Scientific Research Publishing

Journal of Signal and Information Processing, 2018, 9, 73-91
http://www.scirp.org/journal/jsip
ISSN Online: 2159-4481
ISSN Print: 2159-4465

Similar Video Retrieval via Order-Aware Exemplars and Alignment

Teruki Horie[1], Masato Uchida[1], Yasuo Matsuyama[2]

[1]Department of Computer Science and Engineering, Waseda University, Tokyo, Japan
[2]Faculty of Science and Engineering, Waseda University, Tokyo, Japan
Email: teruki.horie@gmail.com, m.uchida@waseda.jp, yasuo2@waseda.jp

How to cite this paper: Horie, T., Uchida, M. and Matsuyama, Y. (2018) Similar Video Retrieval via Order-Aware Exemplars and Alignment. *Journal of Signal and Information Processing*, **9**, 73-91.
https://doi.org/10.4236/jsip.2018.92005

Received: March 7, 2018
Accepted: May 27, 2018
Published: May 31, 2018

Abstract

In this paper, we present machine learning algorithms and systems for similar video retrieval. Here, the query is itself a video. For the similarity measurement, exemplars, or representative frames in each video, are extracted by unsupervised learning. For this learning, we chose the order-aware competitive learning. After obtaining a set of exemplars for each video, the similarity is computed. Because the numbers and positions of the exemplars are different in each video, we use a similarity computing method called M-distance, which generalizes existing global and local alignment methods using followers to the exemplars. To represent each frame in the video, this paper emphasizes the Frame Signature of the ISO/IEC standard so that the total system, along with its graphical user interface, becomes practical. Experiments on the detection of inserted plagiaristic scenes showed excellent precision-recall curves, with precision values very close to 1. Thus, the proposed system can work as a plagiarism detector for videos. In addition, this method can be regarded as the structuring of unstructured data via numerical labeling by exemplars. Finally, further sophistication of this labeling is discussed.

Keywords

Similar Video Retrieval, Exemplar Learning, M-Distance, Sequence Alignment, Data Structuring

1. Introduction

The compilation of videos, or moving images, through the Internet is growing rapidly. This situation is due to the spread of smartphones and sensor cameras. The ease of the collection and upload of videos has created a contemporary unorganized data structure that hinders the retrieval of appropriate videos.

Videos on the web usually have associated text, such as a title, description, and category, that has been annotated by hand. Such text-based meta information describes the content of a whole video. This means that the temporal resolution of textual information is low, and it cannot represent well temporally local context. Thus, retrieving a time span in videos according to visual context is impractical for text-based video retrieval systems. To enhance the temporal resolution, we would need to create more textual labels for a video, which incurs additional cost. In addition, the retrieved results are still obtained based on only the textual labels of the videos, not the visual contents.

Therefore, in this paper, we present a set of learning algorithms and their systems for content-based video retrieval. The concept and variants of this type of retrieval are surveyed in [1]. In our content-based video retrieval, each query is a video itself. Because our system does not need text-based meta information, we retrieve videos based on their visual contents without any manual labeling. For this purpose, we investigate machine learning methods so that the unorganized video set can be organized by a set of numerical labels for retrieval.

To automatically generate numerical labels for videos, we perform the following steps.

Step 1: We extract the representative features for each frame of each video. (Note that every frame of a video is a still image.) Such features correspond to the labels.

Step 2: Using the frame-wise label, a set of representative frames, or exemplars, is determined by an inter-frame learning algorithm. The set of order-aware exemplars with their positions and followers becomes the final label for each video.

Step 3: We use matching algorithms to compare two videos using these labels. This method is able to compute the similarity of two videos through the positions of the exemplars and their number of the followers.

Step 4: For the results of a given query video, the retrieval ranking is presented according to similarity score.

The feature extraction for each frame of **Step 1** can be implemented using different methods by a system designer. Our final selection in this system is the Frame Signature of ISO/IEC [2] so that the total system is practical. In **Step 2**, we present an unsupervised learning algorithm for the exemplar identification. Its variants are presented and the speed and performance of these algorithms are compared. **Step 3** presents a novel similarity comparison for exemplar sets. This method is called the M-distance, and it generalizes the Levenshtein distance [3], dynamic time warping [4], Needleman-Wunsch algorithm [5], and the Smith-Waterman algorithm [6].

The organization of the rest of this paper is as follows. In Section 2, we explain the concept of content-based similar video retrieval. Section 3 presents feature extraction and conversion methods to generate video descriptors for each frame. In Section 4, we present a set of inter-frame learning algorithms for selecting order-aware exemplars. Section 5 is devoted to the presentation of the M-distance similarity computation. Section 6 shows the performance of the designed system, which is evaluated by 11-point interpolated precision-recall curves. The results demonstrate that the proposed system is applicable to detect the unlicensed insertion of video clips. In the concluding remarks of Section 7, we discuss the structuring of general data by more sophisticated descriptors.

2. Content-Based Video Retrieval

Content-based video retrieval is the problem of retrieving videos from a database that are similar to an input video. The overall process of our system is illustrated in Figure 1. Because both the query and the database are videos themselves, content-based video retrieval systems need to analyze the visual features of videos to handle video data. To search for similar videos, systems also need to be able to measure the similarity between a query video and those in the database.

Let a collection of videos be $\mathcal{V}$, comprising elements of video $V \in \mathcal{V}$. Each video is a time series of frame images $V = \{\boldsymbol{v}_t\}_{t=1}^{T}$ of length T. Every frame image is indexed by t, from 1 to T. Here, we simply denote one of the videos in collection $\mathcal{V}$ as V, without an index to avoid complicated notation. Although we omit the index, lengths and contents of videos in $\mathcal{V}$ can vary.

The first step of the process is to label videos with numerical features. This step is shown in the upper box in Figure 1. As preprocessing, we extract a feature of each video frame $\boldsymbol{v}_t$ as its descriptor $\boldsymbol{x}_t$ using an existing method mentioned in Section 3. We hence obtain $X = \{\boldsymbol{x}_t\}_{t=1}^{T}$ as the descriptor for V. The length of X is exactly the same as V because the extraction of $\boldsymbol{x}_t$ is

