & for \{\ldots, (x_{i'}, y_{j'}), \ldots, (x_{o'}, y_{p'}) \ldots\} \subseteq (T_X, T_Y), \\ b) & \sum_{x_{j'}} \mathsf{m}(x_{i'}, y_{j'}) = 1 \text{ and } \sum_{y_{i'}} \mathsf{m}(x_{i'}, y_{j'}) = 1 \\ & for \ \forall (x_{i'}, y_{j'}) \in (T_X, T_Y). \end{aligned} \tag{7}$$

Here $\mathcal{L}(T_X, T_Y)$ denotes the matching length between two subsequences of X and Y (denoted by T_X and T_Y). Note that (7) is to find a set of longest matching subsequences between X and Y with one-to-one ascending matchings, then (6) is to select the one with the maximum total weight from them. In (7), the first condition expresses the ascending matching constraint, while the second one indicates that every matching pair in (T_X, T_Y) should be an one-to-one matching (i.e., for $\forall (x_{i'}, y_{j'}) \in (T_X, T_Y)$, the indexes i' and j' can appear only once). For simplicity, we also introduce an operator $\prec$. For two matching pairs (x_k^*, y_l^*) and (x_o^*, y_p^*), if $k < o$ and $l < p$, then we have $(x_k^*, y_l^*) \prec (x_o^*, y_p^*)$. As such, $\Gamma(X, Y)$ can be viewed as a special case of the longest increasing subsequences (LIS).

It is easy to apply this definition in our problem settings so as to find the optimal LAMS between $S_{Q_j}^{(\mathrm{M})}$ and $S_{R_k}^{(\mathrm{M})}$. The first half part of Algorithm 2 illustrates the procedure (as shown in the upper part of Fig. 5): In the initialization stage, we need to evaluate the frame similarity between $S_{Q_j}^{(\mathrm{M})}$ and $S_{R_k}^{(\mathrm{M})}$, and then construct a set of matching frames $\mathbb{M}^{(\mathrm{M})} = \left\{(q_s, r_t) \mid q_s \in S_{Q_j}^{(\mathrm{M})}, r_t \in S_{R_k}^{(\mathrm{M})}\right\}$ by selecting at most T similar frames for each query frame ($T = 10$ in the experiments). Then we will calculate the lengths for potential LAMSs in $(S_{Q_j}^{(\mathrm{M})}, S_{R_k}^{(\mathrm{M})})$ by using dynamic programming. Let $\mathscr{L}_{s,t}$ denote the length of a

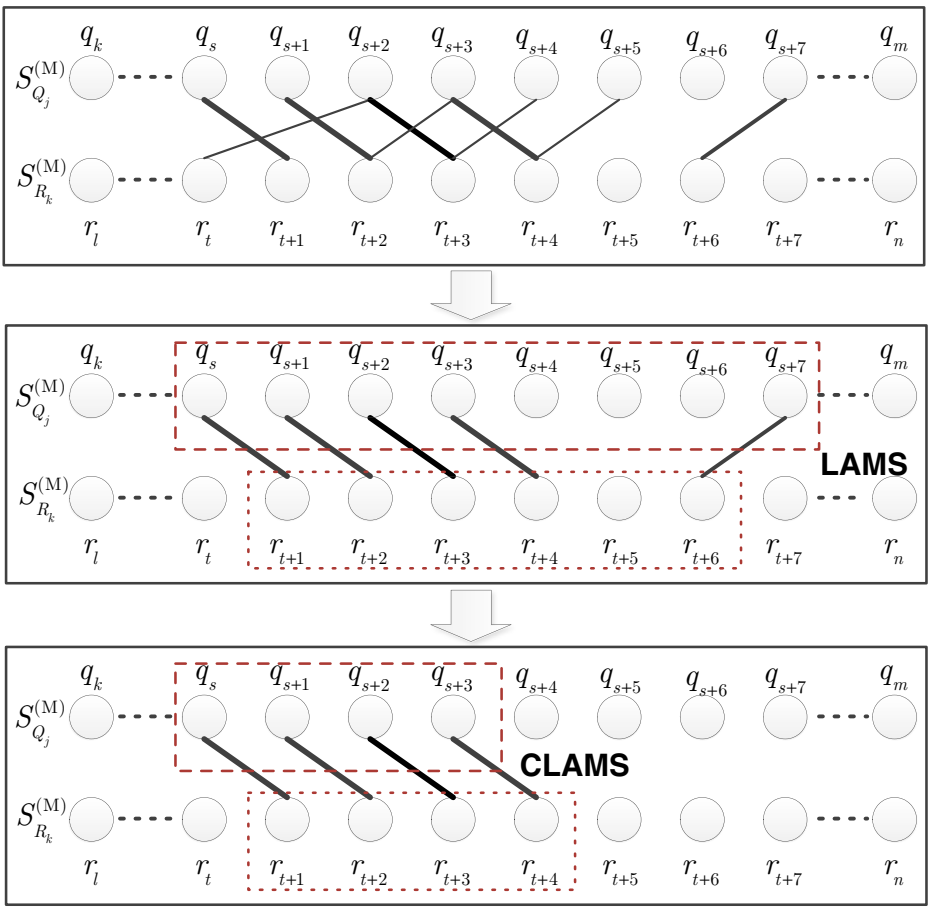

Fig. 5 Illustration of applying the CLAMS algorithm to find two most similar segments between $S^{(\mathrm{M})}_{Q_j}$ and $S^{(\mathrm{M})}_{R_k}$

