Received August 27, 2019, accepted September 26, 2019, date of publication October 11, 2019, date of current version October 28, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2946889

User-Guided Clustering for Video Segmentation on Coarse-Grained Feature Extraction

XINHUI PENG, RUI LI, (Member, IEEE), JILONG WANG, AND HAO SHANG
College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China

Corresponding author: Rui Li (rui@hnu.edu.cn)

This work was supported in part by the National Key Research and Development Plan of China under Grant 2016YFB020045 and Grant 2012AA01A301-01, and in part by the National Natural Science Foundation of China with Grant 61672217 and 61932010.

ABSTRACT Video segmentation is the task of temporally dividing a video into semantic sections, which are typically based on a specific concept or a theme that is usually defined by the user's intention. However, previous studies of video segmentation have that far not taken a user's intention into consideration. In this paper, a two-stage user-guided video segmentation framework has been presented, including dimension reduction and temporal clustering. During the dimension reduction stage, a coarse granularity feature extraction is conducted by a deep convolutional neural network pre-trained on ImageNet. In the temporal clustering stage, the information of the user's intention is utilized to segment videos on time domain with a proposed operator, which calculates the similarity distance between dimension reduced frames. To provide more insight into the videos, a hierarchical clustering method that allows users to segment videos at different granularities is proposed. Evaluation on Open Video Scene Detection(OVSD) dataset shows that the average F-score achieved by the proposed method is 0.72, even coarse-grained feature extraction is adopted. The evaluation also demonstrated that the proposed method can not only produce different segmentation results according to the user's intention, but it also produces hierarchical segmentation results from a low level to a higher abstraction level.

INDEX TERMS Clustering methods, dimension reduction, feature extraction, user centered design, video segmentation.

I. INTRODUCTION

Video segmentation is the task of temporally dividing a video into semantic sections [1], which are typically based on a specific concept or a theme usually defined by the user's intention. In video segmentation, different segmentation granularities may exist, which mainly refer to shots and scenes. Generally speaking, the shot refers to a series of frames taken from the same camera in continuous time. The scene is a sequence of semantically related and temporally adjacent shots depicting a high-level concept or story. Video segmentation is fundamental to the process of summarizing, retrieving, understanding, and classifying the content of a video.

Currently, there are three basic research approaches that have been adopted for video segmentation. The first is the rules-based method, which uses heuristic rules derived from the film industry to divide videos [2]–[7]. However, merely relying on editing rules limits this method's ability to process certain genres and certain types of video productions. The second is the graph-based method, which uses graphs to represent the arrangement of shots and then partitions the graphs into sections [8]–[13]. However, these methods lack a unified agreement on how to represent the video. For example, some represent the individual shots as nodes while others represent shot clusters as nodes. Due to this reason, the motivations, rationality, and conclusions from given experiments cannot necessarily be shared or easily migrated between the methods [8]. The third is the clustering-based method, which uses the similarities of shots to group frames that are then put into meaningful clusters [8], [14]–[18]. These methods ignore the temporal consistency of videos. Since videos have obvious time structures and intrinsic sequences, algorithms that do not consider the temporal position and order of video frames may divide video frames into non-contiguous segmented sections. Specifically, situations such as outlier shots and ping-pong shot sequences are likely to cause complete mislabeling of the segmented sections [8].

The associate editor coordinating the review of this manuscript and approving it for publication was Jenny Mahoney.

In our opinion, user intention plays an important role in video segmentation. In normal conditions, user intention refers to the user having their own segmentation requirements when segmenting a video. This means that different people will produce different segmentation results on the same video at different times. For example, sometimes people pay more attention to the outline of the video, so a coarse-grained partition is needed. Other times people pay more attention to the details of the video, so we need to divide the video more finely. In addition, user intention is also slightly different from weak supervision. Weak supervision is a new programming paradigm for machine learning, which uses weaker forms of supervision, such as heuristically generating training data with external knowledge bases, patterns, rules. Essentially, weak supervision is the way to programmatically generate training data—or more succinctly, programming training data [19].

However, previous studies of video segmentation have not taken user intention into consideration. This paper proposes a temporal clustering method to deal with user intention for video segmentation, which utilizes several segmentation reference points set by users to temporally cluster frames into semantic sections. Our contributions in this work are as follows.

1. A two-stage video segmentation framework for video stream processing was presented, including dimension reduction and temporal clustering. Instead of extracting features frame-by-frame, the dimension reduction extracts the features in a coarse granularity, which can save time and overhead space.
2. An operator was proposed to measure the similarity distance between frames after the dimension reduction stage. The operator calculates the similarities between sub-blocks of two frames with different partitions, and it then chooses the minimum sum of sub-block distances as the similarity distance between the two frames.
3. A user-guided temporal UGT clustering method was proposed, which utilizes user intention information to segment videos on a time domain. The proposed method regresses the cluster radius in chronological order according to the segmentation reference points set by users, and then clusters all of the data with the regressed radius in chronological order.
4. A hierarchical clustering method was subsequently proposed that allows users to perform video segmentation at different granularities. This method uses a frame with the smallest difference from other frames to represent each low-level cluster, and it then treats these frames as a new frame sequence. It conducts the temporal mean-shift clustering method again to obtain high-level clusters.

To the best of our knowledge, we are the first that study the video segmentation method on coarse-grained feature extraction. The approach proposed in the research provides a promising path for video segmentation in a fast and iterative manner.

II. RELATED WORK

There are a large number of published studies conducted on video segmentation, which can be classified into three main types: the rules-based method, the graph-based method, and the clustering-based method.

A. THE RULES-BASED METHOD

The rules-based method uses heuristic rules derived from the film industry to divide videos. Vasileios *et al.* took color histograms as features to segments video, which used the Euclidean distance and the spectral clustering algorithm to cluster frames [2]. Ellouze *et al.* repaired the over-segmentation problem by classifying sections based on tempo features and fuzzy CMeans [3]. Das *et al.* defined a unified interval type-2 fuzzy rule-based model using a fuzzy histogram and fuzzy co-occurrence matrix to detect cuts and various types of gradual transitions [4]. Based on a singular value decomposition (SVD), Bendraou *et al.* proposed a video segmentation method, which employed the Frobenius norm of low rank approximation matrices, in order to perform an adaptive feature extraction [5]. Dadashi and Kanan detected segmentation points by evaluating a set of fuzzy rules [6]. This method incorporated spatial and temporal features to describe video frames and model cut situations according to temporal dependency of video frames as a set of fuzzy rules. Küçüktunç *et al.* proposed a fuzzy color histogram-based shot-boundary detection method for videos, where heavy transformations would occur [7]. The method was able to detect shot-boundaries using a fuzzy color histogram and extracted a mask for stationary regions for a window of picture-in-picture transformation. However, relying on editing rules limits the rules-based method's ability to process certain genres and certain types of video productions.

B. THE GRAPH-BASED METHOD

The graph-based method uses graphs to represent the arrangement of shots and then partitions the graphs into sections. Rasheed and Shah utilized color and motion features to represents shots similarities, and then split the shot similarity graph (SSG) into sub-graphs by applying normalized cuts for graph partitioning [9]. Sakarya and Telatar proposed a method for video segmentation based on SSG [10], whereby a one-dimensional signal was constructed by graph partitions that were obtained from the similarity matrix in a temporal interval. After filtering each one-dimensional signal, an unsupervised clustering was conducted for finding video scene boundaries. Sidiropoulos *et al.* proposed a method based on a scene transition graph (STG), which can automatically construct multiple STG for each feature that is extracted from the visual and the auditory channel. It then adopts a probabilistic consolidation to calculate results [11]. Mezaris *et al.* proposed two multi-modal automatic scene segmentation techniques, both building upon the STG [12]. The first approach used speaker diarization results to introduce a post-processing step to the STG construction

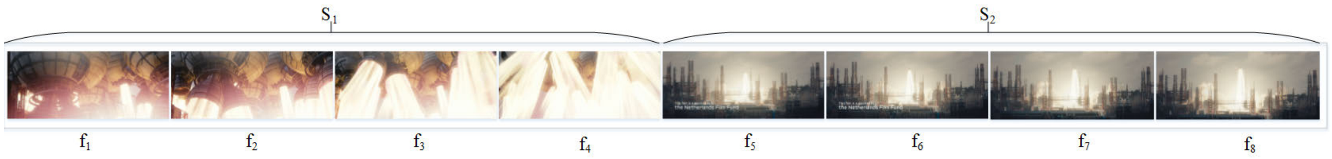


FIGURE 1. An example of video segmentation.

algorithm. In parallel to the original STG based on visual information, the second approach employed speaker diarization and the results of an additional audio analysis to construct a separate audio-based STG. The two STGs are subsequently combined. Xu *et al.* proposed a method [13] that constructed a coherence signal by a graph modal obtained from the similarity matrix in a temporal interval. It then used a STG analysis and audio classification to optimize the signal and then scene boundaries were identified using the window function. However, these methods lack a unified agreement on how to represent a video. This leads to the assumption that the motivation, rationality, and conclusion from the given experiments cannot be shared or migrated between methods [8].