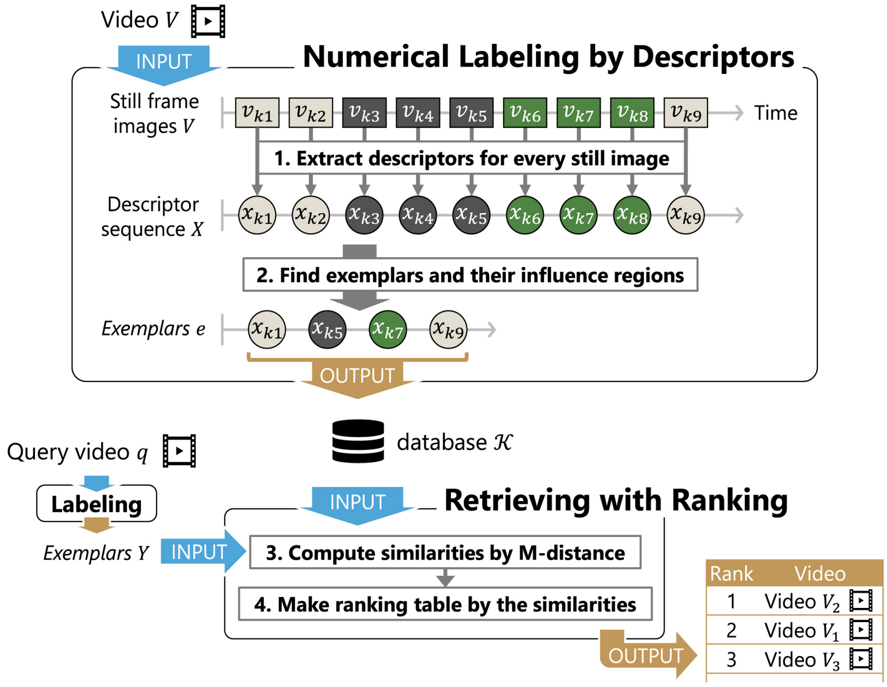

Video V
INPUT
Numerical Labeling by Descriptors
Still frame images V
v_{k1} v_{k2} v_{k3} v_{k4} v_{k5} v_{k6} v_{k7} v_{k8} v_{k9}
Time
1. Extract descriptors for every still image
Descriptor sequence X
x_{k1} x_{k2} x_{k3} x_{k4} x_{k5} x_{k6} x_{k7} x_{k8} x_{k9}
2. Find exemplars and their influence regions
Exemplars e
x_{k1} x_{k5} x_{k7} x_{k9}
OUTPUT
database $\mathcal{K}$
Query video q
Labeling
Exemplars Y
INPUT
INPUT
Retrieving with Ranking
3. Compute similarities by M-distance
4. Make ranking table by the similarities
OUTPUT
Rank Video
1 Video V_2
2 Video V_1
3 Video V_3

Figure 1. Content-based video retrieval using video descriptors.

conducted for each frame image $\boldsymbol{v}_t$. After preprocessing, we extract representative descriptors $\boldsymbol{e}_n \in X, (n = 1, \cdots, N)$ to reduce the size of the video data. These descriptors are called exemplar frames, or simply exemplars. As mentioned in the previous section, we find exemplars and their followers using our machine learning algorithms, not by hand. We also calculate the number of followers E_n for each exemplar $\boldsymbol{e}_n$. As a result, we obtain the feature $\mathcal{E} = \{\boldsymbol{e}_n, E_n\}_{n=1}^{N}$ of a video *V*. Finally, we store $\mathcal{E}$ into a database $\mathcal{K}$.

When we use a query video to search videos in our system, the system retrieves videos in database $\mathcal{K}$ using their features. Because the query is not included in the database generally, the system extracts the feature of the query video before retrieval. Let $V_q = \{\boldsymbol{v}_{qt}\}_{t=1}^{T_q}$ be a query video of length T_q. The system obtains the descriptors of query $X_q = \{\boldsymbol{x}_{qt}\}_{t=1}^{T_q}$ and features $\mathcal{E}_q = \{\boldsymbol{e}_{qn}, E_{qn}\}_{n=1}^{N_q}$ using the same processes used to build $\mathcal{K}$, that is, using machine learning algorithms. We then obtain the ranking of the similarity by comparing $\mathcal{E}_q$ with all $\mathcal{E} \in \mathcal{K}$. This step is shown in the lower box in **Figure 1**. This comparison needs to absorb the difference in cardinalities between $\mathcal{E}_q$ and $\mathcal{E}$. For this purpose, we developed the M-distance measure. M-distance generalizes existing sequence-matching algorithms by incorporating the influence of the followers to the exemplars. Section 5 presents the details of the M-distance algorithms. The ranking is output by the ordering of the similarity scores obtained by the M-distance computation.

3. Extraction of Video Descriptors

Each frame of a video is a still image whose complexity and size may differ from those of other videos. Therefore, we need a universal and concise expression, or a descriptor $\boldsymbol{x}_t$, for frame $\boldsymbol{v}_t$. In our system, we support two types of such descriptors: the Color Structure Descriptor (CSD) [7] and the Frame Signature [2].

3.1. Color Structure Descriptor

The CSD of MPEG-7 is a standard color histogram obtained using a small sliding window. In our system, we first quantize the color information of the image with $\{12, 8, 8\}$-levels in HSV (Hue-Saturation-Value) color space. Then, we use a window of size 8×8 pixels. The window fills $12 \times 8 \times 8 = 768$ color bins to make the the histogram while sliding across the image pixel by pixel. A simple example of this method is illustrated in **Figure 2**. The image on the left is an example image with 20×20 pixels, and the yellow square is the sliding window. In this case, colors 1, 2, 3, and 4 are found in the window at this position, so we add 1 to the associated bins of the histogram. Note that the actual number of colored pixels appeared in the window is not considered when incrementing.

The final histogram is normalized to unity. Therefore, each video frame $\boldsymbol{v}_t$ generates a nonnegative real-numbered vector $\boldsymbol{x}_t$, and the summation of all its

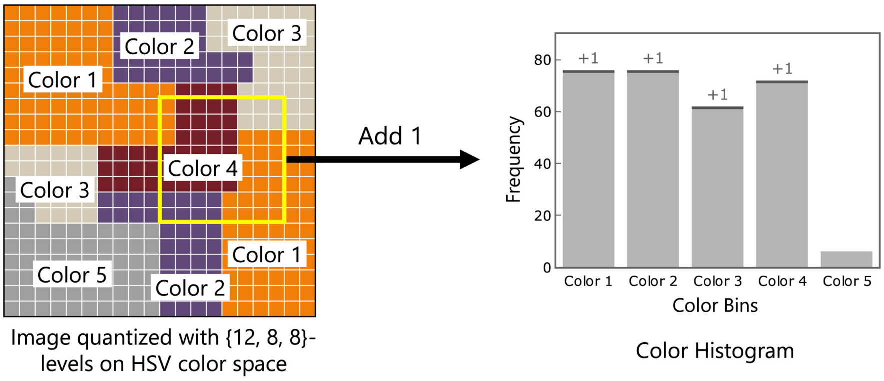


Figure 2. Color structure descriptor.

elements is 1. This vector is 768 dimensions and resides on a 767-dimensional simplex.

3.2. Frame Signature

The Video Signature Tools is an ISO/IEC standard for multimedia contents [8] that is still being developed by additions and enhancements to its toolbox. An important part is the Frame Signature, which is a luminance histogram rather than a color histogram. This change has already been released as an amendment [2]. We employ the Frame Signature as another video descriptors in our system. Here, we only briefly explain the algorithm of the Frame Signature because of its complexity. A complete explanation of the Frame Signature can be found in [2].