potential LAMS that is ended with (q_s, r_t) and $W_{s,t}$ be the weight score of (q_s, r_t). In order to calculate the value of $\mathscr{L}_{s,t}$, we traverse all the matching pairs in $\mathbb{M}^{(\mathrm{M})}$ and check whether $\exists(q_{s'}, r_{t'}) \in \mathbb{M}^{(\mathrm{M})}$ that can meet the condition $\mathscr{L}_{s,t} + 1 > \mathscr{L}_{s',t'}$, where $(q_s, r_t) \prec (q_{s'}, r_{t'})$. If so, we can update $\mathscr{L}_{s',t'}$ with $\mathscr{L}_{s,t} + 1$ and the corresponding weight score $W_{s',t'}$ of $(q_{s'}, r_{t'})$ with $W_{s,t} + \omega(q_{s'}, r_{t'})$. At the same time, we need to update $P_{s',t'}$ with (s, t), where $P_{s',t'}$ records the previous matching pair index of (s', t') in the potential LAMS; otherwise, if $\mathscr{L}_{s,t} + 1 = \mathscr{L}_{s',t'}$ and $W_{s,t} + \omega(q_{s'}, r_{t'}) > W_{s',t'}$, we also need to update the weight score $W_{s',t'}$ with $W_{s,t} + \omega(q_{s'}, r_{t'})$ and $P_{s',t'}$ with (s, t). For simplicity, we define the operator $\Phi(s', t', s, t)$ as follows:

$$\Phi(s', t', s, t) : \begin{cases} P_{s',t'} = (s, t), \\ W_{s',t'} = W_{s,t} + \omega(q_{s'}, r_{t'}). \end{cases} \tag{8}$$

After calculating the values $\mathscr{L}_{s,t}$ and $W_{s,t}$ that correspond to every matching pair (q_s, r_t) in $\mathbb{M}^{(\mathrm{M})}$, we can find (q_{s^*}, r_{t^*}) with the maximal values of both $\mathscr{L}_{s^*,t^*}$ and W_{s^*,t^*}. Then the subsequence with (q_{s^*}, r_{t^*}) as its end position will be selected as the optimal LAMS of $(S^{(\mathrm{M})}_{Q_j}, S^{(\mathrm{M})}_{R_k})$.

After getting the optimal LAMS between $S^{(\mathrm{M})}_{Q_j}$ and $S^{(\mathrm{M})}_{R_k}$, we need to further apply the dense matching constraint. In this study, we refer to the densest matching subsequence of a LAMS as the *Constrained LAMS* (CLAMS for short). Intuitively, for a matching $(q_s, r_t) \in \Gamma$, we need to calculate its *denseness* scores with respect to its previous and next matchings in Γ (denoted by $(q_{\overleftarrow{s}}, r_{\overleftarrow{t}})$ and $(q_{\overrightarrow{s}}, r_{\overrightarrow{t}})$ respectively) so as to determine whether it belongs to some dense matching subsequence or not. In out study, such denseness scores are measured by two Euclidean distances $\overleftarrow{D}$ and $\overrightarrow{D}$,

$$\begin{aligned} \overleftarrow{D}_{s,t} &= \sqrt{[t(q_s) - t(q_{\overleftarrow{s}})]^2 + [t(r_t) - t(r_{\overleftarrow{t}})]^2}, \\ \overrightarrow{D}_{s,t} &= \sqrt{[t(q_{\overrightarrow{s}}) - t(q_s)]^2 + [t(r_{\overrightarrow{t}}) - t(r_t)]^2}, \end{aligned} \tag{9}$$

where the function $t(x)$ denotes the corresponding timestamp of the keyframe x. Note that $\overleftarrow{D}_{s,t} = \mathrm{INF}$ for the first matching in Γ and $\overrightarrow{D}_{s,t} = \mathrm{INF}$ for the last matching. In practice, for Γ, we only need to calculate $\overleftarrow{D}_{s,t}$ or $\overrightarrow{D}_{s,t}$ totally by $(|\Gamma| - 1)$ times because $\overleftarrow{D}_{s,t} = \overrightarrow{D}_{\overleftarrow{s},\overleftarrow{t}}$ if (q_s, r_t) is a non-first matching and $\overrightarrow{D}_{s,t} = \overleftarrow{D}_{\overrightarrow{s},\overrightarrow{t}}$ if (q_s, r_t) is a non-last matching.

Let Λ denote the current dense matching subsequence and $(q_s, r_t) \in \Gamma$ be the matching under consideration, then we can use the following rules to update Λ:

$$\begin{cases} \Lambda = \emptyset, & \text{if } \overrightarrow{D}_{s,t} > \tau \text{ and } \overleftarrow{D}_{s,t} > \tau; \\ \Lambda = \{(q_s, r_t)\}, & \text{if } \overrightarrow{D}_{s,t} \leqslant \tau \text{ and } \overleftarrow{D}_{s,t} > \tau; \\ \Lambda \leftarrow (q_s, r_t), & \text{otherwise.} \end{cases} \tag{10}$$

where τ be a pre-defined denseness threshold ($\tau = 7$ in our study). That is, if $\overrightarrow{D}_{s,t} > \tau$ and $\overleftarrow{D}_{s,t} > \tau$ (i.e., (q_s, r_t) is an isolated matching in Γ), then Λ should be an empty sequence; if $\overrightarrow{D}_{s,t} \leqslant \tau$ and $\overleftarrow{D}_{s,t} > \tau$, then (q_s, r_t) should be the first matching in Λ; otherwise, Λ is currently not an empty sequence and thus (q_s, r_t) should be added into Λ. In particular, if $\overleftarrow{D}_{s,t} \leqslant \tau$ and $\overrightarrow{D}_{s,t} > \tau$, (q_s, r_t) should be the last matching in a dense matching subsequence Λ. In this case, we should further check whether $|\Lambda|$ is larger than the length of Γ^* or not, where Γ^* is the dense matching subsequence that has been already obtained. If so, Γ^* should be updated with Λ. This process is illustrated in the third step in Algorithm 2 (as shown in the lower part of Fig. 5). For simplicity, we use $\overleftarrow{D}$ and $\overrightarrow{D}$ instead of $\overleftarrow{D}_{s,t}$ and $\overrightarrow{D}_{s,t}$ in the algorithm.

