Adaptive Subspace Symbolization for Content-based Video Detection

Xiangmin Zhou, Xiaofang Zhou, Senior Member, IEEE, Lei Chen, Member, IEEE, Yanfeng Shu, Athman Bouguettaya, Senior Member, IEEE and ACM and John A. Taylor,

Abstract—Efficiently and effectively identifying similar videos is an important and non-trivial problem in content-based video retrieval. This paper proposes a subspace symbolization approach, namely SUDS, for content-based retrieval on very large video databases. The novelty of SUDS is that it explores the data distribution in subspaces to build a visual dictionary with which the videos are processed by deriving the string matching techniques with two-step data simplification. Specifically, we first propose an adaptive approach, called VLP, to extract a series of dominant subspaces of variable lengths from the whole visual feature space without the constraint of dimension consecutiveness. A stable visual dictionary is built by clustering the video keyframes over each dominant subspace. A compact video representation model is developed by transforming each keyframe into a word that is a series of symbols in the dominant subspaces, and further each video into a series of words. Then, we present an innovative similarity measure called CVE, which adopts a complementary information compensation scheme based on the visual features and sequence context of videos. Finally, an efficient two-layered index strategy with a number of query optimizations is proposed to facilitate video retrieval. The experimental results demonstrate the high effectiveness and efficiency of SUDS.

Index Terms—Video detection, subspace symbolization, variable length partition, query optimization.

1 INTRODUCTION

CURRENT commercial web search engines, such as Google Video and Youtube, index videos and search them successfully by text. The search problem that these video search engines grapple with is not a CBVR problem, since all these systems lack in the understanding of the rich media content. It is valuable to have practical solutions on content based video retrieval for its wide applications. In CBVR, a special problem is to find the similar videos duplicated to each other for database consolidation or copyright enforcement. It defines similar videos as the clips with almost identical content but possibly compressed at different qualities, reformatted to different sizes and frame-rates, undergone minor editing in either spatial or temporal domain, or summarized into keyframe sequences [9]. In more recent studies, videos with different variations such as contrast enhancement are also considered as similar [34].

To retrieve similar videos efficiently and effectively, several key issues need to be addressed. First, a video is required to be represented compactly and informatively. This issue is important, as a video typically contains a number of keyframes, and the similarity between two videos is usually measured by finding the closest match for every single keyframe of them [34]. Thus, searching large databases over raw video data is computationally expensive. The second issue is how to measure the similarity between videos based on their summaries. We need a distance function that can effectively capture the (dis)similarity between two videos. In addition to the visual features, a good similarity measure should take into account the temporal and sequence context inherent in videos as well. The third issue is how to index video summaries. Sequentially scanning all summaries to identify similar videos from a large database is clearly undesirable. Efficiently indexing video summaries is necessary to avoid unnecessary space access and costs.

A variety of techniques have been developed for effectively matching videos and compressing video representations in large datasets [1], [3], [9], [22], [27], [35]. Considering the temporal information, edit distance variants have been proved to be effective among the existing similarity measures [1], [3], [6], [7]. However, the computational complexity of them is very high. Given two videos containing \( m \) and \( n \) keyframes respectively, the edit distance computation cost between them is \( mn \), thus inappropriate for the high dimensional video sequences. Although existing works represent each video by their keyframes [27] or quantizing each dimension of the keyframe into a symbol [1] to compact the video data and improve the efficiency of video matching, they describe a keyframe as a \( D \)-dimensional vector or \( D \) symbols, where \( D \) is the dimensionality of the original video.
frames. Thus, neither of these representations reduces the complexity of each single keyframe. Meanwhile, while each object is transformed into a single symbol by quantizing the videos into clusters over the whole space [31], [35], this keyframe quantization method tends to produce an unstable visual vocabulary of large size, which is undesirable for fast video retrieval. More efficient query processing techniques are required for fast video identification. In [38], we propose a basic solution of SUBspace Descriptor Symbolization(SUDS) for effective and efficient video detection. The major novelty of SUDS is that it explores the data distribution in subspaces to construct a high-quality visual dictionary, with which each video is transformed into a highly compact word sequence. Being symbolic, it allows the use of the matching approaches in text retrieval and facilitates the efficiency of video identification. Compared to the basic framework proposed, in this paper, we make the following extensions. An adaptive subspace partition strategy is first proposed to optimize the subspace partition and improve the visual dictionary, so a video keyframe can be represented as a more compact word with maximal discrimination power. The video query processing is further optimized by effective query summarization and filtering strategies. Moreover, a complementary similarity measure guarantees the high effectiveness of video detection. To summarize, in this paper, we made the following contributions:

- We first propose an adaptive subspace partition strategy named Variable Length Partition (VLP) to produce a set of subspaces of varying lengths such that their discrimination power is maximized. A visual dictionary is then constructed by clustering the video keyframes over the dominant subspaces that are of maximal variances over their dimensions. Each cluster in a dominant subspace is represented as a unique id. By looking up this visual dictionary, a keyframe is then mapped into a word that is a series of ids, and a video into a sequence of words.
- We present a novel distance measure called Complimentary Video Edit distance (CVE) to serve as the similarity measure. CVE takes both the visual and context information of videos into consideration by deriving the conventional edit distance into video matching and embedding the sequence context difference into the distance measure.
- We propose an effective two-layer indexing structure. In the first layer, the symbols in a dominant subspace are indexed by the optimal one dimensional indexing method using a B+ -tree [30]. All the symbols in the whole space are indexed by multiple B+ -trees. In the second layer, an inverted file is used to link each video symbol to its occurrence place.
- We further develop a number of query optimization methods, including effective query summarization and candidate sequence filtering strategies, to improve search performance. We have conducted an extensive performance study over large video database. The experimental results demonstrate the efficiency and effectiveness of SUDS.

The rest of the paper is organized as follows: Section 2 reviews the related work; The video data modelling is described in Section 3; Section 4 presents our video indexing structure and strategy, followed by the video query processing techniques in Section 5. Section 6 presents our performance study and reports the experimental results. Finally Section 7 concludes the paper.

2 RELATED WORK

We focus on three topics in content based video retrieval that are most relevant to our work, including video representation, video matching and video indexing.

Video Representation. Video representation primarily aims at reducing the temporal or spatial redundancy of video data. A popular video representation technique is to represent each video sequence through its keyframes [12], [34], [40]. Several other video models have been proposed, and they offer various simplifications targeting efficient video sequence matching. Examples include MBR [22], ViTri [30], ViSig [9], [8], BCS [16] and the SVD-based motion video summary [14] etc. Visual vocabulary/dictionary has been proposed for efficient video retrieval by quantizing the video descriptions into clusters that can be managed with the techniques used in text retrieval [1], [35], [31], [39], [26], [25]. Instead of directly matching each high dimensional vector pair, it finds matched pairs by using visual key words that can be accessed from a visual vocabulary.

In [1], authors proposed a symbol representation model based on dimension-wise quantization, called VString. The real-valued features are transformed into some discrete classes. Each class is described as a symbol, and a set of all symbols form the alphabet. Each dimension of a feature is represented as a symbol, accordingly, each feature vector as a multidimensional video string. The similarity between videos is measured by VString edit distance. While this work introduces edit distance to video matching, it transforms each D-dimensional visual feature into D symbols, thus the representation is not compact. Unlike dimension-wise symbolization, in [31], authors construct a visual dictionary by clustering the visual objects over the global space, as such a high dimensional object is mapped into one single symbol. In some recent works, the idea of visual vocabulary over global space is extended by combining context information. For example, in [35], the semantic context of a keyframe refers to the story text transcripts extracted through speech recognition in the audio track, which is utilized with the visual key words jointly, and provides a meaningful cue for near duplicate keyframe retrieval. In [39], we proposed to combine the sequence relationship between neighboring frame samples and the keyframe symbolization over global space. However, due to the high dimensionality of video data, this global
space symbolization tends to produce a large visual dictionary with poor-quality clusters, which raises a big challenge for effective and efficient video detection.

**Video Matching.** An important topic in video matching is to design a distance function, such that the similarity between videos is effectively determined. Various distance measures have been developed [22], [30], [6], [7], [1], [3], [32], [42], [36]. Among them, edit distance variants have been proved to be effective, since they can better deal with the sequence alignment and temporal ordering that are inherent in videos.