For the Frame Signature, we first resample each frame image $\boldsymbol{v}_t$ to $32 \times 32 = 1024$ sub-blocks by dividing the width and height of the image into 32 parts. Each sub-block is assigned the average value of the luminance of the pixels in the block. The luminance is the Y component of YCbCr color, which is computed from RGB components [9].

$$\mathrm{Y} = [0.299, 0.587, 0.114][\mathrm{Red}, \mathrm{Green}, \mathrm{Blue}]^{\mathrm{T}} \tag{1}$$

Note that the range of an RGB component is $[0,1]$, and the value of the Y component is quantized to 256 levels. Thus, we obtain a monochrome video $\hat{V} = \{\hat{\boldsymbol{v}}_t\}_{t=1}^{T}$, where $\hat{\boldsymbol{v}}_t$ contains the 1024 luminance values of the sub-blocks obtained by Equation (1).

Second, we generate a descriptor X from $\hat{V}$. The Frame Signature algorithm provides a 380-dimensional vector $\boldsymbol{x}_t$ in which every element x_{td} takes a ternary value of $\{0,1,2\}$ according to $\hat{\boldsymbol{v}}_t$. For the *d*-th element x_{td} in $\boldsymbol{x}_t$, the algorithm determines its value using the average luminance of a sub-region, which is a region composed by several sub-blocks. A sub-region is linked to a single dimension d of $\boldsymbol{x}_t$, and the composition of the sub-blocks in each sub-region is defined by the standard in [2]. Here, let y_d be the average luminance of the sub-region corresponding to x_{td}. Each element x_{td} is determined as follows.

- For the first 32 dimensions $d = 1, \cdots, 32$, x_{td} is computed using the corresponding y_d as follows.

$$x_{td} = \begin{cases} 2, & \text{if } y_d - 128 > \theta_{\text{type}} \\ 1, & \text{if } \left| y_d - 128 \right| \le \theta_{\text{type}} \\ 0, & \text{if } y_d - 128 < -\theta_{\text{type}} \end{cases} \quad (2)$$

Threshold θ_{type} is determined by the y_d values across some of the dimensions [2].

- For the remaining parts $d = 33, \cdots, 380$, the computation of x_{td} compares two non-overlapping sub-regions y_{d1} and y_{d2} using threshold θ_{type}, as defined in [2].

$$x_{td} = \begin{cases} 2, & \text{if } y_{d,1} - y_{d,2} > \theta_{\text{type}} \\ 1, & \text{if } \left| y_{d,1} - y_{d,2} \right| \le \theta_{\text{type}} \\ 0, & \text{if } y_{d,1} - y_{d,2} < -\theta_{\text{type}} \end{cases} \quad (3)$$

Finally, we obtain each $\boldsymbol{x}_t$, which is a vector of $\{0,1,2\}^{380}$. The above steps are applied to each frame to obtain the final descriptor X of video V [10].

4. Machine Learning Algorithms for Exemplar Selection

In this section, we explain our machine learning methods that can find exemplars and their influence regions from frame descriptor sequence X. The influence region is the span of follower frames that reside on both sides of an exemplar. We obtain a set of exemplars automatically by machine learning. Five clustering algorihtms were considered for this system. They are described as follows.

4.1. Affinity Propagation Type

Time-Bound Affinity Propagation (TBAP)

This method is based on affinity propagation [11]. However, we found that affinity propagation, in its basic form, cannot reflect the sequential nature of video frames when obtaining an exemplar set. Therefore, we introduced a time-bound property that uses a sliding window for the frames. This TBAP method has been successful in content-based video retrieval [12]. However, our later studies found that other exemplar selection methods, that is, generalized competitive learning and vector quantization methods, are less computationally expensive. Therefore, we omit further details regarding this algorithm.

4.2. Competitive Learning Types

We consider two competitive learning methods. The first type is the harmonic competition method [13] or generalized *k*-means. In these methods, we obtain pseudo exemplars, which are computed as centroids of frames. Then, we identify the nearest neighbor frames to the centroids as the exemplars. We consider two learning algorithms that differ in the method of reflecting the sequential properties of the frames.

4.2.1. Time-Partition *k*-Means (TPKM)

First, we divide $V = \{\boldsymbol{v}_t\}_{t=1}^{T}$ into non-overlapping frame sets for the time axis using a fixed length *b*. Thus, we have a total of $\lceil T/b \rceil$ blocks. In each block, a set of centroids is computed by the *k*-means method, *i.e.*, the vector quantization. These centroids are pseudo exemplars because they are not video frames. Therefore, we find the frame that is the nearest to each pseudo exemplar. The set of such frames comprises the exemplars.

4.2.2. Time-Split *k*-Means (TSKM)

In our preliminary experiments, the TPKM method often generates large clusters that contain elements over a wide range of times. This case occurs when block length b is large. In such a case, it is desirable to split clusters containing elements that are distant in time. Except for this splitting mechanism, the rest of this method is the same as TPKM.

Our preliminary experiments proved that TPKM and TSKM are significantly faster than TBAP by two orders of magnitude. However, TBAP still has theoretical merit. That is, exemplars can be obtained without computing the centroids as pseudo exemplars. Therefore, we consider *k*-means methods that can identify exemplars directly. For this reason, we omit further details regarding TPKM and TSKM, however, they can be found in [14].

4.2.3. Modified Pairwise Nearest Neighbor

Next, we pay attention to an approximate *k*-means method called the pairwise nearest neighbor vector quantization (PNN-VQ, or simply PNN) [15] which inherits the sequential decimation of data points of [16]. We further modify the original PNN as follows so that exemplars can be obtained directly [14].

Step 1: Set δ to the desired (non-negative) minimum distance between exemplars.

Step 2: Compute the centroid of all data points, say $\boldsymbol{c}$.

Step 3: Find the two points that are the closest in distance. This is the nearest neighbor pair.

Step 4: If the distance is equal to or less than threshold δ, then go to **Step 5**; otherwise, go to **Step 6**.

Step 5: Remove the member of the pair that is farther from centroid $\boldsymbol{c}$. Note that the original PNN removes both points but inserts their centroid.

Step 6: Output the remaining data points as exemplars.

Based on our PNN method, we propose the following learning algorithms to obtain a feature of a video data.

4.2.4. Time-Bound PNN (TB-PNN)

The TB-PNN method is almost same as the modified PNN above but uses a different method to find the closest pair. The modified PNN measures the distances of all data points to find the closest pair. However, TB-PNN only measures distances from a data point and its surrounding ones along the time axis. This mechanism reduces the range over which the distance between data points must be measured. Consequently, an exemplar will not possess data points distant in time. Thus, we can find exemplars while taking the order of time into account. The length of the bounding range is a design parameter, that must be given.

4.2.5. Time-Partition PNN (TP-PNN)

The TP-PNN algorithm is similar to TPKM but it utilizes our modified PNN instead of *k*-means. First, we divide $V=\{\boldsymbol{v}_t\}_{t=1}^{T}$ into non-overlapping frame sets along the time axis by a fixed length *b*. Thus, there are $\lceil T/b \rceil$ blocks. Then, for each block, exemplars are obtained using the modified PNN.