The CLAMS algorithm is only calculated on part of the matching frame pairs between the query video sequence and the reference video sequence, so the procedure has relatively high efficiency.

Algorithm 2: The CLAMS Algorithm at the Middle Scale

Input :
$\forall(S_{Q_j}^{(\mathbf{M})}, S_{R_k}^{(\mathbf{M})}) \in \mathbb{C}_Q$ - A candidate near-duplicate pair.
Output:
$\Gamma^*(S_{Q_j}^{(\mathbf{M})}, S_{R_k}^{(\mathbf{M})})$ - The CLAMS of $(S_{Q_j}^{(\mathbf{M})}, S_{R_k}^{(\mathbf{M})})$.
1. Initialization:
$\mathbb{M}^{(\mathbf{M})} = \emptyset$;
for $\forall q_s \in S_{Q_j}^{(\mathbf{M})}$ **do**
  $\mathbb{M}_s^{(\mathbf{M})} = \{(q_s, r_t) \mid r_t$ is among top-T most similar frames in $S_{R_k}^{(\mathbf{M})}$ for $q_s\}$;
  Sort $\mathbb{M}_s^{(\mathbf{M})}$ in ascending order by $r_t \in S_{R_k}^{(\mathbf{M})}$;
  $\mathbb{M}^{(\mathbf{M})} \leftarrow \mathbb{M}_s^{(\mathbf{M})}$;
end
2. Find the optimal LAMS $\Gamma(S_{Q_j}^{(\mathbf{M})}, S_{R_k}^{(\mathbf{M})})$:
$W_{s,t} = [0]$, $\mathscr{L}_{s,t} = [1]$, $P_{s,t} = \emptyset$ for $\forall(q_s, r_t) \in \mathbb{M}^{(\mathbf{M})}$;
for $\forall(q_s, r_t) \in \mathbb{M}^{(\mathbf{M})}$ **do**
  Let $\mathcal{M}$ be the subset of $\mathbb{M}^{(\mathbf{M})}$ after (q_s, r_t);
  if $\mathscr{L}_{s,t} = 1$ **then**
    $W_{s,t} = \omega(q_s, r_t)$;
  end
  for $\forall(q_{s'}, r_{t'}) \in \mathcal{M}$ **do**
    if $\mathscr{L}_{s,t} + 1 > \mathscr{L}_{s',t'}$ **then**
      $\mathscr{L}_{s',t'} = \mathscr{L}_{s,t} + 1$;
      $(P_{s',t'}, W_{s',t'}) = \Phi(s', t', s, t)$;
    end
    else if $\mathscr{L}_{s,t}+1=\mathscr{L}_{s',t'}$ & $W_{s,t}+\omega(q_{s'}, r_{t'})>W_{s',t'}$ **then**
      $(P_{s',t'}, W_{s',t'}) = \Phi(s', t', s, t)$;
    end
  end
end
$(q_{s^*}, r_{t^*}) = \text{argmax}_{\forall(q_s,r_t)\in\{(q_s^\dagger, r_t^\dagger)\}} W_{s,t}$,
  where $\{(q_s^\dagger, r_t^\dagger)\} = \text{argmax}_{\forall(q_s,r_t)\in\mathbb{M}^{(\mathbf{M})}} \mathscr{L}_{s,t}$;
Obtain $\Gamma(S_{Q_j}^{(\mathbf{M})}, S_{R_k}^{(\mathbf{M})})$ by backtracking $\{P_{s,t}\}$ from (q_{s^*}, r_{t^*});
3. Calculate the CLAMS $\Gamma^*\left(S_{Q_j}^{(\mathbf{M})}, S_{R_k}^{(\mathbf{M})}\right)$:
$\Gamma^*(S_{Q_j}^{(\mathbf{M})}, S_{R_k}^{(\mathbf{M})}) = \emptyset$;
for $\forall(q_s, r_t) \in \Gamma(S_{Q_j}^{(\mathbf{M})}, S_{R_k}^{(\mathbf{M})})$ **do**
  $\overleftarrow{D} = \overrightarrow{D} = \text{INF}$;
  Calculate $\overleftarrow{D}$ and/or $\overrightarrow{D}$ for (q_s, r_t);
  Update Λ using (10);
  if $\overleftarrow{D} \leqslant \tau$ & $\overrightarrow{D} > \tau$ & $|\Lambda| > |\Gamma^*\left(S_{Q_j}^{(\mathbf{M})}, S_{R_k}^{(\mathbf{M})}\right)|$ **then**
    $\Gamma^*(S_{Q_j}^{(\mathbf{M})}, S_{R_k}^{(\mathbf{M})}) = \Lambda$;
  end
end
return $\Gamma^*\left(S_{Q_j}^{(\mathbf{M})}, S_{R_k}^{(\mathbf{M})}\right)$.

After obtaining $\Gamma^*(S_{Q_j}^{(\mathrm{M})}, S_{R_k}^{(\mathrm{M})})$, we can use (3) to check $\mathcal{D}\left(S_{Q_j}^{(\mathrm{M})}, S_{R_k}^{(\mathrm{M})}\right)$, and further determine whether $\mathcal{D}(Q, R)$ holds or not (recalling that $\mathcal{D}\left(S_Q^{(\mathrm{C})}, S_R^{(\mathrm{C})}\right) = 1$).