C. THE CLUSTERING-BASED METHOD

The clustering-based method uses the similarities of shots to group frames into meaningful clusters. Baraldi *et al.* proposed a simple spectral clustering technique and achieved comparable results [15]. They also proposed a Siamese network, which was used to learn about the distances between shots, and a spectral clustering was then used to detect coherent sequences [14]. Then they employed the Siamese network to arrange distances between shots into a similarity matrix and obtained the final sections by spectral clustering [16]. Zeng *et al.* proposed a time constraint dominant-set clustering algorithm [18], which is based on the autocorrelogram feature, which is a motion feature with time constraints. Sakai and Imiya proposed a randomized algorithm of spectral clustering and applied it to appearance-based video segmentation [17]. The algorithm exploited random projection and sub-sampling techniques to reduce the dimensionality and cardinality of data. However, these methods ignore the temporal consistency of videos. As videos have obvious time structures and intrinsic sequences, algorithms that do not consider the temporal position and order of video frames, may ultimately divide video frames into non-contiguous segmented sections [8].

In addition, few studies have been identified that have explored the effect of granularity and user intention on video processing. Yang *et al.* proposed a multiple granularity analysis framework for video object segmentation problem, which contains three levels from coarse to fine [20]. Liu *et al.* divided a video into multiple small clips and used the ag-of-words model to describe each clips. It then used the temporally consistent NMF model for clustering and motion segmentation. The authors proposed that two operations MERGE and ADD to allow the user to adjust the results [21]. Oh *et al.* presented a deep learning method for the interactive video object segmentation by allowing user intervention provided in a user-friendly form [22]. They proposed two operations: interaction and propagation. Both two operations are conducted by a CNN.

Finally, it is worth noting that although video segmentation and video summarization are two closely related video processing tasks, they still differ significantly. Video summarization aims to facilitate large-scale video browsing by producing short, concise summaries that are diverse and representative of the original videos. Video segmentation aims to find precise boundaries of semantic sections, which are typically based on a specific concept or a theme usually defined by the user's intention.

III. PROBLEM STATEMENT

Video segmentation is the task of dividing a video into semantic sections temporally. Given the sequence of video frames $S = (f_1, f_2, \ldots, f_n)$, where f_i represents a frame, let $(S_1, S_2, \ldots, S_k)$ denotes a sequence partition of S, the video segmentation refers to the task of finding a partition $S = (S_1, S_2, \ldots, S_e)$ that satisfies the following properties:

1. Unique: a frame must belong to and can only belong to a unique S_i.

2. Continuity: for an arbitrary S_i and an arbitrary fragment $(f_j, f_k) \subseteq S_i$, where $j < k, f_{j+1} \in S_i$ must be satisfied, i.e., the frames in sequence S_i must maintain continuity in the time domain.

3. Local optimum: for a given metric, a frame f_k must be classified into its optimal sequence S_i, i.e., for two consecutive sequences S_i, S_j and a frame f_k belongs to S_i or S_j, f_k should be classified into one of the sequences in which frame f_k gets its optimal partition according to the metric.

As shown in Fig. 1, f_1 to f_8 are frames of a video. Each frame must belong to and only belong to a section. For example, f_1 to f_4 belong to and only belong to S_1, while f_5 to f_8 belong to and only belong to S_2 (Unique property). In every section, if a frame's predecessor and successor frames belong to a same section, the frame also belongs to the same section. For example, since f_2 and f_4 belong to S_1, f_3 must belong to S_1 (Continuity property). For each frame, an evaluation should be conducted on a given metric. Then determine in an optimal manner if it belongs to a new section or the previous section. For example, the result that f_4 belongs to S_1 while f_5 belongs to S_2 should be obtained (Local optimum property).

IV. METHOD

A. OUR FRAMEWORK

Most studies of video segmentation usually convert video into images, then extract features from images and classify them

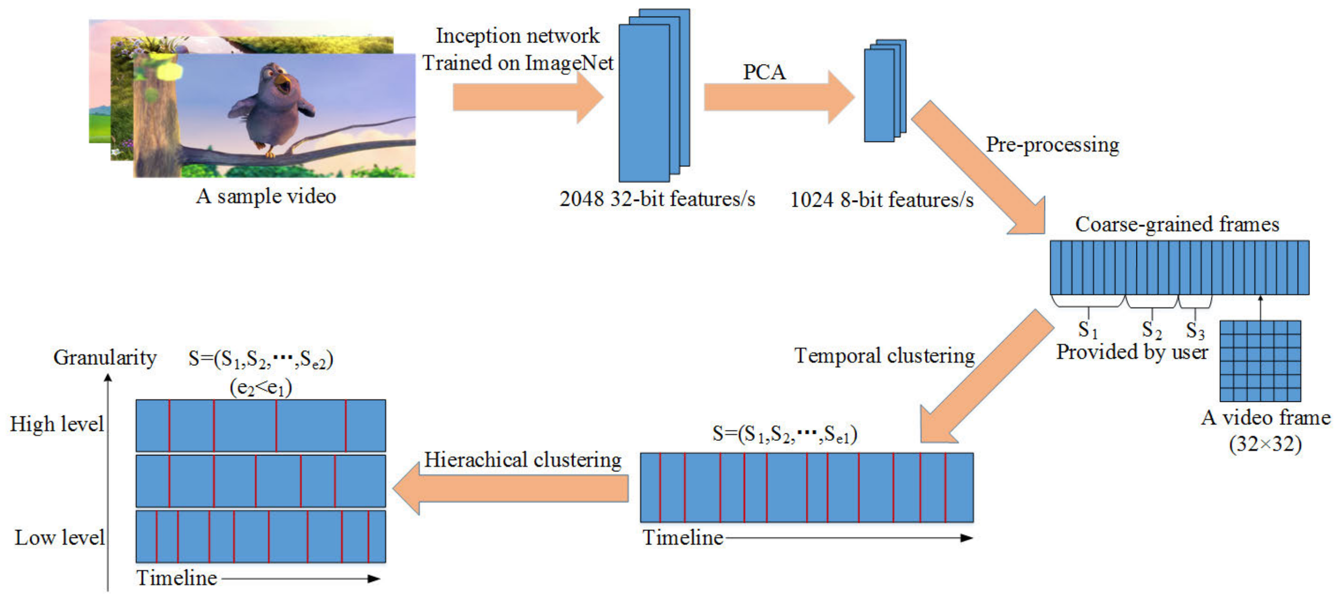


FIGURE 2. An overview of the video segmentation framework. The framework consists of two stages: (1) dimension reduction, (2) temporal clustering. After that, a hierarchical clustering method is implemented to cluster the initial clustering results to higher level.

into semantic sections. These methods are time consuming and computing resources consuming.

In this paper, we present a two-stage video segmentation framework for video stream processing, including dimension reduction stage and temporal clustering stage (Fig. 2). Instead of extracting features frame-by-frame, the dimension reduction stage extracts features in a coarse granularity, which is adopted by YouTube-8M [23] and it can save time and overhead space. The stage employs a deep CNN model pre-trained on ImageNet to encode a video at one-frame-per-second and extract the hidden representation as a 32*32 matrix immediately. For convenience, in the following sections, the frame refers to the frame that was extracted by the deep CNN model per-second, not the original frame.