Edit distance variants [6], [7], [1], [3], [23], [39], [24] are proposed by extending the conventional edit distance [21], which is widely used in string matching, for video sequence matching. In [6], Edit distance with Real Penalty (ERP) was proposed to support the local time shifting in a metric measure by a marriage of L1-norm and the edit distance. Given two sequences $S_1$ and $S_2$, the ERP between them is defined as the total cost of insert, delete or replace operations that are required to change $S_1$ to $S_2$, where the cost of each operation is decided by the L1-norm distance between the corresponding element pair. In [7], Edit Distance on Real sequence (EDR) was proposed by taking into account the effect of noise and local time shifting in measurement. Although the ERP and EDR methods are focused on the similarity-based search of moving object trajectories, they can also be applied to video identification. Vstring edit distance was proposed by transforming the problem of video sequence matching into that of pattern matching [1], such that the techniques for string matching can be directly applied. However, the main problem with them lies in their high computational complexity. The matching cost is exponential to the lengths of sequences, which is inappropriate to matching the video sequences consisting of high dimensional vectors. Although other video matching methods proceed by using local alignment to find the sequences of similar values that are compact video signatures [32], [42], they can only process the matching between video copies with the same matching complexity as the edit distance variants.

**Video Indexing.** Many indexing approaches to organizing high dimensional data have been proposed [5], [20], [33], [13], [37], [19], [2], [15], [29], [10], [17], [30], [11]. The conventional multidimensional index structures directly store the high dimensional data by space partitioning [29] or data partition [10]. Recently, some indexing methods [17], [30] like iDistance have been proposed to effectively manage the data in high dimensional metric space by transformation. Using these methods, high-dimensional data are first mapped into single values, and one-dimensional indexing methods such as B$^+$-tree are then utilized. In the iDistance, the high-dimensional vectors are transformed into single values by computing their distances from a selected reference point. However, CVE is a non-metric measure, with which these metric indexes can not be applied directly.

3 **VIDEO SYMBOLIZATION**

We first present how to build a dictionary, by which each video is symbolized, followed by the similarity measure between two videos over the symbol space. Finally, we analyze the dictionary size and the key parameters in dictionary construction and similarity measure.

3.1 **Visual Dictionary Construction**

Before proceeding to the detailed strategies of visual dictionary construction, we first introduce five important conceptions, Subspace, Sub-descriptor, Discrimination Power, Dominant Subspace and Miss Ratio.

**Definition 1:** Given a D-dimensional feature space, a subspace is a projected space over $d$ dimensions of this D-dimensional space, where $d < D$. Given a keyframe, the projected feature over a subspace is its sub-descriptor.

**Definition 2:** Given a set of keyframes, $\Sigma: <f_1 \ldots f_n>$, let $C$ be the center of $\Sigma$, $d_i$ denote a segment connecting $C$ and $f_i$, the discrimination power of a subspace $j$ is defined as the length standard deviation of all $d_i$ over it, denoted as $SDPower(j)$.

$$SDPower(j) = \left(\frac{\sum_{i=1}^{n} (Proj(d_i))^2 - n \sum_{i=1}^{n} \text{Proj}(d_i)^2}{n}\right)^{1/2}$$

where $j$ refers to the id of a subspace, $\text{Proj}(d_i)$ is the projected segment length of $d_i$ over the $j^{th}$ subspace.

**Definition 3:** A subspace holding the maximum discrimination power is called dominant subspace. A 1-dimensional dominant subspace is also called dominant dimension.

**Definition 4:** Suppose that the subspaces are ranked by their discrimination powers, we define the first $\gamma$ subspaces as dominant ones, then the discrimination power of all dominant subspaces is: $SDPower = \sum_{j=1}^{\gamma} SDPower(j)$.

We represent each keyframe as a word that is a series of symbols. Each symbol is a cluster id in a dominant subspace. In [38], to compare the words of two keyframes, we assume that they are matched if they are identical, and unmatched otherwise. However, since the neighboring clusters in multidimensional space may be overlapped with each other, two similar sub-descriptors falling into the overlapping part of clusters may be represented as different symbols, thus misjudged as dissimilar ones. Accordingly, the matched keyframes to the words containing overlapping symbols may be considered as unmatched, which degrades the accuracy of word sequence matching. We call the missed matchable words missed matches. The amount of missed matches over a subspace is inversely proportional to its discrimination power. A high discrimination power causes a smaller quantity of missed matches. Next, we define the miss ratio of subspaces, which reflects the amount of missed matches caused by subspace symbolization.
Definition 5: Given a set of dominant subspaces $\Omega := \langle S_1, \ldots, S_\gamma \rangle$, the miss ratio of them is defined as:

$$MRatio = \prod_{j=1}^\gamma \frac{1}{SDPower(j)}$$

The miss ratio of dominant subspaces shows their ability of effectively excluding an unmatched pair over them. It is the probability of simultaneously missing an unmatched pair over all of the dominant subspaces. A smaller $MRatio$ value indicates a stronger ability of unmatched pair identification.

Recall that, while dimension-wise quantization cannot compact the video representation [1], clustering high dimensional objects over the whole space tends to produce poor-quality clusters and unstable visual dictionary of large size [4], [31], [35]. This poses a big challenge for the effectiveness and scalability of visual dictionary. Fortunately, since the data distributions over different subspaces vary to a large extent, the dissimilarity between two keyframes is usually determined by several dominant subspaces. Given a set of high-dimensional vectors, the dominant subspaces are constructed by selecting the dominant dimensions of the whole space and dividing them into multiple subspaces based on the data distribution of each. A stable visual dictionary is then constructed based on the keyframe clustering over each dominant subspace.

Space Partition. A naive method for space partition is to divide the whole space into a set of subspaces, each of which consists of equal number of consecutive dimensions. The dominant subspaces are then decided by recording those with maximal discrimination power. We name this approach equal length partition (ELP) in [38]. ELP is simple and easy to operate. However, dividing the whole space into equal-length subspaces may contradict the real data distribution, accordingly, the miss ratio of dominant subspaces is not minimized. Further work is required to partition the dominant dimensions into several subspaces adaptively without looking at dimensions consecutively, so the miss ratio of the dominant subspaces $MRatio$ is minimized.

Theorem 1: The minimal miss ratio can be obtained when $SDPower(1) = \ldots = SDPower(\gamma)$.

Proof: By equation 1, if $MRatio$ is minimized, then $\prod_{j=1}^\gamma SDPower(j)$ is maximized. We will prove, when $SDPower(1) = \ldots = SDPower(\gamma), \prod_{j=1}^\gamma SDPower(j)$ is maximized.

Since $\prod_{j=1}^\gamma SDPower(j) \leq (\frac{\gamma}{\gamma}) SDPower(1)^\gamma$, and $\prod_{j=1}^\gamma SDPower(j) = (\frac{\gamma}{\gamma}) SDPower(1)^\gamma$ can be obtained only when $SDPower(1) = \ldots = SDPower(\gamma)$, we can conclude that the minimal $MRatio$ can be obtained when $SDPower(1) = \ldots = SDPower(\gamma)$.

Based on Theorem 1, we propose the idea of the adaptive space partition. The most important point is to divide the whole space into $\gamma = 2^\chi$ subspaces such that each one has the same discrimination power, where $\chi$ is an integer. This approach aims at obtaining the optimal space partition, which is defined as follows.

Definition 6: Given a set of positive numbers $\Omega := \langle DP_1, \ldots, DP_n \rangle$, where $DP_i$ denotes the discrimination power of a dominant dimension, the Optimal Space Partition problem is to determine whether there is a partition of $\gamma$ subsets such that the sum of the elements of each subset is equal, i.e., determine whether there exists $I \subseteq \{1, \ldots, n\}$ such that $\sum_{i \in I} DP_i = (\sum_{i=1}^n DP_i)/\gamma$.

Given the discrimination power of each dimension, no polynomial time algorithm can efficiently determine the optimal space partition, as demonstrated by the following theorem:

Theorem 2: The problem of Optimal Space Partition is NP-Complete.

Proof: We will prove that optimal space partition is an NP-complete problem via a reduction from the Subset Sum problem. Let $\chi = 1$, then $\gamma = 2$. The optimal space partition is a special case of Subset Sum. Given an instance of Subset Sum, we have to construct an instance of Partition. Let the instance of Subset Sum have items of size $DP_1, \ldots, DP_n$ and a parameter $\kappa$, and let $A = \sum_{i=1}^n DP_i$. Consider the instance of $Partition_{DP_1, \ldots, DP_n, b, c}$, where $b = 2A - \kappa$ and $c = A + \kappa$. Then the total size of the items of the $Partition$ instance is $4A$. We look for the existence of a subset of $DP_1, \ldots, DP_n, b, c$ that sums to $2A$.

Suppose there exists $I \subseteq \{1, \ldots, n\}$ such that $\sum_{i \in I} DP_i = \kappa$. Then $\sum_{i \in I} DP_i + b = \kappa + 2A - \kappa = 2A$.