In this work, we chose TP-PNN for our final content-based video retrieval system because of the results of our preliminary experiments. We hence present more details of TP-PNN here. Figure 3 shows the partitioning and the PNN that yields exemplars. In the example in this figure, descriptor sequence $X=\{\boldsymbol{x}_t\}_{t=1}^{T}$ for a video V is given. The length of the video is $T=11$ and the block length is $b=3$. The TP-PNN is calculated as follows.

Step 1: Divide the descriptor sequence X into $\lceil T/b \rceil$ blocks.

Step 2: For each partition:

Use the modified PNN to obtain the exemplar(s). Although each exemplar can be any of $\{\boldsymbol{v}_t\}_{t=1}^{T}$, here, we denote it by $\boldsymbol{e}_n$. A descriptor that appears at the *n*-th item in the exemplars along the time axis in X is denoted by $\boldsymbol{e}_n$.

Meanwhile, we determine the number of data points following the exemplars. The followers removed at **Step 5** of the modified PNN indicate the time range possessed by an exemplar. The method is as follows: if we leave $\boldsymbol{e}_n$ and remove $\boldsymbol{e}_{n'}$ because $\boldsymbol{e}_{n'}$ is more distant from the centroid, then we set $E_n \leftarrow E_n + E_{n'} + 1$. The last constant term 1 means that $\boldsymbol{e}_{n'}$.

Step 3: Output all the results acquired from all partitions in **Step 2** as $\{\boldsymbol{e}_n, E_n\}_{n=1}^{N}$.

The output $\{\boldsymbol{e}_n, E_n\}_{n=1}^{N}$ is the feature of a video *V*. Each exemplar $\boldsymbol{e}_n$ possesses the E_n followers which are eliminated points. In the example in Figure 3, we obtained the feature $\{\boldsymbol{e}_n, E_n\}_{n=1}^{5}$ of a video V that has five exemplars.

In actual cases, the total number of exemplars N varies because of threshold δ

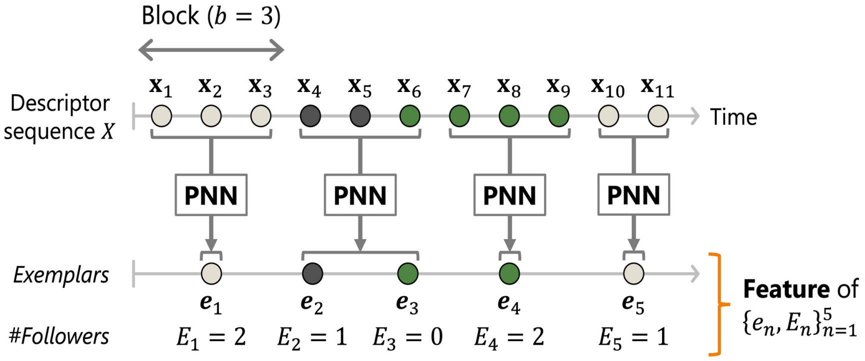

Block (b = 3)
Descriptor sequence X
$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_3$ $\mathbf{x}_4$ $\mathbf{x}_5$ $\mathbf{x}_6$ $\mathbf{x}_7$ $\mathbf{x}_8$ $\mathbf{x}_9$ $\mathbf{x}_{10}$ $\mathbf{x}_{11}$
Time
PNN
PNN
PNN
PNN
Exemplars
$\boldsymbol{e}_1$ $\boldsymbol{e}_2$ $\boldsymbol{e}_3$ $\boldsymbol{e}_4$ $\boldsymbol{e}_5$
#Followers
$E_1 = 2$ $E_2 = 1$ $E_3 = 0$ $E_4 = 2$ $E_5 = 1$
Feature of V
$\{e_n, E_n\}_{n=1}^{5}$

Figure 3. Visualization of the time-partition pairwise nearest neighbor algorithm.

of the modified PNN. The data size of a video is clearly reduced by TP-PNN. Therefore, TP-PNN works as a data compressor.

5. Similar Video Retrieval

To find similar videos, we need a method to compare the exemplars. In addition, the comparison methods should reflect the number of followers of each exemplar. Therefore, we present a set of methods that extend the sequence alignment method. Sequence alignment is an algorithm that discovers the best matching pattern of two sequences and provides its degree of fitness as their similarity. We refer to the degree of fitness as the matching score, or simply the score. Our methods are based on the Levenshtein distance (L-distance) [3], dynamic time warping [4], Needleman-Wunsch algorithm for global alignment [5], and Smith-Waterman algorithm for local alignment [6]. We call this set of methods the M-distance after the name of the "L-distance" as well as the original inventors (Matsuyama and Moriwaki). The set consists of two methods: one for the global alignment and the other for the local alignment. Thus, we have a global alignment M-distance and a local alignment M-distance.

5.1. Sequence Alignment for Similarity Computation

The computation of sequence alignment algorithms can be done given the similarity between any two elements comprising two sequences. In our similar video retrieval system, a sequence is an array of exemplars $\{\boldsymbol{e}_n\}_{n=1}^{N}$. Generally speaking, a sequence is an array of descriptors $\{\boldsymbol{x}_t\}_{t=1}^{T}$ because an exemplar sequence is a (sub-)sequence of descriptors. We denote the similarity between elements $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ by $s(i,j)$. The similarity must satisfy the following.

- $s(i,j)=0$ if and only if $\boldsymbol{x}_i=\boldsymbol{x}_j$.
- $s(l,i)>s(i,j)$ if $\boldsymbol{x}_i$ is more similar to $\boldsymbol{x}_l$ than $\boldsymbol{x}_j$.

Because we use a distance measure in the exemplar selection, we can formulate the similarity measure using that distance measure, say $d(i,j)$, as follows.

$$s(i,j)=\bar{D}-d(i,j) \tag{4}$$

Here, $\bar{D}$ is a constant value determined by a system designer. It indicates the maximum similarity between elements as $d(i,j)=0$ if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are identical. In addition, we can find the upper bound $\bar{D}_{\max}$ theoretically as the distance between the vertex and the centroid of a simplex. Therefore, the value of $\bar{D}\geq 0$ is used after the exemplars have been learned.

In practical situations, the lengths of sequences must differ. To arrange such sequences, we need to make the lengths even using padding. Further, if a similar or the same pattern of a sub-sequence in one sequence is not found in the other, we create padding that indicates there is no pattern. Within the sequence alignment, such padding is called a gap. A gap is interpreted as a special element of the exemplars. The similarity between a gap and a normal element is usually a negative constant value. This is because inserting gaps should reduce the total similarity of the sequences. For this reason, it is referred to as a gap penalty.