2.4 Fine-scale localization with bi-directional scanning

If it is asserted that $\mathcal{D}(Q, R) = 1$, the remaining problem is to determine the precise locations of the near-duplicate segments in Q and R. This will be done at the fine scale because the fine-scale representation can provide a more detailed view about a video, consequently enabling much better localization precision. As mentioned in Section 2.1, the basic idea of this localization process is to find the longest near-duplicate segments in $Q^{(\mathrm{F})}$ and $R^{(\mathrm{F})}$ that are with the asserted $\left(T_Q^{(\mathrm{M})}, T_R^{(\mathrm{M})}\right) = \Gamma^*\left(S_{Q_j}^{(\mathrm{M})}, S_{R_k}^{(\mathrm{M})}\right)$ as their center.

As said by (4), we need to carry out bi-directional scannings (including forward and backward scanning operations) in $Q^{(\mathrm{F})}$ and $R^{(\mathrm{F})}$. During the processes of forward and backward scannings, the sliding window is used to iteratively scan $Q^{(\mathrm{F})}$ and $R^{(\mathrm{F})}$ simultaneously. Moreover, the forward and backward scanning operations can be performed independently. Take the backward scanning as the example. The operation of backward scanning in $Q^{(\mathrm{F})}$ starts from the beginning position of $T_Q^{(\mathrm{M})}$, or equivalently $T_Q^{(\mathrm{F})}$ at the fine scale. At each iteration, the sliding window moves backward with the step length of Δt; meantime, the backward scanning in $R^{(\mathrm{F})}$ starts from the beginning position of $T_R^{(\mathrm{F})}$ and also moves backward with the step length Δt. Without loss of generality, we use $\delta_{Q_k}^{(\mathrm{F})-}$ and $\delta_{R_k}^{(\mathrm{F})-}$ denote the windowed segments of $Q^{(\mathrm{F})}$ and $R^{(\mathrm{F})}$ at the k^{th} iteration of backward scanning. Then we can use the MWM algorithm (i.e., similar to (5) except using the fine-scale representation) to calculate the segment similarity between $\delta_{Q_k}^{(\mathrm{F})-}$ and $\delta_{R_k}^{(\mathrm{F})-}$, namely, $Sim\left(\delta_{Q_k}^{(\mathrm{F})-}, \delta_{R_k}^{(\mathrm{F})-}\right)$. If $Sim\left(\delta_{Q_k}^{(\mathrm{F})-}, \delta_{R_k}^{(\mathrm{F})-}\right)$ is lower than the threshold θ_2, the backward scanning process will be terminated. Finally, the segment that starts from the terminal position to the start position in $Q^{(\mathrm{F})}$ during the backward scanning process can be considered as $\Delta S_Q^{(\mathrm{F})-}$, while the segment starts from the terminal position to the start position in $R^{(\mathrm{F})}$ can be considered as $\Delta S_R^{(\mathrm{F})-}$. Similarly, in the forward scanning process, we can obtain $\Delta S_Q^{(\mathrm{F})+}$ and $\Delta S_R^{(\mathrm{F})+}$.

After getting $\Delta S_Q^{(\mathrm{F})-}$, $\Delta S_R^{(\mathrm{F})-}$, $\Delta S_Q^{(\mathrm{F})+}$ and $\Delta S_R^{(\mathrm{F})+}$, we can then determine the scope of near-duplicate segments (S_Q, S_R) between Q and R:

$$\begin{aligned} S_Q &= \Delta S_Q^{(\mathrm{F})-} \bowtie T_Q^{(\mathrm{F})} \bowtie \Delta S_Q^{(\mathrm{F})+}, \\ S_R &= \Delta S_R^{(\mathrm{F})-} \bowtie T_R^{(\mathrm{F})} \bowtie \Delta S_R^{(\mathrm{F})+}, \end{aligned} \tag{11}$$

where the operator $\bowtie$ denotes the sequence concatenation operation. Note that the similarity between S_Q and S_R, namely, $Sim(S_Q, S_R)$, can also be calculated by the MWM algorithm. Often, this similarity score can be used for the final result ranking.

3 Experiments

In this section, we evaluate the effectiveness of the proposed MS-VSM by comparison with several state-of-the-art methods.

3.1 Experiment settings

The most widely-used benchmark datasets are adopted in our experiments, including MUSCLE-VCD-2007 and CC_WEB_VIDEO.

MUSCLE-VCD-2007 [20] contains a reference database of 101 raw videos (about 100 hours) collected from various resources and two query video sets (i.e., ST1, ST2) of 15 and 3 videos transformed from some reference and non-reference videos. Only visual transformations such as camcording, subtitles, reencoding, crop ang change of color were used to generate the query videos. The objectives of ST1 and ST2 are slightly different: ST1 is to determine whether a query is a near-duplicate of some reference video in the database, while ST2 is to localize the near-duplicate segments with the starting and ending timestamps. For the ST1 task, Quality (Q) is calculated as the percentage of correct answers; while for ST2 task, two metrics, QualitySegment (QS) and QualityFrame (QF), are adopted to assess the detection correctness and localization accuracy of near-duplicate segments respectively.

$$\mathbf{QS} = \frac{TP_{seg} - FA_{seg}}{N_{seg}} \tag{12}$$

$$\mathbf{QF} = 1 - \frac{F_{Miss}}{F_{Total}} \tag{13}$$

where TP_{Seg} (or FA_{Seg}) is the number of correctly-matched (or mismatched) video segments, N_{Seg} is the total number of segments in all queries, F_{Miss} denotes the number of mismatched frames, while F_{Total} is the total number of frames in all queries.