Reversely, we assume: (1) there exists a subset of $DP_1, \ldots, DP_n, b, c$ that sums to $2A$; and (2) $\sum_{i \in I} DP_i \neq \kappa$ for any $I \subseteq \{1, \ldots, n\}$. Let $I = \{1, \ldots, n\} - \bar{I}$, by (2), for any subset $\bar{I} \subseteq \{b, c\}$, we have $\sum_{i \in \bar{I}} DP_i + \sum_{i \notin \bar{I}} x = 2A$ and $\sum_{i \notin \bar{I}} DP_i + \sum_{i \in \bar{I}} x \neq 2A$. Thus, two assumptions contradict with each other. Accordingly, there exists $I \subseteq \{1, \ldots, n\}$ such that $\sum_{i \in I} DP_i = \kappa$. We can conclude that the partition exists if and only if there exists $I \subseteq \{1, \ldots, n\}$ such that $\sum_{i \in I} DP_i = \kappa$.

When $\chi > 1$, the optimal space partition is a variation of the partition for the case of $\chi = 1$, which continuously finds the optimal one over the subsets of $\Omega$. Thus, the solution for optimal space partition problem can be converted from that of Subset Sum as well. The Optimal Space Partition is NP-Complete.

We propose an adaptive space partition approach named variable length partition (VLP), which heuristically produces dominant subspaces using linear time. The VLP relaxes the requirement of partition. It allows the subspaces to have arbitrary number of inconsecutive dimensions. With this approach, the dominant subspaces are formed adaptively and the discrimination power of each dominant subspace is approximated. Figure 1 outlines how the dominant subspaces are obtained with VLP scheme. Given a set of keyframes $\Sigma$, let $\tau$ be a threshold value, the algorithm consists of three steps. In the first step (line 2), it computes the discrimination power of each dimension $SDPower(j)$ over the given keyframe set $\Sigma$. In the second step (line 3~4), the
dimensions of the whole space are ordered based on their discrimination powers and the median of all the discrimination power values is set as the \( \tau \) value. This is to optimize the partition based on the analysis in Section 3.3. In the third step (line 5~9), we scan the sorted \( DPower(j) \) values sequentially and find the subspace that has the minimal discrimination power for insertion. If the \( DPower(j) \) value of a dimension is not smaller than \( \tau \), this dimension is merged with the selected subspace (line 7~8). Otherwise, we stop the whole process (line 9). The merge operation is conducted over all dimensions until all the ones with discrimination powers no less than \( \tau \) are distributed into the suitable subspaces.

**Dictionary Construction.** Once the whole space is divided into a series of dominant subspaces, we can map the visual feature of each keyframe into the subspaces. Given a keyframe \( f_i < d_{1j}, d_{2k} > \), its projected feature over a subspace is called its sub-descriptor. Suppose that a subspace consists of the \( i^{th} \), \( j^{th} \) and \( k^{th} \) dimensions, then the sub-descriptor of \( f_i \) over this subspace is \( < d_{1j}, d_{2k} > \). As such, each keyframe can be described with a series of dominant sub-descriptors, each of which consists of variable number of its dimension values. The sub-descriptors in different subspaces belong to different groups. The ones in the same group can be clustered using a distance measure (e.g., the Euclidian distance), a cluster’s radius threshold \( \epsilon \), and a hierarchical clustering algorithm, such as the \( k \)-means [30], [18]. In this work, we use the hierarchical 2-means algorithm [30] to generate clusters whose radii are not greater than \( \epsilon \). A cluster \( C \) is denoted as \( < c, a, r, n, t > \), where \( c \) is the cluster id, \( a \) and \( r \) are the cluster center and the radius of the cluster respectively, \( n \) is the number of keyframes in the cluster, and \( t \) is the id of the subspace containing this cluster. A visual dictionary is then used to maintain all clusters of the video database.

Since we process the videos with slight variations, each keyframe can be described as a high dimensional vector, which is its color feature in this work. The idea of visual dictionary can be extended to process the local features as well. We leave this part of research for future investigation on the detection of near duplicate videos with large variations. Using a visual dictionary, a video \( S \) of \( n \) keyframes \( < f_1,...,f_n > \) can be symbolized into a word sequence \( < w_1...w_n > \), where \( w_i \) consists of multiple \( ids \) of the clusters containing \( f_i \) in different dominant subspaces. The mapping from \( f_i \) to \( w_i \) can be simply done by looking up the visual dictionary and checking if the distance between each sub-descriptor to the cluster center over the corresponding subspace is not greater than \( r \). If no cluster of a sub-descriptor is found in the dictionary, a special symbol \( \cdot \) is used to represent it. It is possible for a sub-descriptor to be contained in multiple clusters in a high dimensional space, due to the overlapping between them. In such a case, the cluster having the smallest distance to the sub-descriptor is selected for mapping.

### 3.2 Similarity Measure

Note that in the rest of this paper, we denote a series of symbols transformed from the dominant sub-descriptors of a keyframe as its word. Considering two sequences \( S \) and \( S' \) of \( m \) and \( n \) elements respectively, a straightforward way to measure their similarity is to use the Edit Distance (ED), a commonly used string matching that counts the minimal number of updates (including insertion, deletion and substitution operations) to transform one string into another [28]. Let \( S_{m-1} \) be the subsequence of the first \((m-1)\) elements of \( S \), and \( S[i] \) be the \( i^{th} \) element in \( S \). The edit distance between \( S \) and \( S' \), \( ED(S,S') \), is defined as follows:

\[
ED(S,S') = \begin{cases} 
\max(m,n) & m = 0 \text{ or } n = 0 \\
\min\{ED(S_{m-1},S'_{n-1}) + p, \} & \text{otherwise} \\
ED(S_{m-1},S'_{n}) + 1 & \text{otherwise}
\end{cases}
\]

In video matching, \( p \) is decided by the inter-keyframe distance between \( S[m-1] \) and \( S'[n-1] \). If the distance is within \( \epsilon \), \( p = 0 \); otherwise, \( p = 1 \). For string matching, \( p = 0 \) if \( S[m-1] = S'[n-1] \); and \( p = 1 \) otherwise. Unlike a string sequence where the difference between two words is simply a Boolean value, in the matching of symbolized video sequences, different words in our video symbol sequences may have some similarity due to the fact that their corresponding clusters may have certain overlap in the space. Meanwhile, different from the character comparison in text retrieval where the same characters are completely matched, the same symbols to different sub-descriptors can not reflect the similarity between the keyframes containing them. Next, we will address two key problems in the measure of symbolized word sequences, i.e., deciding the similarity between two
words and compensating the information loss caused by video symbolization.

**Probability Measure** Since certain overlapping may exist among the neighboring clusters, neighboring clusters may have certain similarity. Given two clusters $C_i$ and $C_i'$, we define the dissimilarity between two overlapping clusters as follows:

$$p_i = \frac{|C_i - C_i'| \ast |C_i' - C_i|}{|C_i| \ast |C_i'|}$$

where $C_i$ and $C_i'$ are the clusters in a certain sub-space of visual features, $|C_i|$ represents the number of video keyframes in $C_i$. Note that $0 \leq p_i \leq 1$ is calculated based on the geometry probability that a keyframe in $C_i$ and a keyframe in $C_i'$ are dissimilar. A greater $p_i$ value for two symbols indicates that their keyframes are less likely to be similar in the corresponding sub-space. To ensure that the sub-descriptors of keyframes in the same cluster are highly similar to each other, the space for each cluster is usually very small. One can assume uniform data distribution in a small cluster. Therefore, $|C_i - C_i'|$ can be rapidly estimated as $|C_i - C_i'| \approx \frac{V(C_i')|C_i|}{V(C_i)}$, where $V(C_i)$ is the volume of cluster $C_i$. $V(C_i)$ and $V(C_i' - C_i)$ in a high-dimensional space can be computed by the methods proposed in [30]. Based on the $p_i$ values of two sub-descriptors, we propose Probability Word Edit distance (PWE), which extends edit distance by redefining the value of $p$ as $\max \{p_i\}$. This is shown in Proposition 1.

**Proposition 1:** PWE does not satisfy the triangular inequality

**Proof:** By counter example. Consider three video sequences $v_1(f_{11}), v_2(f_{21})$, and $v_3(f_{31})$. Here, only one keyframe is contained in each. Let $d(f_{11}, f_{j})$ be the Euclidean distance between $f_{11}$ and $f_j$. $d(f_{11}, f_{21}) = \frac{2}{3}$, $d(f_{11}, f_{21}) = \frac{4}{7}$, $d(f_{11}, f_{31}) = \frac{4}{7}$. By the definition of probability value, we get $p(f_{11}, f_{21}) = p(f_{11}, f_{31}) = 0, p(f_{11}, f_{31}) = 1$.

Assume that PWE does not satisfy the triangular inequality, then we have $p(f_{11}, f_{21}) + p(f_{31}, f_{21}) \geq p(f_{11}, f_{31})$. So, $p(f_{11}, f_{31}) = 0$. This contradicts with the given value of $d(f_{11}, f_{31})$. Thus, we conclude that the assumption is wrong and PWE measure does not satisfy the triangular inequality.