The temporal clustering method clusters frames on time domain. We noticed that video segmentation has different segmentation results depending on the user's intention. According to this phenomenon, we propose a user-guided temporal clustering method for video segmentation, which is described in section IV-C. Considering that users expect to divide a video into semantic sections at different granularities, we introduce a hierarchical clustering method to provide different segmentation results with levels, which is described in section IV-D.

B. DIMENSION REDUCTION

In video segmentation, several features are commonly used, such as color histogram [24], background similarity [25], [26], spectral features [27], and motion features [28], which are extracted from videos frame by frame. This can result in a lot of time spent and a mass of data generation. Instead of extracting features frame-by-frame, we adopt a dimension reduction method mentioned in [23] to extract features in a coarse granularity, which can save time and overhead space.

The dimension reduction of a video can be described as follows. Firstly, the video is decoded at one-frame-per-second. Then decoded frames are fed into a deep convolutional neural network (CNN) pre-trained on ImageNet [29] — the Inception network, to extract the hidden representation of the decoded frames into 2048-dimensions per second immediately. Finally, the principal components analysis (PCA) is applied to reduce feature dimensions to 1024 (32*32 matrix), followed by quantization (1 byte per coefficient).

As shown in Fig. 2, a video is split to video slices, then decoded into one-second frames to extract 2048-dimensions per second, at last 1024-dimensions per second are obtained by the PCA method.

C. (UGT-CLUSTERING METHOD)

The clustering-based method uses the similarity of shots to group frames into meaningful clusters [8]. Several traditional clustering methods have been widely adopted in video segmentation, for example, k-means, global k-means and kernel k-means [2]. However, these methods all need to pre-specify the number of clusters k. They also never consider the temporal position and the order of frames in videos, which will cause the videos to be divided into non-continuous frames sequences.

Considering that the user's intention has a significant impact on the segmentation results, we propose a user-guided temporal clustering method, which is based on mean-shift clustering method [30]. Considering a set of points in two-dimensional space, mean-shift is a clustering method, which assumes a circular window centered at C and having radius r as the kernel and involves shifting this kernel

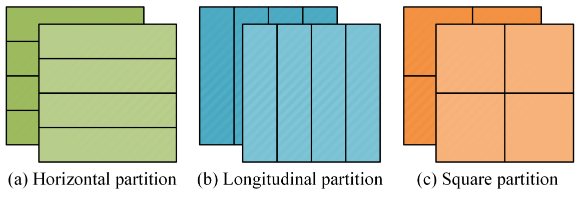


FIGURE 3. Partition means for calculating the inter-frame distance.

iteratively to a higher density region until convergence. However, the mean-shift is a spatial clustering method that does not maintain temporal continuity, i.e., using this method directly will destroy the continuity property (section III).

The UGT-clustering method regresses the cluster radius in chronological order according to the segmentation reference points set by users, and then clusters all data with the regressed radius in chronological order.

The remaining part of this section is organized as follows. (i) We introduce an operator for calculating the similarity distance of inter-frame. (ii) We utilize the operator to regress the cluster radius with the user's intention on time domain. (iii) We propose a temporal clustering method based on the regressed radius.

1) AN OPERATOR FOR CALCULATING THE SIMILARITY DISTANCE OF INTER-FRAME

The key of clustering is calculating the difference between frames. One of the most well known tools for calculating difference is the Euclidean distance. However, the Euclidean distance cannot handle the differences between frames due to frame translation and transformation. In order to handle the difference more accurately, we introduce an operator for inter-frame distance based on the Euclidean distance.

For every frame f_i, we partition it into four blocks by three means, including horizontal partition, longitudinal partition, and square partition (Fig. 3). For each partition, the difference between the frame f_i and mean-point centroid C could be calculated by

$$d(f_i, C) = \sum_{k=1}^{4} \min_j(|f_{ik} - C_j|), \tag{1}$$

where $|f_{ik} - C_j|$ denotes the Euclidean distance between block k of f_i and block j of C, $\min_j()$ means the minimum difference between block k of f_i and every block of centroid C. Among the above three partition means, the minimum $d(f_i, C)$ is chosen as the final difference between frame f_i and centroid C. Through this processing, the operator can handle the translation and transformation of frames, since the horizontal partition and the longitudinal partition is capable of capturing the difference caused by translation, and the square partition is capable of capturing the difference caused by transformation.

2) USER-GUIDED REGRESSION OF THE CLUSTER RADIUS

Considering that the user's intention has a significant impact on the segmentation results, we propose a user-guided regression of the cluster radius based on the segmentation reference points provided by users.

In this regression, m consecutive segmentation reference points on a video should be provided by users, which represent the user's intention about how to segment the video. The goal of regression is obtaining a cluster radius for segmentation by matching the segmentation reference points as much as possible. Here, we conduct a temporal mean-shift regression method using the operator mentioned above to regress the cluster radius.

Given m consecutive segmentation reference points, $m-1$ segmentation sections are formed. Let $S_1, S_2, \ldots, S_{m-1}$, where $m \geq 3$ denote these sections. For every two consecutive sections S_j, S_{j+1}, regress the radius $radius_j$ that will be used to segment S_j and S_{j+1}, where $j, j+1 \in [1, m-1]$ by matching the segmentation reference point split the two consecutive sections as much as possible. For all regressed $radius_j$, select the appropriate radius as the cluster radius. The details are as follows.

1. For two consecutive sections S_j and S_{j+1}, the segmentation reference point spilt these two sections is provided (Fig. 4(1)). We start with the radius regression of the left section S_j, then the radius regression of the right section S_{j+1}. For convenience, let $radius_l$ denotes the regressing radius of S_j, and $radius_r$ denotes the regressing radius of S_{j+1}.

2. Initialize $radius_l$ to the similarity distance of the first two frames of S_j according to (1). Let C denotes the regressing centroid. Initialize C to the mean value of the first two frames of S_j (Fig. 4(2)).

3. For the next frame f_i in S_j, calculate the distance between the frame f_i and the centroid according to (1). There are two situations here, depending on the relationship between the distance and the $radius_l$. (i) If the distance is greater than or equal to $radius_l$ (Fig. 4(3a)), set $radius_l$ to the distance, and update C to the mean value of frames that have been processed (Fig. 4(3b)). (ii) If the distance is less than $radius_l$ (Fig. 4(3a′)), update C to the mean value of frames that have been processed (Fig. 4(3b′)).

4. Once the frames in S_j are all processed like f_i in Fig. 4(3), a test is needed to make sure that $radius_l$ is the minimum threshold that could distinguish S_j from S_{j+1}. Calculate the distance between C and the first frame of S_{j+1} according to (1). If the distance is greater than $radius_l$ (Fig. 4(4)), it means that $radius_l$ is the appropriate radius, which just happens to cluster all the frames in S_j into one group and exclude the first frame in S_{j+1}. If the distance is less than or equal to $radius_l$ (Fig. 4(5)), it implies that $radius_l$ couldn't distinguish S_j from S_{j+1}. Another regression should be conducted to match the segmentation reference point as much as possible. Drop the left half of S_j (now S_j is equal to the right half of the original S_j) and repeat the regression method mentioned about until the radius that is capable of distinguishing S_j from S_{j+1} is found (Fig. 4(6)).