In the next two sections, we describe the two alignment M-distance algorithms. For clarity, we provide the notation here. The symbols are also described in Section 2. We compare two videos A and B. Each video consists of a sequence of frame images $V_A = \{\boldsymbol{v}_{At}\}_{t=1}^{T_A}$ and $V_B = \{\boldsymbol{v}_{Bt}\}_{t=1}^{T_B}$ with lengths T_A and T_B, respectively. Using a video descriptor method (CSD or Frame Signature), the descriptor sequences $X_A = \{\boldsymbol{x}_{At}\}_{t=1}^{T_A}$ of video A and $X_B = \{\boldsymbol{x}_{Bt}\}_{t=1}^{T_B}$ of video B are obtained. Then, we find features $\mathcal{E}_A = \{\boldsymbol{e}_{An}, E_{An}\}_{n=1}^{N_A}$ and $\mathcal{E}_B = \{\boldsymbol{e}_{Bn}, E_{Bn}\}_{n=1}^{N_B}$ by applying an exemplar selection method, *i.e.*, TP-PNN.

5.2. Global Alignment M-Distance

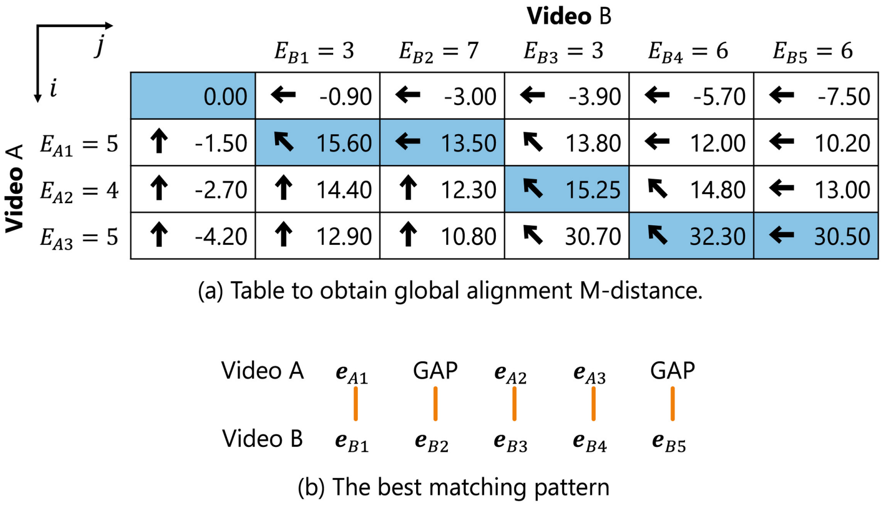
The global alignment M-distance computes the similarity between videos A and B over their whole lengths through features $\mathcal{E}_A$ and $\mathcal{E}_B$. Our algorithm for the global alignment M-distance computes $\mathcal{E}_A$ and $\mathcal{E}_B$. Suppose that we have two videos A and B whose lengths of features are $N_A = 3$ and $N_B = 5$, respectively. To grasp the overall process of the global alignment easily, we provide **Figure 4**, in which the gap penalty is $g = 0.3$. In each video, the number of followers of each exemplar is shown in **Figure 4(a)**. The algorithm of the global alignment M-distance consists of the following steps.

Step 1: Prepare an $(N_A + 1) \times (N_B + 1)$ table. Each cell (i, j) has a single value $f(i, j)$ that indicates the maximum fitness of the matching pattern between videos A and B from the beginning position $(0,0)$ to (i, j).

Step 2: Set $f(0,0) = 0$.

Step 3: Fill the first row by $f(0, j \geq 1) = -g \sum_{j'=1}^{j} E_{Bj'}$ and place a left arrow ← in each cell.

Step 4: Fill the first column by $f(i \geq 1, 0) = -g \sum_{i'=1}^{i} E_{Ai'}$ and place an upward arrow ↑ in each cell.



		Video B				
		$E_{B1} = 3$	$E_{B2} = 7$	$E_{B3} = 3$	$E_{B4} = 6$	$E_{B5} = 6$
	0.00	← -0.90	← -3.00	← -3.90	← -5.70	← -7.50
Video A $E_{A1} = 5$	↑ -1.50	↖ 15.60	← 13.50	↖ 13.80	← 12.00	← 10.20
$E_{A2} = 4$	↑ -2.70	↑ 14.40	↑ 12.30	↖ 15.25	↖ 14.80	← 13.00
$E_{A3} = 5$	↑ -4.20	↑ 12.90	↑ 10.80	↖ 30.70	↖ 32.30	← 30.50

(a) Table to obtain global alignment M-distance.

(b) The best matching pattern

Figure 4. Example of a global alignment M-distance ($g = 0.3$). The value of each cell is a score $f(i, j)$, and the arrow is a direction to traceback. The cells filled with blue are the best matching patterns of A and B found by the global alignment M-distance.

Step 5: Starting from the position $i = j = 1$, fill all the cells in the table by

$$f(i,j) = \max\left\{ f(i-1,j) - gE_{Ai}, f(i-1,j-1) + r(i,j)s(i,j), f(i,j-1) - gE_{Aj} \right\}. \quad (5)$$

Here, r is a coefficient of similarity that reflects the magnitude of the followers, and s is the similarity between the exemplars mentioned above. While filling the table, we insert an arrow as a pointer from position (i,j) to the position of $\{(i-1,j),(i-1,j-1),(i,j-1)\}$ that gives the maximum score.

Step 6: (Tracebacking) Traceback from the end of cells at (N_A, N_B) in the table to $(0,0)$ along pointer arrows. The alignment of the exemplars $\{\boldsymbol{e}_{An}\}_{n=1}^{n_A}$ and $\{\boldsymbol{e}_{Bn}\}_{n=1}^{n_B}$ is the result of this traceback. The three types of arrows in the path have the following meaning.

- A diagonally upward arrow ↖ from (i,j) to $(i-1,j-1)$ means that the exemplar i in the column and the exemplar j in the row are matching elements. Here, "matching" need not mean identical, but similar. This case is also called a "substitution" or "match."
- The horizontal arrow ← from (i,j) to $(i,j-1)$ means that a gap is inserted at i of $\{\boldsymbol{e}_{Ai}\}_{i=1}^{N_A}$. The gap insertion is equivalent to the deletion of $\boldsymbol{e}_{Bj}$.
- The upward arrow ↑ from (i,j) to $(i-1,j)$ means that a gap is inserted at j of $\{\boldsymbol{e}_{Bj}\}_{j=1}^{N_B}$. The gap insertion is equivalent to the deletion of $\boldsymbol{e}_{Ai}$.

Step 7: (Normalizing the global alignment score) Normalize the total score $f_{\text{last}} = f(N_A, N_B)$ depending on the lengths of A and B by the following formula.

$$u(A,B) = \frac{f_{\text{last}}}{w\left(\{E_{Ai}\}_{i=1}^{N_A}, \{E_{Bj}\}_{j=1}^{N_B}\right)} \quad (6)$$

where w is an averaging function.

Step 8: Output the best matching pattern found by the traceback in **Step 6** and the score $u(A,B)$ as the similarity of videos A and B.