CC_WEB_VIDEO [43] contains totally 12,790 Web video clips that can be divided into 24 groups. All videos were collected from YouTube, Google Video and Yahoo! Video, often with low video quality. For each group, one video is selected as the query video, while the others are regarded as "near-duplicate" or "irrelevant" in the ground-truth. Note that these near-duplicate videos were typically generated by applying transformation of format conversion, encoding parameters change, photometric variations (e.g. color, brightness changes), editing operations (e.g. logo insertion and border adding) and content modifications on the original videos. According to the dataset evaluation protocol, the precision-recall curve is used to assess the performance, where precision and recall are calculated as follows [4]:

$$\mathbf{Precision} = \frac{TP}{TP + FP}, \tag{14}$$

$$\mathbf{Recall} = \frac{TP}{TP + FN}. \tag{15}$$

Here true positive (TP) stands for the number of correct identifications in the whole positive samples, false negative (FN) indicates the number of positive samples which are mistaken as negatives, while false positive (FP) signifies the count of negative examples erroneously assumed to be as positives. Moreover, the performance can also be evaluated by another metric - mean average precision (MAP) [9]:

$$\mathbf{MAP}(\mathbb{Q}) = \frac{1}{|\mathbb{Q}|} \sum_{j=1}^{|\mathbb{Q}|} \frac{1}{m_j} \sum_{k=1}^{m_j} \frac{k}{\gamma_k}, \tag{16}$$

where the set of relevant videos for $Q_j \in \mathbb{Q}$ is $\{R_1, \cdots, R_{m_j}\}$, m_j is the number of its relevant videos, and γ_k is the rank of the k^{th} retrieved relevant video.

3.2 Implementation details

In practice, the implementation of the MS-VSM method is mainly related to two issues, namely, the preprocessing and audio-visual feature extraction.

Preprocessing One of the most important preprocessing operations is to extract keyframes to derive the multiscale representation of a video. As mentioned in Section 2.1, the sampling rate at the fine scale is 10:1 if the frame rate of a video X is 30fps. That is, visual keyframes are obtained for the fine-scale representation $X^{(\mathrm{F})}$ by uniformly sampling its visual component at a rate of 3fps. This can effectively avoid the error caused by the different frame rates of videos. Then its middle-scale representation $X^{(\mathrm{M})}$ can be generated from $X^{(\mathrm{F})}$ at the sampling rate of 5:1, while its coarse scale representation $X^{(\mathrm{C})}$ can be derived from $X^{(\mathrm{M})}$ at the sampling rate of 3:1. As such, for an one-minute video sequence X with the frame rate of 30 fps, the frame numbers of $X^{(\mathrm{F})}$, $X^{(\mathrm{M})}$ and $X^{(\mathrm{C})}$ are 180, 36, and 12, respectively. Obviously, this keyframe extraction is very simple and computationally efficient.

For the dataset who contains audio component in each video or which involves audio transformations in generating near-duplicates, we also need to extract audio frames for each video. It should be noted that the extracted audio frames or keyframes should be aligned with the corresponding visual frames or keyframes. To do so, in our study, an audio frame with a length of 0.37 seconds is extracted from the audio signal for every interval of 11.6 milliseconds. Meanwhile, the overlap factor of two consecutive audio frames is set to 31/32. Thus for an one-minute video segment, about 5,120 audio frames can be extracted, which roughly correspond to 180 visual frames. From such a fine-scale representation, we can still follow the sampling rate of 5:1 and approximately 3:1 to derive its middle-scale and coarse-scale representations.

Picture-in-picture (PiP) detection is also performed in the preprocessing step. Instead of using a simple Hough-transform-based method, we use a recently-proposed PiP detection method [33] that introduces the spatio-temporal slicing to establish the probability measurement of the corresponding edge surface and then uses an optimization algorithm to refine vertical and horizontal edge lines. For queries with PiP, the foreground and non-foreground key-frames will be processed respectively to check whether the corresponding videos are copies. In addition, queries asserted as non-copies will be flipped and matched again to deal with flip transformation.

Feature extraction In the implementation, we use SIFT [28, 29] as the visual feature and then apply the BoW technique to convert each SIFT descriptor into a visual word (1000 words generated from the flickr 1M dataset). Meanwhile, the position in the 2×2 partition of the image, scale (large vs. small), and orientation (quantized into 12 bins) are also taken into account. Only two SIFT features mapped to the same word and with similar position, scale and orientation can be regarded as a match. For all reference videos, SIFT BoWs are stored into an inverted index table for quick matching.

For audio feature, this implementation uses a robust audio fingerprint (AFP) proposed in [13]. AFP extracts a 32 bit sub-fingerprint for each audio frame by calculating energy differences along the frequency and time axes. Like [13], the bit errors are used to measure the similarity between two AFPs, and all the AFPs for reference videos are organized in a hash lookup table for quick search. For multimodal feature fusion, we adopt the simple

result fusion strategy [31, 41] for near-duplicate detection, since it is beyond the focus of this study. That is, we utilize a set of audiovisual features to construct several detectors and then derive the final result by fusing the detection results from these detectors.

3.3 Experimental results on the MUSCLE-VCD-2007 dataset

This experiment was designed to evaluate the performance of the proposed MS-VSM on the MUSCLE-VCD-2007 dataset. As mentioned above, the video quality in this dataset is relatively good. Thus many existing methods (e.g.,[44]) can achieve very promising near-duplicate detection and localization results. By comparison, several state-of-the-art results on this dataset were cited directly from the literature, including Anguera2009 [1], Tan2009 [40], Cui2010 [7], Yeh2011 [46], Zheng2011 [48], Ren2012 [34], Kim2014 [18] and Wu2014 [44].