5. Except for the following differences, the radius regression of the right section S_{j+1} is almost the same as the radius regression of S_j. (i) Initialize $radius_r$ to the distance between

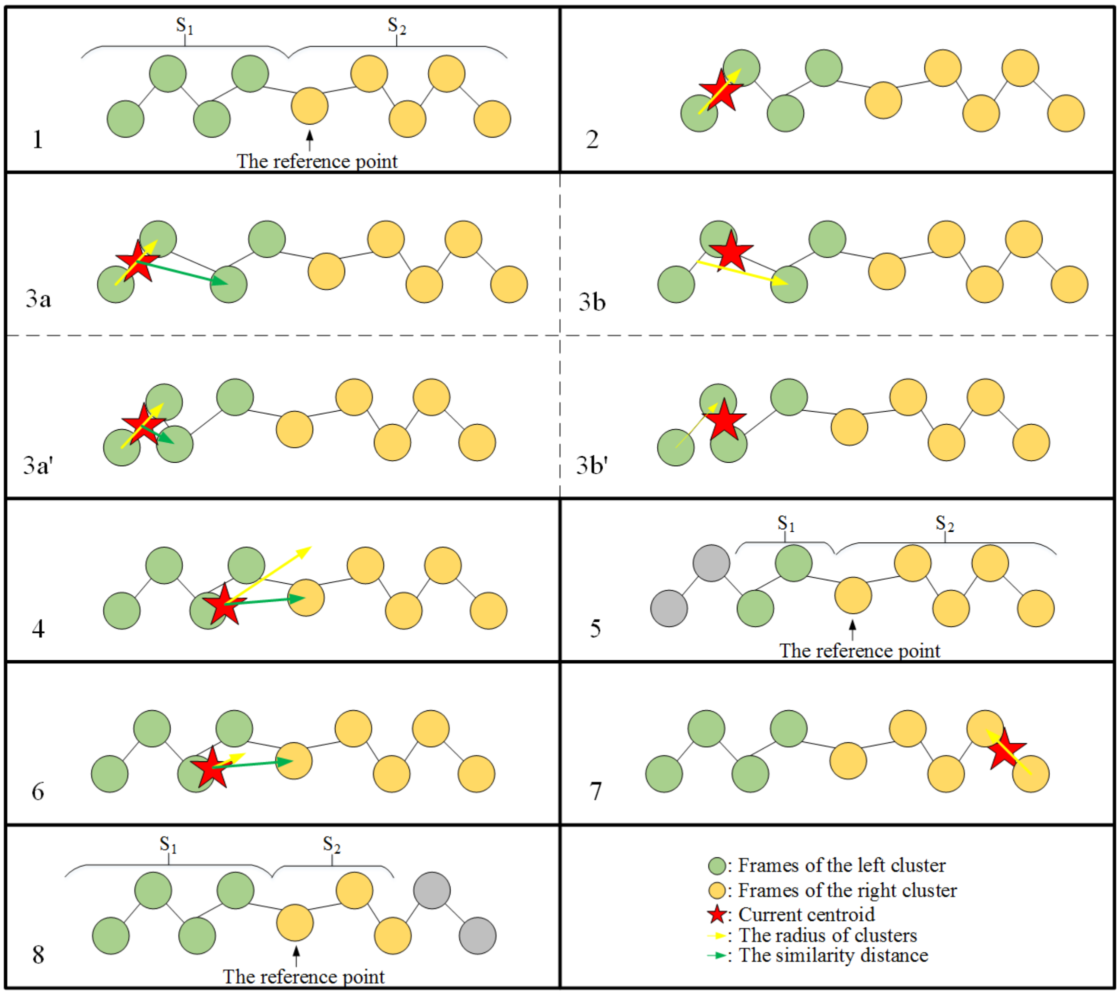


FIGURE 4. The user-guided regression of the cluster radius.

the last two frames of S_{j+1}, and regress $radius_r$ with reverse direction (from the back to the front) (Fig. 4(7)). (ii) When another regression is needed after the test mentioned in step 4, drop the right half of S_{j+1} (Fig. 4(8)).

6. Choose the one with the smaller value of $radius_l$ and $radius_r$ as the regressed radius $radius_j$ of S_j and S_{j+1}.

7. For any two consecutive sections in $(S_i, S_{i+1}, \ldots, S_{i+m-1})$, calculate the regression radius according to the above method, and choose the one with the largest value as the regressed radius of the video.

The complete algorithm is shown in Algorithm 1.

The time complexity of Algorithm 1 is analyzed as follows. Suppose that the number of the consecutive segmentation sections provided by users is m, and the maximum number of video frames in each section is L. The computation time required to find the maximum distance in each section by (1) is $O(L)$ (Line 4 to 9). If the distance does not meet the requirements (Line 10), reduce the section by half and recalculate. Since the maximum number of recalculations is log_2L, the computation time of finding $radius_l$ is $O(L \cdot log_2L)$ (Line 2 to 13). The computation time of finding $radius_r$ is $O(L \cdot log_2L)$, too (Line 14 to 26). For each iteration (Line 1), the two consecutive section are calculated as a group, then the total computation time of Algorithm 1 is $O(2L \cdot log_2L \cdot (m-1))$.

3) TEMPORAL MEAN-SHIFT CLUSTERING

Traditional mean-shift clustering is a non-parametric feature-space analysis technique for locating the maxima of a density function and is widely used in computer vision and image processing. It is a spatial oriented method. But in the field of video segmentation, the property of temporal continuity should be guaranteed first (section III). To maintain this property, we propose a modified mean-shift method named temporal mean-shift clustering that handle the data in chronological order when clustering data.

Considering a sequence of frames on the one-dimensional timeline, our temporal mean-shift clustering assumes a circular window centered at centroid C and having radius r as the kernel. The method shifts the kernel in chronological order iteratively to a higher density region until the difference between a new next frame and the centroid greater than the radius r, where r is calculated by Algorithm 1.

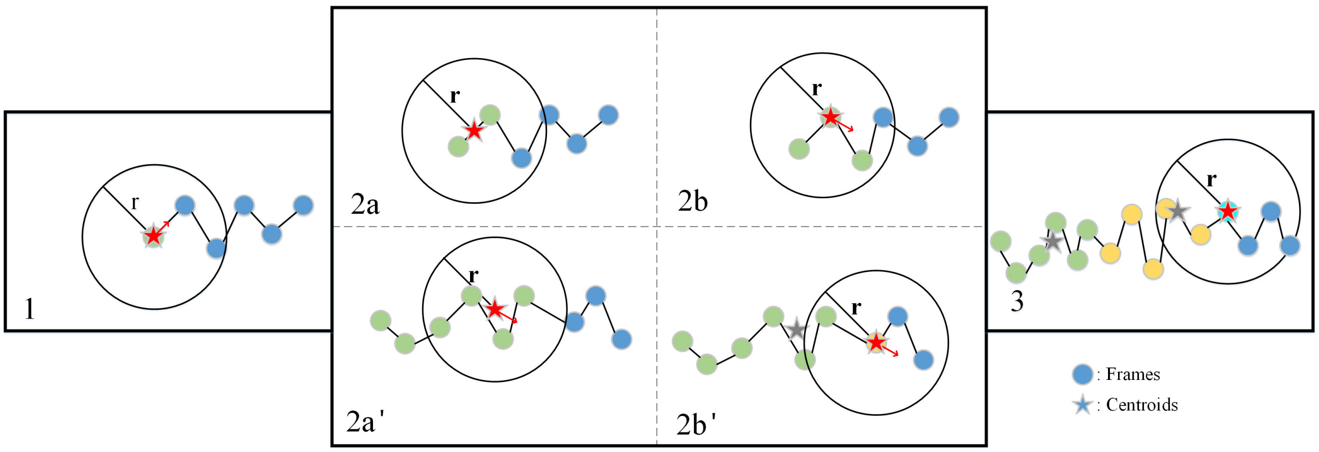


FIGURE 5. The temporal mean-shift clustering.

For a video $S = (f_1, f_2, \ldots, f_n)$ and a regressed radius r, the temporal mean-shift clustering takes as input the frames of S one by one, and returns clusters of frames in chronological order. The main steps are described as follows.

1. Label f_1 as the start frame of a new cluster, initialize the centroid C of the cluster to f_1 and the radius to the regressed radius r (Fig. 5(1)).

2. Shift the kernel by calculating the centroid C as follows.

$$C = \begin{cases} (\sum_{j=s}^{i} f_j)/(i-s+1) & d(f_i, C) \leq r \\ f_i & d(f_i, C) > r \end{cases}, \tag{2}$$

where f_s is the starting frame of the current cluster, f_i is a new next frame, $(\sum_{j=s}^{i} f_j)/(i-s+1)$ denotes the shifted centroid. If $d(f_i, C)$ less than or equal to r, f_i belongs to the current cluster, the centroid will be shifted. Otherwise, a new cluster is started and f_i is the initialized centroid of the new cluster (Local optimum, section III).