**Information Compensation.** Using sequence context to compensate the information loss from video symbolization is a promising way for improving the quality of sequence matching. Practically, due to the various forms of quality degradation, visual features may vary among the similar video clips, which challenges the accuracy performance of video detection. Besides this, considerable information loss from video symbolization has further emphasized the necessity of effective information compensation. With this consideration, we redefine the video similarity measure by embedding the sequence context difference between two videos.

Given a video sequence $S$, let $< f_1, ... f_n >$ be the sequence consisting of its keyframes, its sequence context is formed by the similarity between the neighboring keyframes. We represent the sequence context of $S$ as a context vector, $S_c = < x_1, ... x_{n-1} >$, where $x_i$ denotes the distance between the $i^{th}$ and $(i+1)^{th}$ keyframe. Given two video sequences $S$ and $Q$, suppose that $S_c$ and $Q_c$ are the context vectors of them respectively, we define the context difference between them as $d(S_c, Q_c)$, which is Euclidean distance or other metrics. The dissimilarity between $S$ and $Q$ is redefined as:

$$CVE(S, Q) = \varphi_1 PWE(S, Q) + \varphi_2 d(S_c, Q_c) \quad (3)$$

where $\varphi_1$ and $\varphi_2$ are the weights of the visual feature difference and context difference in the distance function. CVE considers the difference of visual content between two video sequences and that of their sequence relationship, which compensates the information loss from video symbolization, thus, symbolized keyframes can be effectively utilized to reduce the complexity of the measure significantly while the high effectiveness of matching is maintained.

### 3.3 Analysis

Based on the discrimination power of the dominant subspaces $SDPower$ in definition 4, we have another metric, $PD Power$, which shows the amount of preserved information after the dominant subspace selection.

$$PD Power = \frac{SDPower}{\sum_{j=1}^{D} Power(j)}$$

We study the information loss of dominant subspace selection using 18-hour videos. We test the $PD Power$ values to different compression ratios, denoted as $CRatio$, which is computed by $CRatio = \frac{\sum_{j=1}^{D} d_j}{D}$. The statistics results are shown in Table 1. Clearly, when the $CRatio$ is reduced from 62.5% to 50%, there is only a slight drop, from 98.2% to 95.5%, on the $PD Power$. With further compression of data, we see a more dramatic decrease on the $PD Power$. The optimal $\tau$ can be obtained by the median of all the $PD Power$ values, which indicates a good balance of the information preserved and the video compression ratio.

Then, we analyze the optimal $\gamma$ of space partition under VLP scheme using the overall discrimination power of dominant subspaces. The same dataset for the analysis of optimal $\tau$ (18-hour) is used. Table 2 reports the overall discrimination power of dominant subspaces, $SD Power$, for different $\gamma$ values, from 2 to 16. The statistics show that the highest $SD Power$ value is achieved at the point of $\gamma = 4$. Thus, the optimal $\gamma$ value under VLP partition for this real data set is 4.

We analyze the stability of a visual dictionary with subspace-based clustering. Under VLP scheme, the maximal dictionary size is estimated as: $\sum_{k=1}^{\gamma} \left(\frac{1}{2}\right)^{d_k}$, where $d_k$ is the dimensionality of the $k^{th}$ dominant subspace.
TABLE 1

<table>
<thead>
<tr>
<th>CRatio</th>
<th>62.5%</th>
<th>50%</th>
<th>37.5%</th>
<th>25%</th>
<th>12.5%</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDPower</td>
<td>95.2%</td>
<td>95.5%</td>
<td>88.9%</td>
<td>79.3%</td>
<td>62.2%</td>
</tr>
</tbody>
</table>

TABLE 2

SDPower for different γ under VLP

<table>
<thead>
<tr>
<th>γ</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDPower</td>
<td>5264.7</td>
<td>5842.7</td>
<td>5797</td>
<td>5764.9</td>
</tr>
</tbody>
</table>

TABLE 3

SDPower for different d under ELP

<table>
<thead>
<tr>
<th>d</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDPower</td>
<td>4344.5</td>
<td>4458.8</td>
<td>4494.9</td>
<td>4441.1</td>
<td>3148.5</td>
</tr>
</tbody>
</table>

TABLE 4

Clustering results for different dominant space selection

<table>
<thead>
<tr>
<th>OCNum</th>
<th>18.8</th>
<th>216</th>
<th>216</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSize</td>
<td>1812</td>
<td>11851</td>
<td>11847</td>
</tr>
</tbody>
</table>

TABLE 5

Change on dominant dimension selection and space partition for different video data size

<table>
<thead>
<tr>
<th>DataSize(hours)</th>
<th>50</th>
<th>100</th>
<th>150</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCRatio</td>
<td>18.75%</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>PCRatio</td>
<td>43.75%</td>
<td>18.75%</td>
<td>6.25%</td>
<td>0</td>
</tr>
</tbody>
</table>

We analyze the quality of visual dictionary based on the number of overlapped clusters (OCNum) and the size of dictionary (DSize) using three different subspace selection approaches: (1) VLP, (2) PCA-based subspace selection and (3) Non-segment subspace selection. Here, PCA-based method is performed by first using PCA and then choosing the principal components of the transformed data for clustering. Non-segment based method simply picks all the dimensions with maximal discrimination power for the next step clustering. The statistic results over 18-hour videos for them are shown in Table 4. Clearly, with VLP, a compact visual dictionary and higher quality clustering results can be obtained.

We analyze the stability of optimal space partition using two indicators, the change on dominant dimension selection DCRatio and that on partition PCRatio. Given two sets A and B of the dominant dimensions to different dataset, the DC Ratio is defined as the Jaccard distance between them, i.e., $1 - \frac{|A \cap B|}{|A \cup B|}$. Given two partitions to two sets, the PC Ratio on them is defined as the ratio of operation number for transforming one partition to the other to the total number of elements in each. We test the values of these two indicators by increasing the dataset from 25 hours to 200 hours with its previous dataset as the base, i.e. the base of the 50-hour dataset is the 25-hour set, then of the 100-hour one is 50-hour video set etc.. The changes on DC Ratio and PC Ratio are reported in Table 5. Clearly, with the increasing of the dataset, the changes on both indicators reduce and fall to 0 in the end. Thus, our optimal subspace partition is stable, and we do not need to do optimal partitioning of the feature space across all videos of an increasing data collection.

Finally, we analyze the optimal $T$ in PWE measure from the perspective of statistics.

**Observation 1:** In statistics, measurement error is a statistic distribution, called Gaussian distribution, with probability density function

$$p(x) = \frac{1}{\alpha \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

where $p(x)$ is a continuous variable. We use Maximum Likelihood Estimation (MLE) to estimate the optimal threshold value. To obtain the MLE estimates for the mean, $\bar{p}$, and standard deviation, $\sigma_p$, for the normal distribution, we start with the PDF of the normal distribution listed as follows.

$$f(p) = \frac{1}{\alpha \sqrt{2\pi}} e^{-\frac{(p-\bar{p})^2}{2\sigma^2}}$$

If $p_1, ..., p_n$ are known, then the likelihood function is:

$$L(p_1, p_2, ..., p_n | \bar{p}, \sigma_p) = \prod_{i=1}^{n} \frac{1}{\alpha \sqrt{2\pi}} e^{-\frac{(x_i-\bar{p})^2}{2\sigma^2 p}}$$

Let $A = \ln L$, then we take the partial derivatives of $A$ with respect to each of its parameters and set each

partial derivative equal to zero, so the optimal $\bar{p}$ value that minimizes the measurement error is obtained.

\[
\frac{\partial (\Lambda)}{\partial p} = \frac{1}{\sigma_p^2} \sum_{i=1}^{n} (p_i - \bar{p}) = 0
\]  

\[
\frac{\partial (\Lambda)}{\partial \sigma_p} = -\frac{n}{\sigma_p} + \frac{1}{\sigma_p^2} \sum_{i=1}^{n} (p_i - \bar{p}) = 0
\]  

Solving the equations in 7 and 8, we have $\bar{p} = \frac{1}{n} \sum_{i=1}^{n} p_i$. We randomly select 1000 clusters and compute their probability values between each other. The mean value of these probabilities is computed, and $\bar{p} = 0.5$ is obtained. So, the highest accuracy can be obtained with the probability threshold value of $T=0.5$.

4 VIDEO INDEXING

4.1 Index Structure

We propose an effective index structure on video symbol sequences. Our index structure includes two tiers: (1) multiple B+ -trees that are used to index symbols to different dominant subspaces; (2) an inverted file that is used to locate the videos containing the symbols.