Note that in this dataset, only the detection accuracy of the top-1 result should be evaluated in the ST1 task (in terms of **Q**), while both the detection accuracy and localization precision should be evaluated in the ST2 task (in terms of **QS** and **QF**). Table 2 shows the experimental results. In the ST1 task, many methods, including our MS-VSM, achieved excellent detection performance, with **Q** of 1.0. This means they could correctly detect all near-duplicates on the ST1 set. While in the ST2 task, the MS-VSM, also showed significant advantage. These results are even better than the most-recent results in Wu2014 which

Table 2 Comparison on the MUSCLE-VCD-2007 dataset

Method	ST1	ST2	
	Q	QS	QF
ADV	86%	33%	17%
IBM	86%	–	–
CITYU	66%	86%	76%
CAS	53%	–	–
GOS	83%	–	–
SSM	100%	–	–
Yeh2009	93%	–	–
Poullot	93%	–	–
Zheng	100%	90%	85%
SSBelt_CE	100%	95%	90%
SSBelt_LBP	100%	95%	90%
Cui	100%	86%	–
Yeh2011	93%	86%	–
Anguera	100%	–	–
Tan	100%	90%	82%
Yeh2009	86%	–	–
Jiang	100%	–	–
Kim	93%	86%	–
Ren	93%	93%	–
TASC	100%	100%	97.7%
MS-VSM	100%	100%	90.54%

obtained 0.9 **QS** and 0.9 **QF**. These results demonstrate that the proposed MS-VSM exhibits almost perfect near-duplicate detection and localization performance on this benchmark dataset, with very high processing efficiency.

3.4 Experimental results on the CC_WEB_VIDEO dataset

This experiment was to evaluate the performance of the MS-VSM on the CC_WEB_VIDEO dataset. The videos in this dataset were all collected from the Internet, thus the video quality is rather low and the used transformations can also be widely found in other Internet videos. Therefore, this experiment can verify whether the proposed MS-VSM could be used in the large-scale online video deduplication applications.

On this dataset, given a query video, the MS-VSM should retrieve all duplicate and near-duplicate videos. Then the performance is evaluated in terms of **MAP** and precision-recall curve. In this case, the fine-scale localization process should not be performed. For comparison, some state-of-the-art results were also collected from the literature, including Wu2007 [43], Shang2010 [36], Liu2011 [26], Cai2012 [2], Zhou2012 [49], Song2013 [39] and Wu2014 [44]. Note that Liu2011 did not provide its MAP result while Cai2012 did not provide its precision-recall data. Thus they were excluded in the corresponding comparisons.

We evaluated the effectiveness for near-duplicate video retrieval on the CC_WEB_VIDEO dataset by using Precision-Recall curve. Figure 6 illustrates the results for the compared methods. We can see that the MS-VSM can obtain the precision of more than 90% even when the recall reaches 95%. This is consistent with the multiscale sequence matching mechanism, which enables the MS-VSM to preferably find more results under the prerequisite of keeping as high detection accuracy as possible.

Table 3 gives a comparison with respect to the overall detection accuracy. In Table 3, the **MAP** column refers to the Mean Average Precision of the 24 queries. We can see that the **MAP** of the MS-VSM is better than the other state-of-the-art results. This result is consistent with the results shown in Fig. 6. It should be also noted that, the MS-VSM is also computationally efficient on this dataset. These results show that the proposed MS-VSM is capable to effectively and efficiently cope with the large-scale online video deduplication tasks.

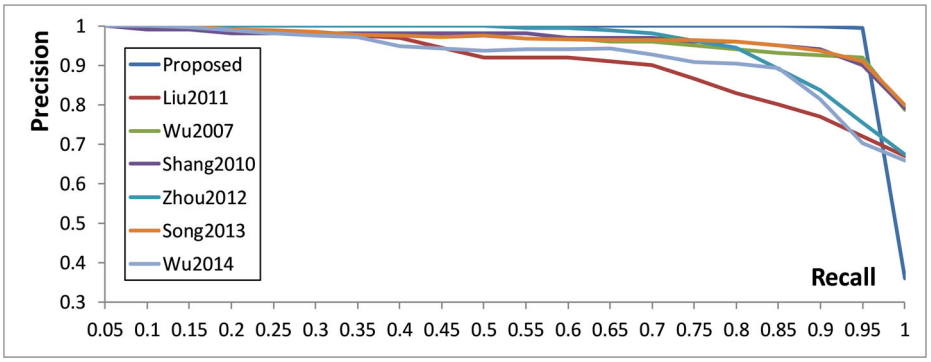

Fig. 6 CC_WEB_VIDEO Precision-Recall curves

Table 3 MAP on CC_WEB_VIDEO

Method	MAP	Method	MAP
STF_CE	0.95	SSBelt_CE	0.92
STF_LBP	0.953	SSBelt_LBP	0.922
SIG_CH	0.892	TASC	0.9857
HIRACH	0.952	SEQ	0.8852
OM	0.91	DTW	0.9136
MFH	0.954	DP	0.9009
ASVT	0.956	G-MFH	0.9582
Cai	0.918	PD	0.8853
Song	0.958	PI-tree	0.9405
Su	0.958	MS-VSM	0.9647

4 Conclusion

In this paper, we introduce a novel multiscale video sequence matching algorithm MS-VSM which uses multiscale sequence matching to determine whether two video sequences contains near-duplicate segments. The proposed algorithm shows good performance. In the future research, we will improve our algorithm with adaptive threshold.

Acknowledgments This work is partially supported by grants from the National Key R&D Program of China under grant 2017YFB1002401, the National Natural Science Foundation of China under contract No. U1611461, No. 61390515, and No. 61425025.