In our experiments, the following function was chosen as $r(i,j)$.

$$r(i,j) = \frac{E_{Ai} + E_{Bj}}{2} \quad (7)$$

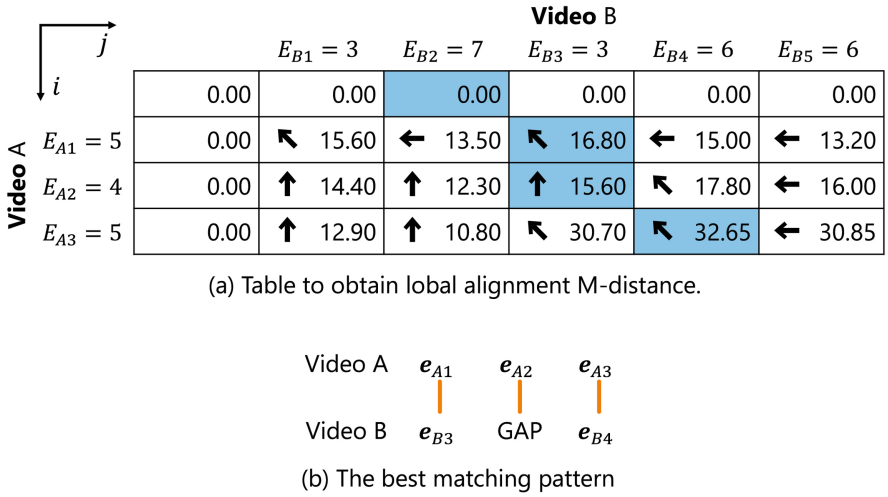
In function $w(a,b)$, we chose to use the algebraic mean of the followers, that is, the mean over all followers of A and B. In this example, the final similarity is $f_{\text{last}} = 30.50$, and the matching pattern is given in Figure 4(b).

5.3. Local Alignment M-Distance

In contrast to the global matching, the aim of the local alignment M-distance is to find matching local time spans in $\mathcal{E}_A$ and $\mathcal{E}_B$. This alignment method is more practical than the global alignment. We provide an example of the local alignment in Figure 5, where the gap penalty is $g = 0.3$, as in Figure 5(a).

Step 1: Prepare an $(N_A+1)\times(N_B+1)$ table.

Step 2: Fill all the cells in the $\{i=0\}$-th row by and $\{j=0\}$-th column all


j
i
Video A
Video B

		$E_{B1}=3$	$E_{B2}=7$	$E_{B3}=3$	$E_{B4}=6$	$E_{B5}=6$
	0.00	0.00	0.00	0.00	0.00	0.00
$E_{A1}=5$	0.00	↖ 15.60	← 13.50	↖ 16.80	← 15.00	← 13.20
$E_{A2}=4$	0.00	↑ 14.40	↑ 12.30	↑ 15.60	↖ 17.80	← 16.00
$E_{A3}=5$	0.00	↑ 12.90	↑ 10.80	↖ 30.70	↖ 32.65	← 30.85

(a) Table to obtain lobal alignment M-distance.

Video A	$\boldsymbol{e}_{A1}$	$\boldsymbol{e}_{A2}$	$\boldsymbol{e}_{A3}$
Video B	$\boldsymbol{e}_{B3}$	GAP	$\boldsymbol{e}_{B4}$

(b) The best matching pattern

Figure 5. Example of a local alignment M-distance ($g = 0.3$).

with 0.

Step 3: Starting from position $i = j = 1$, fill all cells of the table by

$$f(i,j) = \max\left\{0, f(i-1,j) - gE_{Ai}, f(i-1,j-1) + r(i,j)s(i,j), \right. \\ \left. f(i,j-1) - gE_{Bj}\right\}. \tag{8}$$

In this step, we draw an arrow as a pointer from position (i,j) to the position of $\{(i-1,j),(i-1,j-1),(i,j-1)\}$ that gives the non-zero maximum. Note that if we obtain $f(i,j) = 0$, then we do not place an arrow in cell (i,j).

Step 4: Identify the cell that yields the maximum value $f_{\max}$ in this table. This value is the similarity of the local alignment.

Step 5: (Tracebacking) Traceback along the arrows from the cell with $f_{\max}$ until a score of a cell is 0.

Step 6: Output the best matching local span found by the traceback in **Step 5**. Score $f_{\max}$ is the similarity of videos A and B.

In this example, the traceback in **Step 5** starts at $(3,4)$ because the cell at that location gives the maximum matching score in this table. The best similar span of A and B is in Figure 5(b). Note that although the parameters and features $\mathcal{E}_A$ and $\mathcal{E}_B$ are exactly the same as the example of the global alignment, the results are entirely different.

6. Experiments on Video Similarity Ranking

6.1. Video Dataset for Plagiarism Detection

We prepared 2100 videos to form 100 sets of 21 videos from [17] [18]. These videos are treated as non-labeled videos. Their specifications are as follows.

- The video resolution is 640×480 pixels.
- The frame rate is 30 frame per seconds (fps).
- Their lengths are from 30 to 180 (secs). That is, we retrieved videos of different lengths.

In each of the 21 videos, we selected one video as the query at random. Then, we choose more than 10% portions of the query. This part is modified by the following five methods.

1) The frame rate is sped up by removing one frame per 6, 3, or 2 frames.

2) The video was changed to monochromatic scenes using gray-scale transformation.

$$\text{Gray} = [0.29891, 0.58661, 0.11448][\text{Red}, \text{Green}, \text{Blue}]^{\mathrm{T}} \tag{9}$$

Note that the coefficients in the above equation are equivalent to the Y component of YCbCr color space except their value is in higher precision than Equation (1).

3) The brightness of the RGB components were multiplied by 0.6 to 0.9 randomly.

4) The size was randomly changed by a factor of 0.5 to 2.0.

5) The JPEG compression quality was changed using OpenCV, which is an open source image processing library. Specifically, we set the CV_IMWRITE_JPEG_QUALITY parameter to a randomly chosen integer value between 20 and 80 for each video. Its default and maximum values are 95 and 100, respectively.

The videos modified by the above five types were randomly inserted into the remaining 20 videos. Thus, these target videos included plagiarized material.

6.2. Matching Process

In plagiarism detection, the local alignment method of Section 5.3 is appropriate because the global alignment of Section 5.2 absorbs local characteristics as much as possible. To evaluate the set of pseudo-illegal videos prepared by Section 6.1, we used the following steps.

Step 1: A query video q of length T_q is given. Then, its descriptor $X_q = \{\boldsymbol{x}_{qt}\}_{t=1}^{T_q}$ is computed. The video descriptor is either the CSD of Section 3.1 or Frame Signature of Section 3.2.

Step 2: A feature $\mathcal{E}_q = \{\boldsymbol{e}_{qn}, E_{qn}\}_{n=1}^{N_q}$ is computed from $X_q = \{\boldsymbol{x}_{qt}\}_{t=1}^{T_q}$ by TP-PNN, described in Section 4.

Step 3: The local alignment M-distance described in Section 5.3 is computed for the pre-computed exemplar sets in the video database as labels.

Step 4: The similarity ranking is obtained from the M-distance scores.

Note that, for other experiments in which we find whole video properties, we need to use the global alignment M-distance of Section 5.2.