At the first layer, each B+ -tree is used to index all symbols(clusters) generated from all the sub-descriptors of video data in a single dominant subspace. The B+ -tree is selected because of its simplicity and efficiency. Given a query symbol, its similar symbols in a certain subspace can be quickly found by an efficient range search in the corresponding B+ -tree. To build the index, we first associate an indexing key with each symbol, which is generated using the optimal one-dimensional transformation method [30]. Given a set of d-dimensional points and the transformation function $Dist$, an optimal reference point $o_{ref}$ is first determined by finding the point maximizing the variance of the inter-distances of points in transformed one dimensional space, where the inter-distance of two points $o_i$ and $o_j$ in the transformed space is defined as $|Dist(o_i, o_{ref}) - Dist(o_j, o_{ref})|$. Then, the indexing key for symbol $c_i$, denoted as $key(c_i)$, is computed as: $key(c_i) = (a_i, o_{ref})$, where $a_i$ is the cluster center for $c_i$ and $o_{ref}$ is the selected optimal reference point which lies on the first Principal Component identified by Principal Component Analysis and out of the data range determined by all cluster centers in a certain subspace [30]. After the indexing keys for all symbols in a subspace are generated, a B+ -tree is then constructed on all these keys. Multiple B+ -trees are constructed over the symbols of all dominant subspaces. At the leaf node level of each B+ -tree, the cluster information about the symbol, i.e., $C = (c, o, r, n, t)$, is also maintained.

At the second layer, an inverted file is used to identify the videos that contain each symbol. A list of videos for each symbol is maintained. Besides the symbol, it is also necessary to remember the position of its appearance in the sequence so that the temporal ordering can be considered in CVE computation. Since several continuous keyframes may have the same symbol in a dominant subspace, it is necessary to remember the number of consecutive occurrences of a symbol in a sequence. We use a triplet $(V_{id}, pos, len)$ to represent the appearance of a symbol in a sequence, where $V_{id}$ is the video id, $pos$ is the starting position of a symbol in a video symbol sequence, and $len$ is the number of its consecutive occurrences in the video sequence $V_{id}$. Clearly, a triplet $(V_{id}, pos, len)$ can be regarded as a video segment. Representing the high dimensional segments with a triplet (i.e., three values) requires a much smaller storage space.

![Fig. 2. The two-layer index structure](image)

Figure 2 shows our two-layer index structure. In the B+ -trees, the cluster information for each symbol, i.e., $C = (c, o, r, n, t)$, is also maintained in leaf nodes. In the inverted file, each symbol points to a list of triplets $(V_{id}, pos, len)$ for locating video symbol segments that contain it. For performance improvement, the inverted file is stored in a compressed manner, i.e., similar triplets are maintained in the same or consecutive disk pages. The space cost of the index structure depends on the size of visual dictionary and that of the video database. Suppose that the dictionary size is $C_d$ and the database size is $C_D$, let $p$ be the compression ratio of video data to the decomposed triplets, then the space required to store the tree would be $O(C_{d} + C_{D} * p)$.

4.2 Index Maintenance

This section discusses how to deal with the dynamic issues. We are mainly concerned with how video sequences are inserted into the database, and also how video insertions cause the change of clusters. Given a new video $S$, its keyframes are first projected into different subspaces, in which the obtained sub-descriptors are processed and inserted into the index structure.

Figure 3 presents the dynamic insertion algorithm, which mainly consists of two steps, searching B+ -trees and updating the two-tier index structure. It first projects the video features into a set of sub-descriptors, and
decides a sub-descriptor’s destination cluster by searching the corresponding B+-tree and choosing the cluster with the smallest distance (line 1–8). To optimize this process, the sub-descriptors of S in each subspace can be first grouped into a few number of clusters, each of which contains similar sub-descriptors and is processed as a whole (line 3–6). If the distance between the sub-descriptor and its destination cluster is not greater than \( \frac{s}{2} (r \leq \frac{s}{2}) \), the sub-descriptor is symbolized directly (line 9–10). Otherwise, the destination cluster is split, and a new cluster is generated (line 11–13). Accordingly, some video segments in the inverted index are moved from the destination cluster to the new cluster, and the centers and radii of both clusters are recomputed. Finally, a set of triplets is produced based on the symbolized videos, and inserted into the inverted list (line 14–15).

5 QUERY PROCESSING

Given a query \( Q = < f_1...f_n > \), the video identification is performed by three main steps: query mapping, sequence filtering and sequence refining. We will explore each step as following.

5.1 Query Mapping

Given a query video, query mapping aims at two tasks: First the sub-descriptors of each keyframe are mapped into symbols by looking up the visual dictionary. Then, the video segments containing these symbols are retrieved, so the potentially similar video candidates are decided. A keyframe is symbolized by searching the two-layer index structure. In the first layer, i.e., B+-trees, a range search is performed for each dominant sub-descriptor of a query keyframe to find its similar symbols according to the reference sub-descriptor. As a sub-descriptor should have a distance up to \( \frac{1}{2} \epsilon \) from its symbol center, similar symbols can only be identified in the range of \( (x - \frac{1}{2} \epsilon, x + \frac{1}{2} \epsilon) \), where \( x \) is the distance between the reference sub-descriptor and the query one. Following the second layer, i.e., the inverted file, the corresponding triplets \( < V_{id}, pos, len > \) are retrieved and used to find the potentially similar videos. However, for a query video, mapping the sub-descriptors of each keyframe to their symbols may incur heavy cost due to the large number of distance calculations. Next, an optimization based on query summarization is proposed to reduce the computation cost in query mapping.

**Optimization by Query Summarization** To map the keyframes in a query videos to their words, a naive method is to map each of them one by one. As a query sequence may consist of a number of keyframes, given a query containing \( m \) keyframes, this method requires index search over each subspace for \( m \) times. Suppose that the average size of candidate sets in the mapping is \( s \), the computation cost of mapping a video is estimated as \( m * s * \gamma \). Thus the naive approach is impractical in video retrieval. In this part, we present a query summarization method **local clustering**, which greatly improves the search efficiency.

Figure 4 shows the algorithm \( QueryMapping \) based on local clustering. It first projects the visual features of the query into the dominant subspaces (line 1). The sub-descriptors in each dominant subspace are then summarized into a number of small clusters by local clustering (line 2–3). Finally, each cluster is mapped into a set of symbols (line 4–7), by which the triplets containing them are retrieved (line 8–9).

Local clustering is the core of query mapping. Practically, certain similar keyframes usually exist in the same video sequence and potentially share some common query spaces, thus it is possible to cluster the similar sub-descriptors for batch processing. To do so, a simple recursive 2-means clustering algorithm is performed over each query subspace to group similar sub-descriptors together. We choose \( k \)-means algorithm because of its easy implementation and its time complexity of \( O(m) \), where \( m \) is the number of keyframes. Then, a query video se-

---

**Procedure DynamicInsertion.**

\[
\text{input: } S - \text{ a video sequence} \\
1. \{S_D\} \leftarrow \text{ProjectToSubspaces}(S) \\
2. \text{for sub-descriptors in each subspace } S_D \in \{S_D\} \\
3. \{C^{m}_{i}\} \leftarrow \text{Summarization}(S_D) \\
4. \text{for each cluster } C^{m}_{i} \in \{C^{m}_{i}\} \\
5. d_{ref} \leftarrow \text{GetDist}(C^{m}_{i}, ref) \\
6. \{C_{i}\} \leftarrow \text{LocateLeaf}(B^{+}\text{-tree}, d_{ref} - \frac{s}{2}, d_{ref} + \frac{s}{2}) \\
7. \text{for each } f \in C^{m}_{i} \\
8. \{MinD, C_{i}\} \leftarrow \text{MinDistAndSymbol}(f, \{C_{i}\}) \\
9. \text{if}(MinD < \frac{s}{2}) \\
10. s_{f} \leftarrow id(C_{i}) / \ast \text{symbolize } f/ \\
11. \text{else } C_{n} \leftarrow \text{GenerateNewCluster}(C_{i}, f) \\
12. \text{InsertIntoB+trees}(C_{n}) \\
13. s_{f} \leftarrow id(C_{n}) \\
14. \{T_{i}\} \leftarrow \text{ProduceTriplets}\{(s_{f})\} \\
15. \text{InsertIntoInvertedList}(\{T_{i}\})
\]

Fig. 3. The dynamic sequence insertion algorithm.