In another word, for a new next frame on the timeline, calculate the distance between it and the centroid according to (1). If the distances is less than or equal to r, label the new frame into the same cluster and then update C according to (2) (Fig. 5(2a, 2b)). If the distances is greater than r, label the new frame as a start frame of a new cluster and set C to the new frame according to (2) (Fig. 5(2a$'$, 2b$'$)).

3. Repeat step 2 until the end frame of the video (Fig. 5(3)).

The complete algorithm is shown in Algorithm 2.

The time complexity of the Algorithm 2 is analyzed as follows. Suppose that the number of frames in a video is n. In each iteration, the time complexity required to process the current video frame is $O(1)$ (Line 6 to 13). Therefore, the total time complexity of Algorithm 2 is $O(n)$.

D. HIERARCHICAL CLUSTERING

Sometimes users expect to divide a video into semantic sections at different granularities. we introduce a hierarchical clustering method to present different segmentation results with levels for users.

Given a partition $S' = (S_1, S_2, S_3, \ldots, S_{e1})$ generated by the temporal clustering method described in Algorithm 2, the hierarchical clustering selects a frame as the key frame of S_i according to (3), which has the smallest difference from other frames in S_i.

$$f_{S_i}^* = \arg\min_{f \in S_i}(\sum_{k=1}^{|S_i|} d(f, f_k)) \tag{3}$$

Then, the method takes these frames as a new frame sequence and perform the temporal mean-shift clustering method again with the radius equal to $\alpha \cdot r$, where α is a hyper-parameter set by users. Users can adjust the video segmentation granularity by adjusting α. The details are as follows.

1. Initialize a set RF and a set S'' empty, which denote the representative frame set of S' and the new partition generated by the hierarchical clustering algorithm, respectively.

2. For each segmentation section S_i of S', find the key frame of S_i according to (3) and append it into RF in chronological order. Now RF contains all the representative frames of S'.

3. Call Algorithm 2 on the set RF with the radius parameter $r = \alpha \cdot r$, and then obtain the *result* partition.

4. Generate a new partition S'' of video S according to the *result* partition.

It is worth noting that users can perform the hierarchical cluster algorithm on a same video more than once to obtain multiple partitions at different granularities.

The complete algorithm is shown in Algorithm 3.

The time complexity of the Algorithm 3 is analyzed as follows. Suppose that the number of frames in a video is n, the number of sections to be clustered is m, and the maximum number of frames per these sections is L. The computation time of finding the key frame is $O(m \cdot L^2)$ (Line 3 to 6). The computation time of calling Algorithm 2 is $O(m)$ (Line 7). The computation time of generating a new partition is $O(n)$

Algorithm 1 The User-Guided Regression Algorithm of the Cluster Radius

```
Input: A video $S = (f_1, f_2, \ldots, f_n)$, the consecutive segmentation sections $(S_1, S_2, \ldots, S_{m-1})$ provided by users.
Output: The regressed radius $r$ of the video.
1:  for $S_j, S_{j+1}$ in $(S_1, S_2, \ldots, S_{m-1})$ do
2:      Initialize $radius_l$, $C$ to the similarity distance, the mean value of the first two frames of $S_j$, respectively.
3:      Initialize a test frame $f_t$ to the first frame of $S_{j+1}$.
4:      for $f_i$ in $S_j$ do
5:          if $d(f_i, C) > radius_l$ then
6:              $radius_l \leftarrow d(f_i, C)$
7:          end if
8:          $C \leftarrow$ the mean value of processed frames
9:      end for
10:     if $d(f_t, C) \leq radius_l$ then
11:         $S_j \leftarrow$ the right half of $S_j$
12:         goto Line 2
13:     end if
14:     Initialize $radius_r$ to the similarity distance of the last two frames of $S_{j+1}$.
15:     $C \leftarrow$ the mean value of the last two frames of $S_{j+1}$
16:     $f_t \leftarrow$ the last frame of $S_j$
17:     for $f_i$ in $S_{j+1}$ do
18:         if $d(f_i, C) > radius_r$ then
19:             $radius_r \leftarrow d(f_i, C)$
20:         end if
21:         $C \leftarrow$ the mean value of processed frames
22:     end for
23:     if $d(f_t, C) \leq radius_r$ then
24:         $S_{j+1} \leftarrow$ the left half of $S_{j+1}$
25:         goto Line 14
26:     end if
27:     $radius_j = \min(radius_l, radius_r)$
28: end for
29: $r = \max_j(radius_j)$
30: return $r$
```

Algorithm 2 The Temporal Mean-Shift Clustering Algorithm

```
Input: $S = (f_1, f_2, \ldots, f_n)$, the radius $r$
Output: $S' = (S_1, S_2, \ldots, S_{e1})$
1:  $j \leftarrow 0$
2:  $S_1 \leftarrow (f_1, f_1)$
3:  $C \leftarrow f_1$
4:  $S' \leftarrow [\,]$
5:  for all $f_i$ in $S$ do
6:      if $d(f_i, C) \leq r$ then
7:          Update $C$ according to (2)
8:      else
9:          $S_j \leftarrow (\cdot, f_{i-1})$ {update the right end point of $S_j$}
10:         Append $S_j$ to $S'$
11:         $S_{j+1} \leftarrow (f_i, f_i)$
12:         $C \leftarrow f_i$
13:         $j \leftarrow j + 1$
14:     end if
15: end for
16: return $S'$
```

Algorithm 3 The Hierarchical Cluster Algorithm

```
Input: $S' = (S_1, S_2, \ldots, S_{e1}), \alpha$
Output: $S'' = (S_1, S_2, \ldots, S_{e2})$
1:  $RF \leftarrow [\,]$
2:  $f_i^* \leftarrow$ NIL
3:  for all $S_i$ in $S' = (S_1, S_2, \ldots, S_{e1})$ do
4:      Update $f_i^*$ according to (3)
5:      Append $f_i^*$ to $RF$
6:  end for
7:  $result \leftarrow Algorithm2(RF, \alpha \cdot r)$
8:  $S'' \leftarrow [\,]$
9:  for $S_i$ in $result$ do
10:     $f_s \leftarrow$ the first frame of the section represented by $S_i$
11:     $f_e \leftarrow$ the last frame of the section represented by $S_i$
12:     Append $(f_s, f_e)$ to $S''$
13: end for
14: return $S''$
```

(Line 8 to 13). Therefore, the total time complexity of Algorithm 3 is $O((L^2 + 1) \cdot m + n)$.

V. EVALUATION

A. EVALUATION METRICS

we use coverage C, overflow O and F-score F as evaluation indicators for video segmentation. Coverage C is the number of frames that belong to the same section, which are correctly grouped together. Overflow O is the number of frames that do not belong to the same section and are incorrectly grouped together.

Assume a partition $S = (S_1, S_2, S_3, \ldots, S_m)$ which automatically segmented by the method and the ground truth partition $\tilde{S} = (\tilde{S}_1, \tilde{S}_2, \ldots, \tilde{S}_n)$.

The coverage C_t corresponding to $\tilde{S}_t$ refers to the ratio of the maximum number of overlapped frames between S_i and $\tilde{S}_t$ to the number of frames of $\tilde{S}_t$.

Let the numerator be the amount overlap of S_i that overlaps with $\tilde{S}_t$ with ground truth neighbor sections $\tilde{S}_{t-1}$ and $\tilde{S}_{t+1}$, and the denominator be the number of frames in ground truth neighbor sections $\tilde{S}_{t-1}$ and $\tilde{S}_{t+1}$.The overflow O_t corresponding to $\tilde{S}_t$ refers to the ratio of the numerator and the denominator.

$$C_t = \frac{\max_{i=1,2,\ldots,m} \#(S_i \cap \tilde{S}_t)}{\#(\tilde{S}_t)} \tag{4}$$

$$O_t = \frac{\sum_{i=1}^{m} \#(S_i / \tilde{S}_t) \cdot \min(1, \#(S_i \cap \tilde{S}_t))}{\#(\tilde{S}_{t-1}) + \#(\tilde{S}_{t+1})} \tag{5}$$

where $\#(S_i)$ refers to the number of frames in sections S_i, $S_i/\tilde{S}_t$ denotes frames that belongs to S_i but does not belong to S_t.