6.3. Evaluation

We use precision-recall curves to evaluate of our content-based similar video retrieval system. Recall and precision are defined as follows.

$$\text{recall} = \frac{\text{number of correctly retrieved videos}}{\text{number of videos in the same class}} \tag{10}$$

$$\text{precision} = \frac{\text{number of correctly retrieved videos}}{\text{number of to pranked vidoes to be checked}} \tag{11}$$

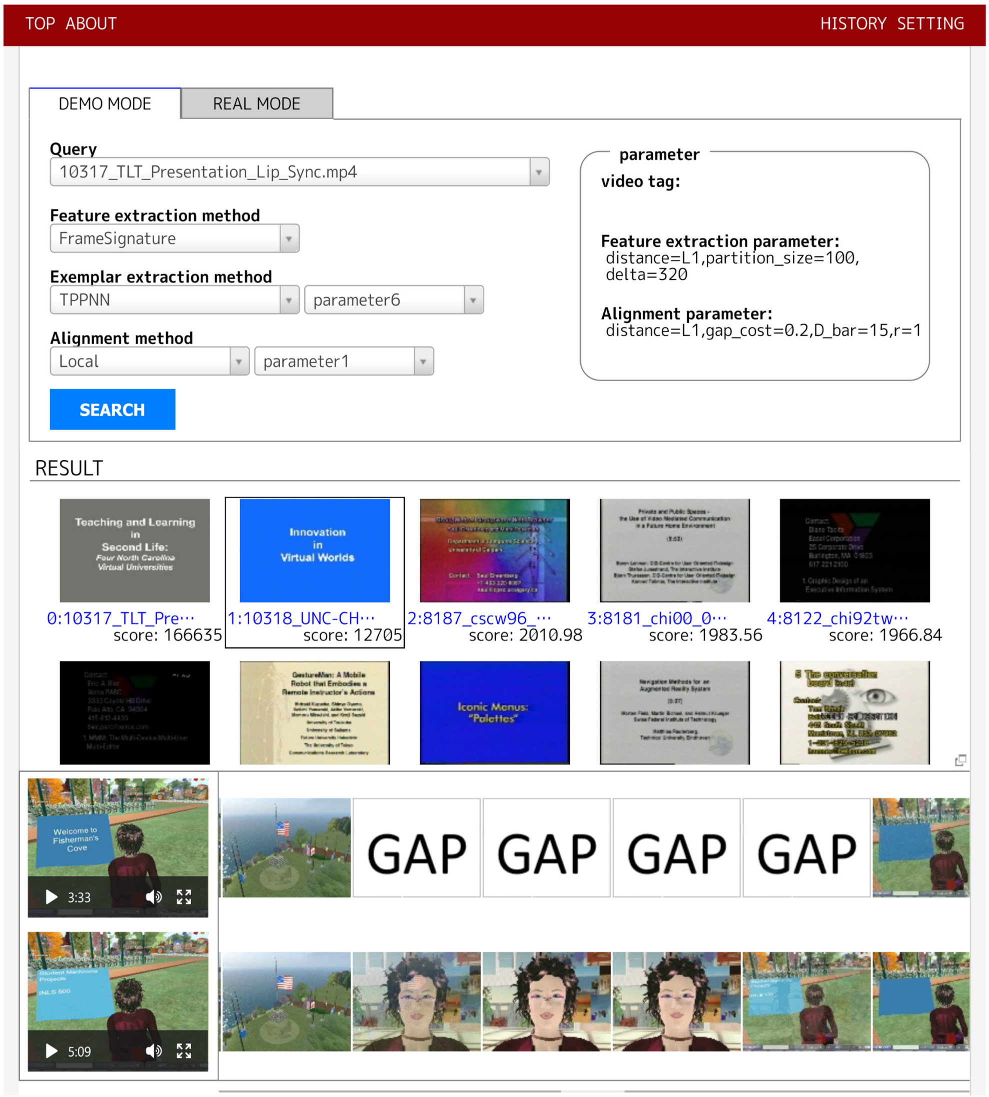
For precision, we use 11-point interpolated precision because our system outputs a ranked result of the plagiarism detection task. An interpolated precision is widely used for ranked results in information retrieval. Let $\text{precision}(r')$ be a precision at recall level r' while we take an item from the top of the ranking. Then, the interpolated precision is defined as follows.

$$\text{precision}_{\text{intp}}(r) = \max_{r' \geq r} \{\text{precision}(r')\} \tag{12}$$

Here, r is one of 11 points of $\{0.0, 0.1, 0.2, \cdots, 1.0\}$. We evaluated performance of algorithms and our system based on the precision$_{\text{intp}}$-recall curves.

6.4. Graphical User Interface Design Experiments

We designed a graphical user interface (GUI) to facilitate the experiments of the similar video retrieval. **Figure 6** presents a screenshot of this system. The top portion of the window specifies a query, feature extraction method, exemplar



Query: 10317_TLT_Presentation_Lip_Sync.mp4

Feature extraction method: FrameSignature

Exemplar extraction method: TPPNN, parameter6

Alignment method: Local, parameter1

parameter

video tag:

Feature extraction parameter: distance=L1,partition_size=100, delta=320

Alignment parameter: distance=L1,gap_cost=0.2,D_bar=15,r=1

0:10317_TLT_Pre··· score: 166635

1:10318_UNC-CH··· score: 12705

2:8187_cscw96_··· score: 2010.98

3:8181_chi00_0··· score: 1983.56

4:8122_chi92tw··· score: 1966.84

Figure 6. Graphical user interface for similar image retrieval.

selection method, and alignment method. The parameter settings for the learning and alignment method are specified here. The middle portion of the window shows thumbnails of the videos ordered by similarity scores. The bottom part of the window shows frames that correspond to the exemplar sets of the query video and the selected video. In this illustration, four gaps are inserted on the query side by the M-distance computation.

6.5. Experimental Results

We conducted experiments on plagiarism detection using the local alignment M-distance. The data were generated by the method described in Section 6.1. There are two sets of experiments depending on the descriptors: CSD or Frame Signature. For these experiments, we used the following computational resources.

1) A conventional standalone machine consisting of two Intel Xeon 2.10 GHz CPUs and 64 GB RAM: This machine was used for the GUI of **Figure 6**, Frame Signature extraction, exemplar identification of TP-PNN, and M-distance matching.

2) Eight virtual machines provided by Amazon Web Services (AWS) that are equivalent to an Intel Xeon 2.8 GHz CPU and 32 GB RAM: These machines were used compute the CSD, TBAP, TPKM, and TB-PNN.

6.5.1. Results Using CSD

First, we show the results yielded by the conventional CSD descriptor. We divided the HSV color space into $\{12,8,8\}$-levels. The distance measure used in the exemplar selection and alignment is an ℓ_2-distance in a 767-dimensional simplex ($767 = 12 \times 8 \times 8 - 1$). Note that the dimension reduction to 767 is due to the normalization of the histogram. In a set of preliminary experiments, we chose the threshold of the nearest neighbor distance to be $\delta = 0.1$ and the block size of TP-PNN to be $b = 100$. For local alignment M-distance, the distance bias was set to $\bar{D} = 0.05$ and the gap penalty was $g = 0.2$.