**Procedure QueryMapping.**

\[
\text{input: } Q - \text{a query sequence} \\
\text{output: } \{T_{i}\} - \text{a set of triplets} \\
1. \{Q_D\} \leftarrow \text{ProjectToSubspaces}(Q) \\
2. \text{for sub-descriptors in each dominant subspace } Q_D \in \{Q_D\} \\
3. \{C^{m}_{i}\} \leftarrow \text{LocalClustering}(Q_D) \\
4. \text{for each cluster } C^{m}_{i} \in \{C^{m}_{i}\} \\
5. d_{ref} \leftarrow \text{GetDist}(C^{m}_{i}, ref) \\
6. \{C_{i}\} \leftarrow \text{LocateLeaf}(B^{+}\text{-tree}, d_{ref} - \frac{s}{2}, d_{ref} + \frac{s}{2}) \\
7. \{c_{i}\} \leftarrow \text{GetNearestSymbols}(C^{m}_{i}, \{C_{i}\}) \\
8. \text{for each } c_{i} \in \{c_{i}\} \\
9. \{T_{i}\} \leftarrow \text{MapToVTriplets}(c_{i}, \text{inverted list}) \\
10. \text{Return } \{T_{i}\}
\]

Fig. 4. The query mapping algorithm.
sequence is grouped into a small number of clusters which contain similar sub-descriptors, and the query mapping can be performed based on the batch processing of these clusters, thus reducing the comparison with the common part of the candidate sets. This method reduces the mapping cost compared with the naive method greatly by paying little query summarization cost.

5.2 Sequence Filtering

Given a set of triplets retrieved by query mapping, a set of potentially similar video sequences can be constructed. By sequence filtering, the number of candidates is further reduced. This filtering is performed by the comparison between each candidate and the query with the position information of each triplet. Since we used Euclidean distance to measure the sequence context difference, the complexity of which is linear. Meanwhile, for two sequences of lengths \( m \) and \( n \), their PWE computation cost is \( O(m \times n) \), which is much higher than Euclidean distance calculation. Thus, the cost of CVE is mainly decided by the number of PWE calculations. It is preferable to filter the number of candidates as much as possible before applying the PWE computation. For this, we proposed two filtering strategies: *maximal gap number filtering* (MaxGap) and *dual filtering* (DUAL) based on the number of dissimilar keyframes.

**MaxGap.** The MaxGap filtering strategy is based on the following consideration: after sequence reconstruction, a candidate sequence may contain many gaps, since the dissimilar keyframe segments in a video database are filtered out and replaced with gaps. Each gap is dissimilar to any keyframe in the query. Given two sequences, if there are \( l \) gaps in either sequence, at least \( l \) transitions are required to transform one to the other. Thus, the number of gaps is a lower bound of PWE.

**DUAL.** The DUAL filtering strategy is extended from the techniques for string matching. In string matching, each string can be transformed into its frequency vector that records the frequency occurrences of its characters. The *frequency distance* is defined as the minimum number of steps to transform one string to another, while not considering the position of each character in them. Although frequency distance is effective in reducing the comparisons by edit distance in string matching, word sequences transformed from videos can not be processed in the same way as strings because of the possible overlap between different clusters. To adapt the frequency distance to the new application, we propose the DUAL filtering, which consists of filtering by a lower bound of PWE and filtering by its upper bound.

**Neighboring Cluster** is an important conception in DUAL strategy. Let \( C_i \) and \( C_j \) be two clusters (i.e., symbols) in the high dimensional subspace, if the spaces of \( C_i \) and \( C_j \) are overlapped with each other, they are Neighboring Clusters. Based on this definition, we propose two important distances, Intersection Frequency Distance (IFD) and Simple Manhattan Distance (SMD), which are lower bound and upper bound of PWE respectively. Let \( S_i \) and \( S_j \) be two video sequences, IFD(\( S_i, S_j \)) is defined as the frequency distance of two sequences when considering the words containing neighboring symbols in all dominant subspaces as matched. Let \( S_i \) and \( S_j \) be two video sequences of the same length, SMD(\( S_i, S_j \)) is the Manhattan distance of them when considering each sequence as a string and the difference between two words as a Boolean value. In SMD, the comparison can only happen between two words at the same positions, and the difference between two compared words(at the same position) is decided by whether they are same or not. For \( S_i \) and \( S_j \) of variable lengths, SMD(\( S_i, S_j \)) is the minimal one among the SMDs between the shorter sequence and the subsequences of a longer one. Together with our PWE, we have the following theorem.

**Theorem 3:** Let \( S_i \) and \( S_j \) be two video sequences, we have IFD(\( S_i, S_j \)) \leq \text{PWE}(\( S_i, S_j \)) \leq \text{SMD}(\( S_i, S_j \));

Proof: IFD is based on the keyframe comparison between two videos by describing them as keyframe sets. In such a case, any change in a keyframe set will correspond to an edit operation in the sequence when considering temporal ordering. Thus, the number of steps in transforming a keyframe set is the lower bound of edit operations. In SMD, keyframes contained in the same cluster in each dominant subspace are similar, but those in two neighboring clusters are considered as dissimilar. For two keyframes, if they are regarded as similar in PWE, they are similar in SMD. Meanwhile, the temporal order is kept strictly, thus the number of edit operations in SMD is bigger than that in PWE. □

**Filtering Cost Analysis.** We analyze the cost of MaxGap, IFD and SMD, to find the best filtering strategy. Given a query video \( Q \) of \( m \) and a video data \( S \) of \( n \), the cost of SMD is \( \min(m, n) \). Suppose that the number of segments in \( Q \) be \( l_Q \), that of \( S \) retrieved from inverted list \( l_s \), then the costs of MaxGap is \( l_s \). The cost of IFD is \( k l_Q \), where \( k \) is the average number of the matched words in \( S \) to each word in \( Q \). Clearly, while MaxGap filtering takes much lower filtering cost than Dual filtering, the latter one provides a tighter lower bound of PWE. Considering the superiority of both, we first use the MaxGap method to prune the candidates, followed by the further DUAL filtering. The whole filtering is shown as Figure 5.
5.3 Sequence Refining

Sequence refinement is performed to refine and rank the candidates returned from the last step by CVE. For a set of triplets generated from query mapping and sequence filtering, we recompose them into sequences based on the location and occurrence information of each similar symbol, i.e., \(<V_{id}, pos, len>\). However, there may be cases where some symbols in a similar sequence are not similar to any symbol in a query at all, thus cannot be identified. By sequence reconstruction, we represent each of these unidentified symbols as a symbol gap to fill the vacant spaces in the reconstructed symbol sequences, and thus keep the similarity of original sequences and facilitate future sequence refinement.

Suppose a query video \(Q < f_1, f_2, f_3 >\) is mapped to two clusters in a subspace, cluster 2 and 3, and also after query mapping, two triplets from this subspace are generated, \(<S_1, 2, 3>\) covered by cluster 2, and \(<S_2, 1, 2>\) covered by cluster 3, which means \(S_1\) and \(S_2\) are two videos possibly similar to \(Q\). As there is no knowledge about other triplets in \(S_1\) and \(S_2\) that have the same role as the triplets of gaps, from the view of this subspace, \(S_1\) and \(S_2\) can only be represented as \(\{-222\}\) and \(\{33-\}\) respectively. The final results are obtained by three operations, first combining the representations of these two sequences in the specific subspaces orderly, then comparing the reconstructed word sequences of candidates and that of the query based on PWE, and finally obtaining the CVE between them.

6 Performance Study

In this section, we evaluate the effectiveness and efficiency of proposed approaches by conducting a series of experiments and report the experimental results.

6.1 Experimental Set-up

We conduct the experiments on a large collection of real videos consisting of commercials, news and weather broadcasting reports captured from TV stations and recorded using the Virtual Dub at the PAL frame rate of 25fps [30]. The similar videos share similar content, and all the video clips can be categorized based on their color histograms. Each video keyframe is compressed using PLCVideo Mjpegs with the resolution of 192×144 pixels. A video keyframe is represented as a 32-dimensional vector that is a color histogram in the RGB color space. We generated two video datasets, which respectively contain: (1) 18 hours 60s-clips for the effectiveness evaluation; (2) 240 hours 15s to 60s clips for the efficiency evaluation. Note that the dataset used for effectiveness evaluation is not an 18-hour footage recorded by a video camera continuously, but thousands of video clips extracted from TV broadcasting of Australian TV channel 7 and 9. They were obtained over months and initially used for commercial monitoring by Nielsen Media Research, Australia.