TABLE 1. The details of the OVSD dataset.

		Video name				
		ED	BBB	Sintel	TOS	Valkaama
Format type		Avi	Avi	Mp4	Mov	Mp4
Video length	Time	9'22	8'08	12'24	9'48	1:33'05
Frame size	Width	1024	854	1024	1920	640
	Height	576	480	436	800	360
Frame per second	Origin	24	24	24	24	25
	Ours	1	1	1	1	1
Feature type		Frame-level				
Feature size		1024-dimensional				

TABLE 2. The details of TRECVID 2001 dataset.

		Video name			
		anni005	anni009	BOR03	BOR08
Format type		MPEG	MPEG	MPEG	MPEG
Video length	Time	6'19	6'50	26'56	28'07
Frame size	Width	320	320	352	352
	Height	240	240	240	240
Frame per second	Origin	29	29	29	29
	Ours	1	1	1	1
Feature type		Frame-level			
Feature size		1024-dimensional			

After calculating the coverage C_t and overflow O_t of each section S_t, the evaluation values C, O of the video can be calculated as follows:

$$C = \sum_{t=1}^{n} C_t \cdot \frac{\#(\tilde{S}_t)}{\sum \#(\tilde{S}_t)} \tag{6}$$

$$O = \sum_{t=1}^{n} O_t \cdot \frac{\#(\tilde{S}_t)}{\sum \#(\tilde{S}_t)} \tag{7}$$

$$F = \frac{2 \cdot (C + (1 - O))}{C \cdot (1 - O)} \tag{8}$$

where F is the harmonic mean of $1 - O$ and C, which can be used to measure both coverage and overflow in a single metric.

we also use recall, precision and F1 as evaluation metrics defined as follows:

$$R = \frac{N_C}{N_C + N_M} \tag{9}$$

$$P = \frac{N_C}{N_C + N_F} \tag{10}$$

$$F1 = \frac{2 \times R \times P}{R + P} \tag{11}$$

where N_C, N_M, and N_F denote the number of boundaries detected correctly, missed and detected falsely. F_1 is a general measurement considering both recall and precision. For $F1$, higher value means better performance.

B. EXPERIMENTAL DATA

We use both the Open Video Scene Detection (OVSD) dataset [31] and the TRECVID 2001 dataset [32] as our experimental data.

The Open Video Scene Detection (OVSD) dataset is an open dataset for the evaluation of video scene detection algorithms, which consists of short films and an open full length film. Table 1 shows the details of the OVSD dataset, including the length, format type, frame size, and so on.

To compare with the existing method, we also evaluate the proposed method on the TRECVID 2001 dataset. The TRECVID 2001 dataset was provided by the National Institute of Standards and Technology (NIST) in 2001, which devoted to research in automatic segmentation, indexing, and content-based retrieval of digital video. However, we found that there are quite a few errors in the annotations of some videos. Hence we only choose four videos with correct annotations as test dataset, which is described in Table 2.

For both datasets, we adopt the same feature extraction method used in YouTube-8M [23]. The method employs a deep CNN model pre-trained on ImageNet to encode a video at one-frame-per-second, and then to extract the hidden representation of these one-second frames as a 32*32 matrix.

C. EXPERIMENTAL RESULTS

1) EVALUATION OF THE PROPOSED METHOD

We compare the F-score, Coverage and Overflow of the following three methods: (1) temporal mean-shift clustering without the operator, (2) temporal mean-shift clustering with the operator, (3) temporal mean-shift clustering with the operator and histogram equalization. Histogram equalization is a method in image processing of contrast adjustment using the image's histogram [35]. Table 3 shows the experiment results of the videos, which are encoded into a format of one-frame-per-second described in section V-B.

The most interesting finding is that the F-score values of the above methods are at an acceptable level (Table 3). It indicates that we can adopt coarse-grained feature extraction (for example, one frame per second) for video segmentation in most situations that do not need accurate video segmentation.

As shown in Table 3, method (2) and method (3) are superior to method (1). Among them, method (2) achieves the best results in Valkaama, and method (3) achieves the best results in ED, Sintel, BBB, and TOS. It is worth noting that method (2) is improved by 14.6% and method

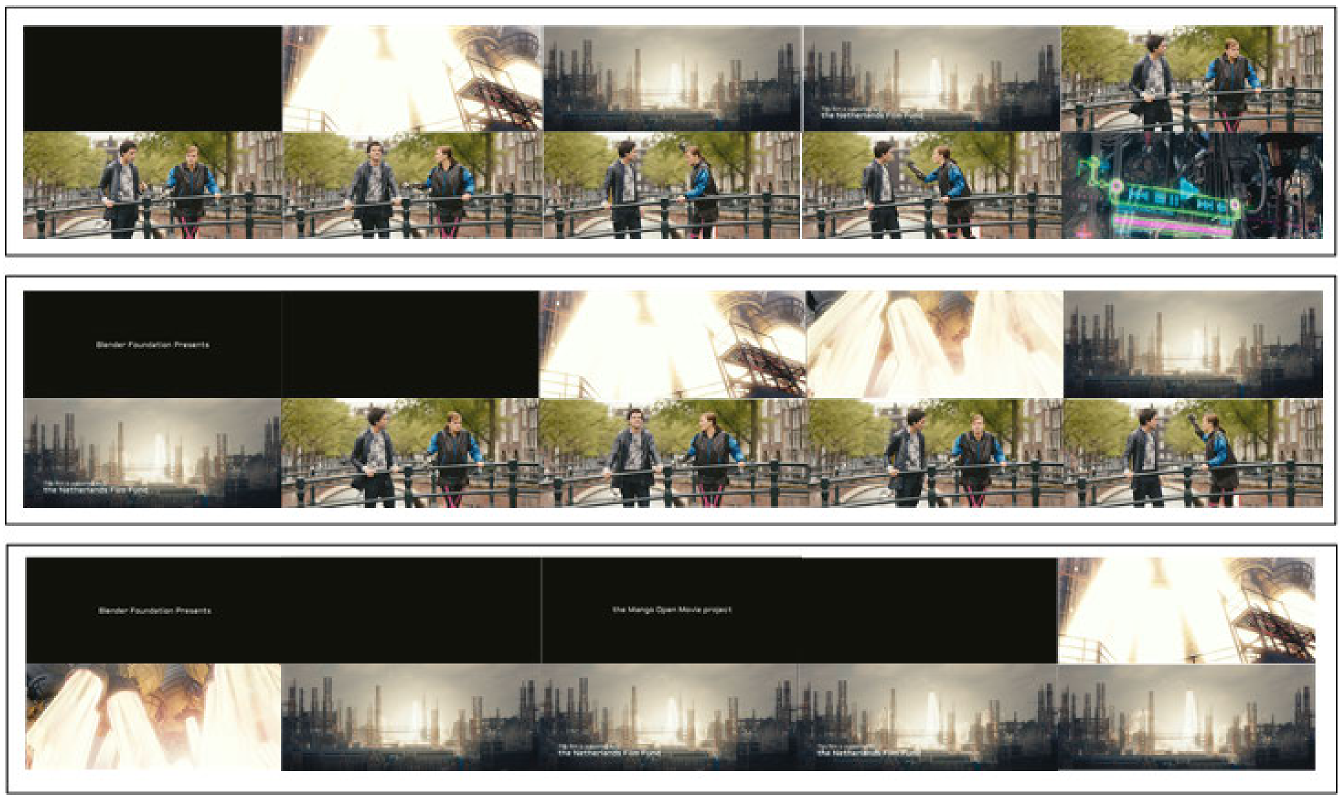

Blender Foundation Presents
the Mango Open Movie project
the Netherlands Film Fund

FIGURE 6. The results of user-guided temporal clustering. By selecting different reference segmentation points, the segmentation granularity is getting finer and finer from the top line to the bottom line.