References

1. Anguera X, Obrador P, Oliver N (2009) Multimodal video copy detection applied to social media. In: Proceedings of the first SIGMM workshop on Social media. ACM, pp 57–64
2. Cai Y, Tong W, Yang L, Hauptmann AG (2012) Constrained keypoint quantization: towards better bag-of-words model for large-scale multimedia retrieval. In: Proceedings of the 2nd ACM International Conference on Multimedia Retrieval. ACM, p 16
3. Chen T, Jiang S, Chu L, Huang Q (2011) Detection and location of near-duplicate video sub-clips by finding dense subgraphs. In: Proceedings of the 19th ACM international conference on Multimedia. ACM, pp 1173–1176
4. Chiu C-Y, Wang H-M, Chen C-S (2010) Fast min-hashing indexing and robust spatio-temporal matching for detecting video copies. ACM Trans Multimedia Comput Comm Appl 6(2):23
5. Chiu C-Y, Tsai T-H, Liou Y-C, Han G-W, Chang H-S (2014) Near-duplicate subsequence matching between the continuous stream and large video dataset. IEEE Trans Multimedia 16(7):1952—-1962
6. Coskun B, Sankur B, Memon N (2006) Spatio–temporal transform based video hashing. IEEE Trans Multimedia 8(6):1190–1208
7. Cui P, Wu Z, Jiang S, Huang Q (2010) Fast copy detection based on slice entropy scattergraph. In: 2010 IEEE International Conference on Multimedia and Expo (ICME). IEEE, pp 1236–1241
8. Diego F, Serrat J, López A. M. (2013) Joint spatio-temporal alignment of sequences. IEEE Trans Multimedia 15(6):1377–1387
9. Dong W, Wang Z, Charikar M, Li K (2008) Efficiently matching sets of features with random histograms. In: Proceedings of the 16th ACM international conference on Multimedia. ACM, pp 179–188
10. Douze M, Gaidon A, Jegou H, Marszałek M, Schmid C, et al. (2008) Inria-lears video copy detection system. In: TREC Video Retrieval Evaluation (TRECVID Workshop)
11. Douze M, Jégou H, Schmid C (2010) An image-based approach to video copy detection with spatio-temporal post-filtering. IEEE Trans Multimedia 12(4):257–266

12. Gengembre N, Berrani S-A (2008) A probabilistic framework for fusing frame-based searches within a video copy detection system. In: Proceedings of the 2008 international conference on Content-based image and video retrieval. ACM, pp 211–220
13. Haitsma J, Kalke T (2012) A highly robust audio fingerprinting system. In: Proceedings of Int'l Symp. Music Information Retrieval, Paris, France, pp 107–115
14. Huang T, Tian Y, Gao W, Lu J (2010) Mediaprinting: Identifying multimedia content for digital rights management. Computer 43(12):0028–35
15. Huang Z, Shen HT, Shao J, Cui B, Zhou X (2010) Practical online near-duplicate subsequence detection for continuous video streams. IEEE Trans Multimedia 12(5):386–398
16. Kim C, Vasudev B (2005) Spatiotemporal sequence matching for efficient video copy detection. IEEE Trans Circuits Syst Video Technol 15(1):127–132
17. Kim H-S, Lee J, Liu H, Lee D (2008) Video linkage: group based copied video detection. In: Proceedings of the 2008 international conference on Content-based image and video retrieval. ACM, pp 397–406
18. Kim S, Choi JY, Han S, Ro YM (2014) Adaptive weighted fusion with new spatial and temporal fingerprints for improved video copy detection. Signal Process Image Commun 29(7):788–806
19. Law-To J, Chen L, Joly A, Laptev I, Buisson O, Gouet-Brunet V, Boujemaa N, Stentiford F (2007) Video copy detection: a comparative study. In: Proceedings of ACM Int'l Conf. on Image and Video Retrieval (CIVR'07), Amsterdam, The Netherlands, pp 371–378
20. Law-To J, Joly A, Boujemaa N (2007) Muscle-vcd-2007: a live benchmark for video copy detection
21. Lin J, Duan L-Y, Wang S, Bai Y, Lou Y, Chandrasekhar V, Huang T, Kot A, Gao W (2017) Hnip: Compact deep invariant representations for video matching, localization, and retrieval. IEEE Trans Multimedia 19(9):1968–1983
22. Liu B, Li Z, Yang L, Wang M, Tian X (2011) Real-time video copy-location detection in large-scale repositories. IEEE Multimedia 18(3):22–31
23. Liu H, Lu H, Xue X (2013) A segmentation and graph-based video sequence matching method for video copy detection. IEEE Trans Knowl Data Eng 25(8):1706–1718
24. Liu H, Zhao Q, Wang H, Lv P, Chen Y (2016) An image-based near-duplicate video retrieval and localization using improved edit distance, Multimedia Tools and Applications
25. Liu J, Huang Z, Cai H, Shen HT, Ngo CW, Wang W (2013) Near-duplicate video retrieval: Current research and future trends. ACM Comput Surv 45(4):44
26. Liu J, Huang Z, Shen HT, Cui B (2011) Correlation-based retrieval for heavily changed near-duplicate videos. ACM Trans Inf Syst 29(4):21
27. Liu Y, Zhao W-L, Ngo C-W, Xu C-S, Lu H-Q (2010) Coherent bag-of audio words model for efficient large-scale video copy detection. In: Proceedings of the ACM International Conference on Image and Video Retrieval. ACM, pp 89–96
28. Lowe DG (1999) Object recognition from local scale-invariant features 2:1150–1157
29. Lowe DG (2004) Distinctive Image Features from Scale-Invariant Keypoints. Int J Comput Vis 60(2):91–110
30. Malekesmaeili M, Fatourechi M, Ward RK (2009) Video copy detection using temporally informative representative images. In: International Conference on Machine Learning and Applications, 2009. ICMLA'09. IEEE, pp 69–74
31. Mou L, Huang T, Tian Y, Jiang M, Gao W (2013) Content-based copy detection through multimodal feature representation and temporal pyramid matching. ACM Trans Multimed Comput Commun Appl 10(1):5
32. Peng Y, Ngo C-W (2006) Clip-based similarity measure for query-dependent clip retrieval and video summarization. IEEE Trans Circuits Syst Video Technol 16(5):612–627
33. Qian M, Mou L, Li J, Tian Y (2014) Video picture-in-picture detection using spatio-temporal slicing. In: Proceedings of ICME'2014 Workshop on Emerg. Multimedia Sys. and Appl., Chengdu, China
34. Ren J, Chang F, Wood T, Zhang JR (2012) Efficient video copy detection via aligning video signature time series. In: Proceedings of the 2nd ACM International Conference on Multimedia Retrieval. ACM, p 14
35. Roopalakshmi R, Ram Mohana Reddy G (2013) A novel spatio-temporal registration framework for video copy localization based on multimodal features. Signal Process 93(8):2339–2351
36. Shang L, Yang L, Wang F, Chan K-P, Hua X-S (2010) Real-time large scale near-duplicate web video retrieval. In: Proceedings of the international conference on Multimedia. ACM, pp 531–540
37. Shen HT, Shao J, Huang Z, Zhou X (2009) Effective and efficient query processing for video subsequence identification. IEEE Trans Knowl Data Eng 21(3):321–334
38. Song J, Yang Y, Huang Z, Shen HT, Hong R (2013) Multiple feature hashing for large scale near-duplicate video retrieval. IEEE Trans Multimedia 15(8):1997–2008