<table>
<thead>
<tr>
<th>Para</th>
<th>Description</th>
<th>Default value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\epsilon)</td>
<td>Inter-keyframe similarity threshold</td>
<td>0.15</td>
</tr>
<tr>
<td>(T)</td>
<td>Probability threshold</td>
<td>0.5</td>
</tr>
<tr>
<td>(\gamma)</td>
<td>Number of dominant subspaces</td>
<td>4</td>
</tr>
<tr>
<td>(\varphi_1)</td>
<td>Weight of visual feature difference</td>
<td>1</td>
</tr>
<tr>
<td>(\varphi_2)</td>
<td>Weight of context difference</td>
<td>2</td>
</tr>
<tr>
<td>(T_{ss})</td>
<td>Inter-sequence similarity threshold</td>
<td>0.3</td>
</tr>
<tr>
<td>(N)</td>
<td>Size of dataset</td>
<td>18-hours, 240-hours</td>
</tr>
<tr>
<td>(L)</td>
<td>Length of Sequences</td>
<td>60s</td>
</tr>
</tbody>
</table>

Six clips are selected from the collection of 18 hours 60s-clips as queries. The selected query clips are dissimilar to each other. For each query, at least three different video clips with different variations in the video collection are similar to it. The maximum number of relevant video clips for a query is 20. For each query, we perform kNN search, and 50 video clips are returned. The retrieved results are ranked by the CVE distance between each and the query clip. We define the similar videos as those containing duplicated content but with two types of differences: (a) Formatting differences including encoding format, frame rate, bit rates; (b) Content differences including slightly photometric variations (color change, lighting change, contrast change), slightly editing (logo insertion, adding borders around frames, cropping, subimage replacement), content modification (adding unrelated frames with different content), versions (same content in different lengths), content replacement (replace a block of frames with the new ones). We have built the ground truth for the 18 hours 60s-clip dataset. For each query clip, its ground truth is constructed based on human relevance judgements. Three postgraduate students participated in the user study with the direction of our video similarity definition above. The ground truth is labeled based on intersubject agreement on the relevance judgements. The major parameters and their default values used in the experiments are listed in Table 6.

The first five parameters, including \(\epsilon, T, \gamma, \varphi_1\) and \(\varphi_2\), decide the effectiveness of the SUDS system. We set a default value of 0.15 to \(\epsilon\) based on our experimental results and the statement in [34]. It has been justified two keyframes that have Euclidean distance less than 0.15 can be identified as similar with high confidence level. The setting of this distance threshold meets our requirement of clustering, i.e., keeping a set of similar keyframes in the same cluster. For parameter \(T\), since our analysis in section 3.3 is based on randomly selected clusters, the default value of it is effective for all video data collections. The default values of \(\gamma, \varphi_1\) and \(\varphi_2\) are tested on a large real dataset extracted from TV broadcasting over long time, which contains enough diverse video data with a stable data distribution. Accordingly, stable default values of these parameters can be obtained. Thus, it is not necessary to change the default values of these parameters over different datasets.
6.2 Evaluation Methodology

We have conducted extensive experiments to evaluate the proposed method, SUDS, from two aspects: (1) effectiveness; (2) efficiency. The detailed evaluation methodology is described as below.

**Effectiveness of CVE.** The CVE is compared with two well-known video matching approaches, ERP [6] and Signature Alignment (SA) [42], which have been introduced in the related work. For the CVE, we perform clustering over each subspace using hierarchical 2-Means [7] based on the Euclidean distance.

We adopt standard evaluation method in TRECVID. The evaluation is based on the average precision and the Precision – Recall curves. Precision is the proportion of clips retrieved that are relevant. Recall is the proportion of relevant clips that are retrieved. The precision is calculated after each relevant clip is retrieved. If no relevant clip is retrieved in the returned results of a query, the precision of this query is 0.0. A precision-recall curve is then produced by measuring precisions at 11 evenly spaced recall points (0, ..., 1.0). All precision values are averaged to get a single number for the performance of a query. The values averaged over all queries lead to the average precision of a search system.

**Efficiency of SUDS.** We also measure the search efficiency in terms of CPU cost during the video retrieval. We do not consider the efficiency of index construction in this work because the indexing method is based on two well studied existing structures [30], [41]. Since our index is not big and can be kept in memory in query processing, the IO cost is not in our consideration. The CPU cost is measured by the number of distance calculations, which is more objective. The 240-hour clip dataset is used in this experiment. We use 30 queries, which are sequences of frames randomly chosen from the 240-hour clip dataset. The final results are obtained by averaging the results of all these queries.

6.3 Experimental Results

6.3.1 Effectiveness of SUDS

With the SUDS approach, four primary parameters are involved in the CVE: the clustering threshold, the coefficients of visual feature difference \( \varphi_1 \) and sequence context difference \( \varphi_2 \), and the probability threshold \( T \) which decides how the inter-keyframe similarities are reflected in the CVE measure. In this part, we study the effectiveness of SUDS from the following aspects: (1) the effect of parameters, \( \epsilon, \varphi_1, \varphi_2, \) and \( T \); (2) the effect of sequence length \( L \); and (3) comparing the SUDS with the existing competitive approaches.

**Effect of \( \epsilon \).** In this experiment, we test the effectiveness of CVE(1, 1), CVE(1, 2) and ED in the original space by varying the similarity threshold \( \epsilon \), where CVE(1, 1) denotes the CVE measure with \( \varphi_1 = \varphi_2 = 1 \), CVE(1, 2) is the CVE measure with \( \varphi_1 = 1 \) and \( \varphi_2 = 2 \), to obtain the best inter-keyframe similarity threshold. Figure 6(a) shows the effect of \( \epsilon \) on the average precision of search with different values of \( \varphi_1 \) and \( \varphi_2 \) for the 18-hour video dataset. The minimal \( \epsilon \) value is fixed to 0.1, since the keyframes that have a distance smaller than this value are very similar and not discriminative. As shown in the figure, for all measures with different values of \( \varphi_1 \) and \( \varphi_2 \), the average precision of search first increases and then decreases with \( \epsilon \) increasing. The search average precision reaches to a peak value when \( \epsilon \) is changed to 0.15. This is mainly because the average precision of search is much affected by the quality of clusters. For very small \( \epsilon \), the number of false negatives increases, thus deteriorating the average precision of the search. While for very big \( \epsilon \) values, the average precision of the search is reduced by the increasing number of false positive discriminations.

**Effect of \( \varphi_1 \) and \( \varphi_2 \).** We test the effect of coefficients of visual feature difference and sequence context difference of CVE, \( \varphi_1 \) and \( \varphi_2 \), by fixing \( \epsilon \) to its default value, the value of \( \varphi_1 \) to 1, and changing \( \varphi_2 \) from 0 to 3. Note: CVE(1,0) is the ED. Figure 6(b) shows the average precision of the search with different values of \( \varphi_2 \). We notice that, since the context information captures more sequence information of clips, the average precision of the search increases greatly when \( \varphi_2 \) is changed from 0 to 1. Then, the average precision of search is optimized slightly from \( \varphi_2 \) of 1 to 2, and reaches to the best value. A slight drop in the effectiveness of search can be seen after \( \varphi_2 \) of 2. Thus, the appropriate ratio of \( \varphi_1 \) and \( \varphi_2 \) is 1/2 in our measure.

**Effect of \( T \).** We test the effect of probability threshold \( T \) on the average precision of search in the symbolized space, by fixing \( \epsilon \) to 0.15 and varying \( T \) from 0.1 to 0.9. Figure 6(c) shows the results. Obviously, for each of PWE, CVE(1,1) and CVE(1,2), when \( T \) is 0.5, the highest precision of search is achieved. This result has verified our prediction of \( T \) value.

**Effect of \( L \).** We test the effect of the sequence length on the PWE and CVE measures over the symbolized space by reducing \( L \) from 20% to 100% of the original clips. Figure 6(d) shows the changing trend of average precision for each measure. Obviously, with the increasing of \( L \) value, CVE with different \( \varphi_1 \) and \( \varphi_2 \) values obtains an increasing average precision. This is because, with the increasing of \( L \) value, the sequence context information can be captured more effectively. While for PWE measure, since only visual feature information is used, the average precision keep steady with the change of \( L \). Thus, for symbolized video sequences, CVE can effectively compensate information loss.

**Comparison of CVE and Existing Measures.** We compare different approaches to demonstrate: (1) VLP is of superiority over the ELP which is proposed in our ICDE short paper [38], while subspace based symbolization is more effective than the whole space based symbolization; (2)
CVE with VLP scheme is more effective than the existing competitors, including ERP [6] and Signature Alignment (SA) [42]. To be fair, the number of dominant subspaces for both ELP and VLP is fixed to the default value of 4. For each of these approaches, the precision at each recall level is reported in Figure 7(a).

From Figure 7(a), we notice that, among the three different symbolization approaches including CVE with VLP (CVE-VLP), CVE with ELP (CVE-ELP) and the whole space symbolization (CVE-G), CVE-VLP achieves the best accuracy performance at each recall level. This is mainly caused by the higher clustering quality in subspaces. In the original whole space, a single keyframe falls into the intersection part of more symbol spaces, which reduces the discrimination power of the symbols. Meanwhile, with VLP strategy, more discriminative subspaces can be preserved.

On the other hand, comparing with the existing measures, CVE-VLP has the best precision at each recall level, with the ERP following it, and the signature alignment performs much worse than the others. This is caused by the effective complementary information compensation scheme in CVE. Since ERP only captures the information of keyframe similarity and the alignment of different sequences, the videos having much visual variation can not be retrieved with it. While signature alignment with the relationship of neighboring frames considered neglects the visual features of each clip, the spatial information of videos can not be captured, leading to worse matching results. CVE overcomes the weakness of ERP and signature alignment by introducing a complementary scheme into the measure, which produces great improvement of effectiveness.