TABLE 3. The evaluation of the proposed method.

		ED	BBB	Sintel	TOS	Valkaama
(1)	F	0.485	0.716	0.694	0.751	0.762
	C	0.670	0.662	0.673	0.603	0.808
	O	0.620	0.220	0.282	0.006	0.279
	T*	1.048	1.028	2.048	1.012	5.067
(2)	F	0.556	0.726	0.679	0.739	**0.764**
	C	0.428	0.678	0.685	0.588	0.801
	O	0.211	0.221	0.327	0.006	0.270
	T*	4.076	4.000	6.048	3.035	9.046
(3)	F	**0.588**	**0.752**	**0.735**	**0.753**	0.749
	C	0.451	0.746	0.717	0.607	0.803
	O	0.156	0.240	0.245	0.006	0.298
	T*	5.082	4.015	8.085	3.066	12.083

* T represents the actual execution time of the algorithm (seconds).

(3) is improved by 21.2% compared to method (1) in the F-score value of ED. These results suggest that methods with the operator perform better. A possible explanation for this might be that the operator can handle the frequent shots switching (for example, ED) by dividing the frame into blocks.

Fig. 6 presents the experimental results of user-guided temporal clustering with different reference segmentation points set by users. By giving different reference segmentation points, our method will fit the user's intention as much as possible. Different reference segmentation points will result in different segmentation results. For example, the segmentation granularity in Fig. 6 is getting finer and finer from top to bottom.

Fig. 7 displays the experimental results of hierarchical clustering. The hierarchical clustering method is another tool to help users to segment videos by clustering the segmentation results repeatedly. As shown in Fig. 7, the hierarchical clustering method clusters the original segmentation results into semantic sections with a higher abstraction level.

2) COMPARISON WITH EXISTING METHODS

We compare the F1-score, recall and precision of [33], [34] as well as our proposed method. Table 4 shows the experiment results of the videos, which are encoded into a format of one-frame-per-second described in section V-B.

As shown in Table IV, the proposed method achieves almost the same results as [34], and is significantly higher than [33]. Specifically, the proposed method achieves the

TABLE 4. Performance comparison on TRECVID 2001 dataset.

File	Proposed Method			Gianluigi's Method [33]			Ma's Method [34]		
	R	P	F	R	P	F	R	P	F
anni005	**0.848**	0.761	0.802	0.434	0.472	0.452	0.820	**0.865**	**0.841**
anni009	**0.770**	0.644	0.702	0.497	0.804	0.614	0.675	**0.810**	**0.736**
BOR03	0.782	0.779	**0.781**	0.405	**0.962**	0.569	**0.878**	0.618	0.725
BOR08	0.714	0.779	0.745	0.252	0.745	0.376	**0.861**	**0.849**	**0.855**

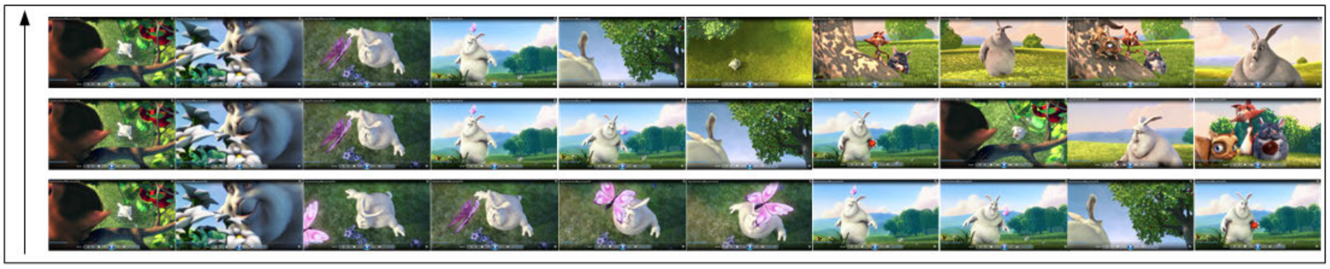


FIGURE 7. The results of hierarchical clustering.The top line shows the original segmentation results generated by temporal mean-shift clustering with operator; the mid line shows the results after hierarchical clustering 3 times with $\alpha = 0.95$; the bottom line shows the results after hierarchical clustering 5 times with $\alpha = 0.95$.

best results in BOR03, while [34] achieves the best results in anni005, anni009 and BOR08. Considering the following facts: (1) the proposed method extracts features in a coarse granularity by decoding a video one-frame-per-second, (2) [33] and [34] extract features in a fine granularity by decoding a video frame by frame, the results of the proposed method are quite attractive.

VI. DISCUSSION

Nowadays, automatic video processing tools are becoming more and more important because the amount of video data is growing dramatically. In these tools, feature extraction is a necessary step, which are now commonly using end-to-end deep learning algorithms to automatically extract features. However, extracting features from videos in a frame by frame manner can result in a significant amount time spent undertaking the process and masses amount of data generation. In order to improve the efficiencies of video processing, it is necessary to study the video processing method on coarse-grained feature extractions.

This is the first known study where the video segmentation method has been implemented on coarse-grained feature extraction. An evaluation of the OVSD dataset demonstrates that the average F-score achieved by our proposed method on coarse-grained feature extraction (one-frame-per-second) is 0.72. This result indicates that video segmentation based on a coarse-grained feature extraction is a more economical and promising method, when accurate segmentation is not required, or when fast and iterative segmentations are needed in accordance with the user's intention.

This paper proposed a user-guided regression method based on the segmentation reference points provided by users, in order to match the user's intention regarding video segmentation. However, how to represent the intention of user is still an open question. It is obvious that the method used in this study is not the only way to take the user's intention into consideration. User intention will play an important role in video segmentation in the future, when the demand for semi-automated video creation increase, and especially since there are no standards or specifications in most video segmentation cases. Therefore, determining how to represent and utilize the user's intention is still an important issue for future research.

An operator for calculating the similarity distance between coarse-grained frames was also proposed in this study. The operator takes into account the effects of frame translation and transformation, and it is influential on the segmentation result (Table 3). However, determining how to calculate the similarity distances between frames in combination with the user's intention is still worthy of further discussion.

VII. CONCLUSION

This paper investigated automatic video segmentation under coarse-grained feature extractions. The investigations conducted during the course of this research are based on user-guided regression of the cluster radius, which takes the user's intention into account. A two-stage video segmentation framework for video stream processing was presented, which included dimension reduction and temporal clustering. The framework makes it possible to segment videos based on coarse-grained feature extraction. A UGT clustering method was also proposed, which utilizes user intention information to segment video on time domain, which the segmented results reflecting the user's intention. Finally, the proposed hierarchical clustering method allows users to perform video segmentation at different granularities, such that users can choose the results from a low level to a higher abstraction level.

REFERENCES

[1] D. Rotman, D. Porat, G. Ashour, and U. Barzelay, "Optimally grouped deep features using normalized cost for video scene detection," in *Proc. Int. Conf. Multimedia Retr.*, Jun. 2018, pp. 187–195.

[2] V. T. Chasanis, A. C. Likas, and N. P. Galatsanos, "Scene detection in videos using shot clustering and sequence alignment," *IEEE Trans. Multimedia*, vol. 11, no. 1, pp. 89–100, Jan. 2009.

[3] M. Ellouze, N. Boujemaa, and A. M. Alimi, "Scene pathfinder: Unsupervised clustering techniques for movie scenes extraction," *Multimedia Tools Appl.*, vol. 47, no. 2, pp. 325–346, Apr. 2010.

[4] S. Das, S. Sural, and A. K. Majumdar, "Detection of hard cuts and gradual transitions from video using fuzzy logic," *Int. J. Artif. Intell. Soft Comput.*, vol. 1, no. 1, pp. 77–98, Nov. 2008.

[5] Y. Bendraou, F. Essannouni, D. Aboutajdine, and A. Salam, "Shot boundary detection via adaptive low rank and svd-updating," *Comput. Vis. Image Understand.*, vol. 161, pp. 20–28, Aug. 2017.