39. Song J, Yang Y, Huang Z, Shen HT, Luo J (2013) Effective multiple feature hashing for large-scale near-duplicate video retrieval, IEEE Trans Multimedia 15(8):1997–2008
40. Tan H-K, Ngo C-W, Hong R, Chua T-S (2009) Scalable detection of partial near-duplicate videos by visual-temporal consistency. In: Proceedings of the 17th ACM international conference on Multimedia. ACM, pp 145–154
41. Tian Y, Huang T, Jiang M, Gao W (2013) Video copy-detection and localization with a scalable cascading framework. IEEE MultiMedia 20(3):72–86
42. Wei S, Zhao Y, Zhu C, Xu C, Zhu Z (2011) Frame fusion for video copy detection. IEEE Trans Circuits Syst Video Technol 21(1):15–28
43. Wu X, Hauptmann AG, Ngo C-W (2007) Practical elimination of near-duplicates from web video search. In: Proceedings of the 15th international conference on Multimedia. ACM, pp 218–227
44. Wu Z, Aizawa K (2014) Self-similarity-based partial near-duplicate video retrieval and alignment. Int J Multimed Inf Retrieval 3(1):1–14
45. Yeh M-C, Cheng K-T (2009) Video copy detection by fast sequence matching. In: Proceedings of the ACM International Conference on Image and Video Retrieval. ACM, p 45
46. Yeh M-C, Cheng K-T (2011) Fast visual retrieval using accelerated sequence matching. IEEE Trans Multimedia 13(2):320–329
47. Zhang L, Zhang B (2014) Quotient space based problem solving: A theoretical foundation of granular computing. Elsevier Inc., Amsterdam
48. Zheng L, Qiu G, Huang J, Fu H (2011) Salient covariance for near-duplicate image and video detection. In: 2011 18th IEEE International Conference on Image Processing (ICIP). IEEE, pp 2537–2540
49. Zhou X, Chen L, Zhou X (2012) Structure tensor series-based large scale near-duplicate video retrieval. IEEE Trans Multimedia 14(4):1220–1233
50. Zhou X, Zhou X, Chen L, Bouguettaya A (2012) Efficient subsequence matching over large video databases. The VLDB J 21:489–508
51. Zhou X, Zhou X, Chen L, Bouguettaya A, Xiao N, Taylor JA (2009) An efficient near-duplicate video shot detection method using shot-based interest points. IEEE Trans Multimedia 11(5):879–891

Yuanyuan Yang is currently studying for a doctorate at Beijing University of Posts and Telecommunications, Her research interests include multimedia content analysis, mining and understanding.

Yonghong Tian received the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2005. He is currently a Full Professor with the National Engineering Laboratory for Video Technology and the Cooperative Medianet Innovation Center, School of Electronics Engineering and Computer Science, Peking University, Beijing, China. He has authored or coauthored more than 160 technical articles in refereed journals and conferences, and has owned more than 55 Chinese and US patents. His research interests include machine learning, computer vision, and multimedia big data.

Prof. Tian is a Senior Member of CIE and CCF, and a Member of ACM. He is currently an Associate Editor of the IEEE TRANSACTIONS ON MULTIMEDIA, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, IEEE MULTIMEDIA MAGAZINE, and IEEE ACCESS., and a co-Editor-in-Chief of the International Journal of Multimedia Data Engineering and Management. He has served as the Technical Program Co-Chair of IEEE ICME 2015, IEEE BigMM 2015, IEEE ISM 2015 and IEEE MIPR 2018, an Organizing Committee Member of more than ten conferences such as ACM Multimedia 2009, IEEE MMSP 2011, IEEE ISCAS 2013, and IEEE ISM 2016, and BigMM 2018, and a PC Member or Area Chair of several conferences such as CVPR, KDD, AAAI, ECCV, and ICME. He was the recipient of two national prizes and three ministerial prizes in China, and was the recipient of the 2015 EURASIP Best Paper Award for the EURASIP Journal on Image and Video Processing. His team was also ranked as one of the best performers in the TRECVID CCD/SED tasks from 2009 to 2012, and PETS 2012.

Tiejun Huang is currently a Professor with the School of Electronic Engineering and Computer Science, Peking University, Beijing, China, where he is also the Director of the Institute for Digital Media Technology. He received the Ph.D. degree in pattern recognition and intelligent system from Huazhong (Central China) University of Science and Technology, Wuhan, China, in 1998, and the master's and bachelor's degree in computer science from the Wuhan University of Technology, Wuhan, in 1995 and 1992, respectively. His research area includes video coding, image understanding, digital right management, and digital library. He has authored or co-authored over 100 peer-reviewed papers and three books. He is a member of the Board of Director for Digital Media Project, the Advisory Board of the IEEE Computing Society, and the Board of the Chinese Institute of Electronics.