**Performance of dynamic video insertion.** We test the effect of dynamic sequence insertion on the retrieval accuracy. Initially, we categorized the dataset into 10 categories based on their color features. Then, we randomly choose 50% sequences from each category, so 50% of clips in the whole dataset were selected. We perform clustering on the keyframes of these sequences, and build the corresponding index. Then we dynamically insert sequences (from the remaining 50% sequences in the dataset) into the database. During sequence insertion, existing clusters may be split and new clusters may be generated. As shown in Figure 7(b), with more sequences inserted, the retrieval accuracy keeps steady. This illustrates the good scalability of our system and high performance with dynamic update.

**6.3.2 Efficiency of SUDS**

In this part, we first compare different query summarization methods, and study their effects on the mapping performance. Then we compare different candidate filtering strategies, and examine what effect they have on reducing the cost of video retrieval.

**Effect of query summarization method.** During query mapping, a query video which consists of a number of keyframes may need to be summarized first to reduce the cost. Since the number of keyframes in each video may be varying even for the equal length videos, the cost of mapping the whole sequence to similar video segments is highly related to the number of keyframes. In this set of tests, to be objective, we compare the
maphier performance for each keyframe when the naive method and query summarization are used, by varying the dataset size from 30 to 240 hours and the query length from 20% to 100% of each original one.

Figure 8 (a), (b) and (c) show the effect of query summarization on the query length, the datasize change and the dimensionality varying respectively. As shown in Figure 8 (a), with the query length increasing, the mapping cost for each keyframe keeps steady. While for query clustering, we see a slow drop on the average mapping cost. Meanwhile, from Figure 8 (b), we can see, with the increasing of dataset size, the computation costs of both methods increase. Compared with the naive method, query clustering optimizes the computation cost greatly with a slight increase of cost by dataset size increasing. We test the overall time for query mapping using two methods. The time for local clustering method includes the clustering and mapping time. Comparing with naive method, local clustering method increases mapping speed by 1.62, 1.95, 2.31 and 2.6 times for the 30, 60, 120 and 240 hours video data collections respectively. Thus, with the increasing of dataset size, the superiority of local clustering is more obvious. Moreover, as shown in Figure 8 (c), with the increasing of dimensionality, the mapping cost for local clustering method is much lower than that for naive method.

**Effect of different filtering strategies.** Next, we first compare MaxGap, IFD, SMD and DUAL that is the combination of IFD and SMD in terms of the filtering capability of each. To do so, we introduce a metric called **filter factor**, which is defined by the number of candidates being filtered out divided by the total number of candidates. Figure 9 (a) and (b) report the results by varying the video similarity threshold \(T_{ss}\) and the size of dataset \(N\) respectively.

Clearly, for MaxGap, IFD and DUAL, as the sequence similarity threshold increases, their filter factors decrease, although in different speeds. This is mainly because, with a large sequence dissimilarity threshold, while the MaxGap and IFD of the compared sequences keep steady, more videos are regarded as similar due to the relax similarity limitation, thus more candidates are taken into consideration. For SMD, with the increasing of \(T_{ss}\), more candidates can be judged as similar without verification, thus stronger filtering ability is obtained. Furthermore, considering the lower bounds of CVE, we notice DUAL holds the best pruning ability, followed by IFD and MaxGap. This is because, DUAL performs the filtering with a hybrid approach, the combination of IFD and SMD, accordingly, the filtering ability is enhanced by the optimization of each other. Meanwhile, IFD is a much tighter lower bound of CVE than MaxGap, thus more effective filtering is obtained. In Figure 9(b), for DUAL, as the size of dataset increases, its filter factor increases, though slowly. This indicates the filtering stability of DUAL method. While for MaxGap, a slow decrease of its filter factor can be noticed as the size of dataset increases. Also, from the figure, we can see that, DUAL is much more effective than MaxGap. For a given dataset, DUAL filters out more than 90% candidates, while MaxGap filters out around 40% candidates.
In this paper, we proposed a new technique SUDS that supports content based video retrieval with a subspace based visual dictionary and the extended string matching techniques. The core of SUDS is to map each video keyframe to a word that is the combination of multiple cluster *ids*. We first proposed a space partition method that adaptively extracts the dominant subspaces without looking at dimensions consecutively, so the discrimination power of each subspace is maximized. Then, we presented an effective distance function, CVE, to measure the similarity between two video clips by taking into account the visual features and sequence contexts of them. Finally, we introduced a two-tier indexing scheme, based on which a series of query optimizations including the query summarization and candidate sequence pruning were proposed. Extensive experiments have verified that SUDS outperforms the existing solutions in terms of effectiveness and efficiency.

**Acknowledgments**

This work was supported by ARC grants (DP0663272 and LE0668542) and Hong Kong RGC grants under project No. 611608. The authors would like to thank Yijun Li for providing video data. Thanks also go to Heng Tao Shen, Jie Shao and Zhu Liu for useful discussions.

**References**


Xiangmin Zhou received the Ph.D. degree in computer science from The University of Queensland, Australia, in 2006. Currently, she is a Postdoctoral Research Fellow in the ICT Center, Commonwealth Scientific and Industrial Research Organization (CSIRO), Canberra, Australia. Her research interests include multimedia information retrieval, multimedia database management, indexing and query processing.

Jiaxin Shi received the BS degree in Computer Science and Engineering from Tsinghua University, China, in 2012. She is currently working toward the PhD degree in Computer Science and Engineering at Texas A&M University, College Station, TX. Her research interests include multimedia content analysis and understanding.

Xiaofang Zhou (SM06) received the B.Sc. and M.Sc. degrees in computer science from Nanjing University, China, in 1984 and 1987, respectively, and the Ph.D. degree in computer science from The University of Queensland, Australia, in 1994. He is a Professor of computer science with The University of Queensland. He is the Head of the Data and Knowledge Engineering Research Division, School of Information Technology and Electrical Engineering. He is also the Director of ARC Research Network in Enterprise Information Infrastructure (EII), and a Chief Investigator of ARC Centre of Excellence in Bioinformatics. From 1994 to 1999, he was a Senior Research Scientist and Project Leader in CSIRO. His research is focused on finding effective and efficient solutions to managing, integrating and analyzing very large amount of complex data for business and scientific applications. His research interests include spatial and multimedia databases, high performance query processing, web information systems, data mining, bioinformatics, and e-research.

Yanfeng Shu received her BSc from Harbin Institute of Technology, China, in 1993, MSc from SouthEast University, China, in 1996, and PhD from National University of Singapore in 2006. From 2006 to 2007, she worked as a post-doctoral fellow at the University of Queensland, Australia. Currently she is a research scientist at CSIRO Tasmanian ICT centre. Her research interests include data integration, and query processing and optimization in databases, information retrieval, P2P, and sensor networks.

Lei Chen received the BS degree in Computer Science and Engineering from Tianjin University, China, in 1994, the MA degree from Asian Institute of Technology, Thailand, in 1997, and the PhD degree in computer science from University of Waterloo, Canada, in 2005. He is now an assistant professor in the Department of Computer Science and Engineering at Hong Kong University of Science and Technology. His research interests include multimedia and time series databases, sensor and peer-to-peer databases, and stream and probabilistic databases. He is a member of the IEEE.

Athman Bouguettaya received the PhD degree in computer science from the University of Colorado, Boulder, in 1992. He is a science leader in the ICT Center, Commonwealth Scientific and Industrial Research Organization (CSIRO), Canberra, Australia. He was previously a tenured faculty member in the Department of Computer Science, Virginia Polytechnic Institute and State University (commonly known as Virginia Tech), Blacksburg. His current research interests are in service-oriented computing. He is on the editorial boards of several journals including the IEEE Transactions on Services Computing, International Journal on Web Services Research, the VLDB Journal, the Distributed and Parallel Databases Journal, and the International Journal of Cooperative Information Systems. He was invited to be a guest editor of a special issue of Internet Computing on database technology on the Web. He also guest edited a special issue of the ACM Transactions on Internet on Semantic Web Services. He served as a program chair of the 2008 International Conference on Service Oriented Computing (ICSOC) and the IEEE RIDE Workshop on Web Services for E-Commerce and E-Government (RIDE-WSECEG 04). He served on numerous program committees of database and service-oriented computing conferences. He is a senior member of the IEEE and the ACM.

John A. Taylor received the Ph.D degree in Air Quality Modeling from the Australian National University. He is currently Leader of CSIROs Computational and Simulation Sciences research group, formerly called Terabyte Science. His scientific developments across broad areas of science including large sensor networks such as radio telescopes, large experiments such as the Synchrotron, and high content, high throughput DNA analysis systems all generate large and complex datasets.