[6] R. Dadashi and H. R. Kanan, "AVCD-FRA: A novel solution to automatic video cut detection using fuzzy-rule-based approach," *Comput. Vis. Image Understand.*, vol. 117, no. 7, pp. 807–817, Jul. 2013.

[7] On. Küçüktunç, U. Güdükbay, and Ö. Ulusoy, "Fuzzy color histogram-based video segmentation," *Comput. Vis. Image Understand.*, vol. 114, no. 1, pp. 125–134, Jan. 2010.

[8] L. Baraldi, C. Grana, and R. Cucchiara, "Recognizing and presenting the storytelling video structure with deep multimodal networks," *IEEE Trans. Multimedia*, vol. 19, no. 5, pp. 955–968, May 2017.

[9] Z. Rasheed and M. Shah, "Detection and representation of scenes in videos," *IEEE Trans. Multimedia*, vol. 7, no. 6, pp. 1097–1105, Dec. 2005.

[10] U. Sakarya and Z. Telatar, "Video scene detection using graph-based representations," *Signal Process., Image Commun.*, vol. 25, no. 10, pp. 774–783, 2010.

[11] P. Sidiropoulos, V. Mezaris, I. Kompatsiaris, H. Meinedo, and I. Trancoso, "Temporal video segmentation to scenes using high-level audiovisual features," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 8, pp. 1163–1177, Aug. 2011.

[12] P. Sidiropoulos, V. Mezaris, I. Kompatsiaris, H. Meinedo, and I. Trancoso, "Multi-modal scene segmentation using scene transition graphs," in *Proc. 17th ACM Int. Conf. Multimedia*, Oct. 2009, pp. 665–668.

[13] S. Xu, B. Feng, P. Ding, and B. Xu, "Graph-based multi-modal scene detection for movie and teleplay," in *Proc. IEEE Int. Conf. Acoustics, Speech Signal Process. (ICASSP)*, Mar. 2012, pp. 1413–1416.

[14] L. Baraldi, C. Grana, and R. Cucchiara, "A Deep Siamese Network for Scene Detection in Broadcast Videos," in *Proc. 23rd ACM Int. Conf. Multimedia*, Oct. 2015, pp. 1199–1202.

[15] L. Baraldi, C. Grana, and R. Cucchiara, "Analysis and re-use of videos in educational digital libraries with automatic scene detection," in *Proc. 11th Italian Res. Conf. Digit. Libraries*, Jan. 2016, pp. 155–164.

[16] L. Baraldi, C. Grana, A. Messina, and R. Cucchiara, "A browsing and retrieval system for broadcast videos using scene detection and automatic annotation," in *Proc. ACM Multimedia Conf.*, Oct. 2016, pp. 733–734.

[17] T. Sakai and A. Imiya, "Randomized algorithm of spectral clustering and image/video segmentation using a minority of pixels," in *Proc. IEEE 12th Int. Conf. Comput. Vis. Workshops, (ICCV) Workshops*, Sep./Oct. 2009, vol. 468, no. 2, pp. 468–475.

[18] X. Zeng, X. Zhang, W. Hu, and W. Li, "Video scene segmentation using time constraint dominant-set clustering," in *Advances in Multimedia Modeling* (including Lecture Notes in Computer Science), vol. 5916. Berlin, Germany: Springer, 2009, pp. 637–643.

[19] A. Ratner, P. Varma, B. Hancock, and C. Ré, "Weak supervision: A new programming paradigm for machine learning," Stanford, CA, USA, Tech. Rep. 2019-3-10, 2019.

[20] R. Yang, B. Ni, C. Ma, Y. Xu, and X. Yang, "Video segmentation via multiple granularity analysis," in *Proc. 30th IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3010–3019.

[21] H. Liu, F. Sun, X. Zhang, and B. Fang, "Interactive video summarization with human intentions," *Multimedia Tools Appl.*, vol. 78, no. 2, pp. 1737–1755, Jan. 2019.

[22] S. W. Oh, J.-Y. Lee, A. Research, and N. Xu, "Fast user-guided video object segmentation by deep networks," in *Proc. CVPR*, 2018, vol. 1, no. 1, pp. 1–3.

[23] M. Akopyan and E. Khashba, "Large-scale youtube-8m video understanding with deep neural networks," *CoRR*, vol. abs/1706.04488, pp. 1–7, Jun. 2017.

[24] J. Li, T. Yao, Q. Ling, and T. Mei, "Detecting shot boundary with sparse coding for video summarization," *Neurocomputing*, vol. 266, pp. 66–78, Nov. 2017.

[25] A. Aner and R. John Kender, "Video summaries through mosaic-based shot and scene clustering," in *Proc. Eur. Conf. Comput. Vis.*, 2002, pp. 388–402.

[26] M. Yeung, B.-L. Yeo, and B. Liu, "Segmentation of video by clustering and graph analysis," *Comput. Vis. Image Understand.*, vol. 71, no. 1, pp. 94–109, Jul. 1998.

[27] H. Liu and H. Zhang, "The segmentation of news video into story units," in *Advances in Web-Age Information Management* (Lecture Notes in Computer Science), vol. 3739. Berlin, Germany: Springer, 2005, pp. 870–875.

[28] Y.-M. Kwon, C.-J. Song, and I.-J. Kim, "A new approach for high level video structuring," in *Proc. IEEE Int. Conf. Multimedia Expo*, Jul./Aug. 2000, pp. 773–776.

[29] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and F. F. Li, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[30] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 8, pp. 790–799, Aug. 1995.

[31] D. Rotman, D. Porat, and G. Ashour, "Robust and efficient video scene detection using optimal sequential grouping," in *Proc. IEEE Int. Symp. Multimedia (ISM)*, Dec. 2016, pp. 275–280.

[32] F. Alan Smeaton, P. Over, and W. Kraaij, "Evaluation campaigns and TRECVid," in *Proc. 8th ACM Int. Workshop Multimedia Inf. Retr.*, Oct. 2007, pp. 321–330.

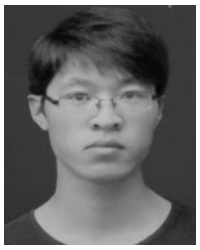
[33] C. Gianluigi and S. Raimondo, "An innovative algorithm for key frame extraction in video summarization," *J. Real-Time Image Process.*, vol. 1, no. 1, pp. 69–88, Mar. 2006.

[34] Y.-F. Ma, J. Sheng, Y. Chen, and H. J. Zhang, "MSR-Asia at TREC-10 video track: Shot boundary detection task," in *Proc. 10th Text Retr. Conf. (TREC)*, Gaithersburg, MD, USA, Nov. 2001.

[35] T. Acharya and A. K. Ray, "Image processing: Principles and applications," *IEEE Trans. Neural Netw.*, vol. 18, no. 2, p. 9227, Aug. 2005.

XINHUI PENG received the B.S. degree from the Guilin University of Electronic and Technology, in 2016. She is currently pursuing the master's degree in computer science and electronic engineering with Hunan University, China. She is a member of the Key Laboratory for Embedded and Network Computing of Hunan Province, China. Her major research interests include machine learning and multimedia.

RUI LI received the Ph.D. degree in computer science from Hunan University, China, in 2007, where he is currently an Assistant Professor of computer science and electronic engineering. His major research interests include machine learning, embedded and real-time systems, and computer vision.



JILONG WANG received the B.S. degree from the Hunan University of Science and Technology, in 2016. He is currently pursuing the Ph.D. degree in computer science and electronic engineering with Hunan University, China. He is a member of the Key Laboratory for Embedded and Network Computing of Hunan Province, China. His major research interests include machine learning, streaming data, and cyber-physical systems.

HAO SHANG received the B.S. degree from Hubei Normal University, in 2016. He is currently pursuing the master's degree in computer science and electronic engineering with Hunan University, China. He is a member of the Key Laboratory for Embedded and Network Computing of Hunan Province, China. His major research interests include machine learning, computer vision, and deep learning.