Figure 7 shows the resultant precision-recall curves. From this figure, we can observe the following.

- The CSD descriptor shows sufficient performance with respect to frame rate and size changes. This performance is due to the ability to detect scattered color pixels.
- For the JPEG quality changes, the CSD descriptor is appropriate only if the top several recalls are used.
- The CSD method is susceptible to changes in brightness.
- If the video is changed to grayscale, the CSD video descriptor is not suitable.

The performance of CSD shows that this video descriptor does not comprise much information about the frames.

6.5.2. Experiments Using Frame Signature

When the descriptor is the Frame Signature, we need not normalize the obtained feature vectors. Therefore, we used a distance measure of ℓ_1-distance for the 380-dimensional Euclidean space in the algorithms of exemplar selection and local alignment. Our preliminary experiments found that the threshold of the nearest neighbor distance is $\delta = 320$. The block size of TP-PNN was the same as the experiments using CSD, *i.e.*, $b = 100$. The distance bias was set to $\bar{D} = 15$ for local alignment. The gap penalty was the same value as the experiments using CSD ($g = 0.2$).

Figure 8 presents the resulting precision-recall curves. From this figure, we observe the following results.

- All precision-recall curves show that the Frame Signature is a creditable descriptor for the content-based video detection.
- The precision-recall curve of the frame rate change degrades more than other changes in high-recall regions. This performance degradation is due to excessively fast forwarding, which is similar to human behavior. However, a choice of a smaller δ can address this problem.
- Because the computation of the Frame Signature is based on integers, its speed is faster than the floating-point computation of the CSD vectors by two orders of magnitude. This property is a great advantage in addition to the retrieval precision.

7. Discussions

The content-based similar video retrieval system presented in this paper showed high performance, as evidenced by Figure 8 when the Frame Signature is used. This system is already viable as a practical system. Comparing its performance with that of Figure 7, whose descriptor is just CSD, we have the following

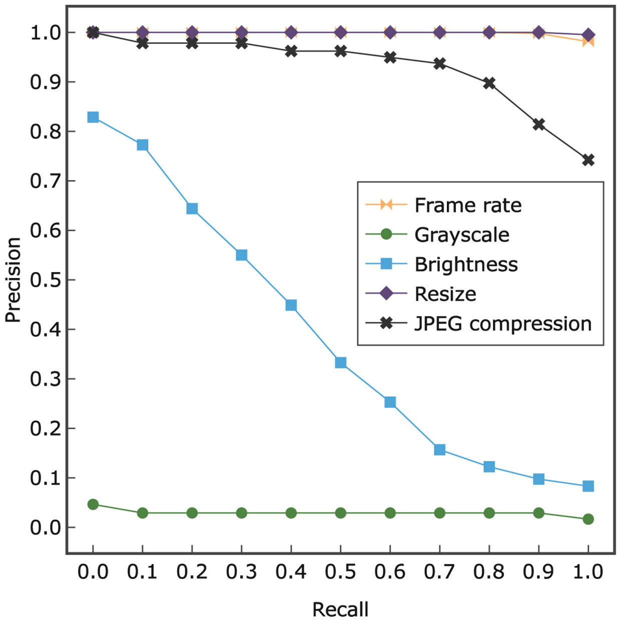


Figure 7. Precision-recall curves for similarity detection using color Structure Descriptor.

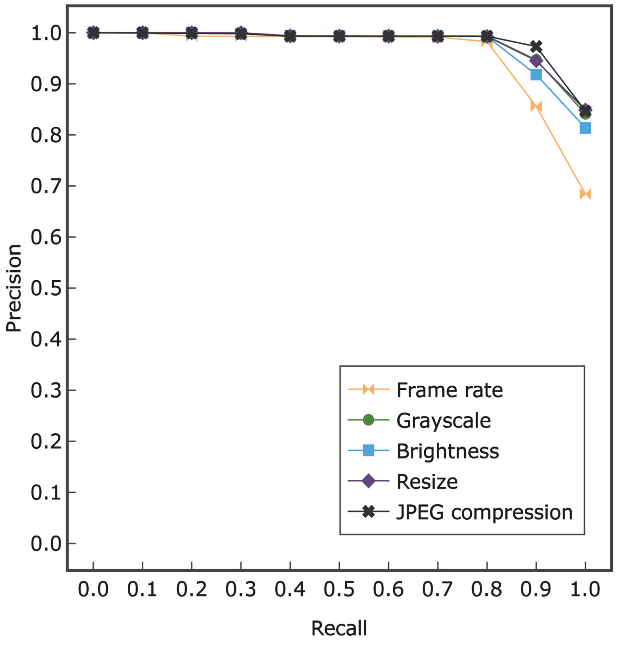


Figure 8. Precision-recall curves for similarity detection using the Frame Signature.

observations. The Frame Signature, standardized by the ISO/IEC, matches the frame descriptor to our content-based video retrieval system. The exemplar set extracted by our machine learning algorithms on each video can function as its numerical label. By attaching such numerical labels to each video, we can structure massive amounts of video data effectively. These labels have a wide range of applications. The detection of inserted plagiarized images in our experiments is one such application. The importance of the choice of descriptor was revealed by the experiments of Figure 7 and Figure 8. As stated in Section 1, we addressed the structuring of unstructured video data.

We have longstanding anticipation whose direction is opposite to the similar video retrieval. That is the machine generation of artificial videos from descriptors. In the case of the still image generation by the Generative Adversarial Nets [19], Gaussian random numbers were used as the source information. Because each video is a time series, independent and identical random numbers may not be appropriate for the source information. On the other hand, the exemplar set $\mathcal{E}$ of this paper is a data-compressed numerical time series. This set would work as the source for a video generation.

Acknowledgements

Mr. Masafumi Moriwaki receives special gratitude for his initial contribution to the invention of the M-distance together with the last author. Dr. Ryota Yokote, and Messrs. Shota Ninomiya, Akihiro Shikano, and Hiromichi Iwase receive thanks from the authors for their ideas and help developing the total system. This paper is a part of the outcome of research performed under a Waseda University Grant for Special Research Projects (Project number: 2016K-176). In addition, this work was supported in part by the Japan Society for the Promotion of Science through Grants-in-Aid for Scientific Research (C) (17K00135).

References

[1] Hu, W., Xie, N., Li, L., Zeng, X. and Maybank, S. (2011) A Survey on Visual Content-Based Video Indexing and Retrieval. *IEEE Transactions on Systems, Man, and Cybernetic*, **41**, 797-819. https://doi.org/10.1109/TSMCC.2011.2109710

[2] ISO/IEC 15938-3:2002/Amd.4:2010, Information Technology—Multimedia Content Description Interface—Part 3: Visual, AMENDMENT 4: Video Signature Tools.

[3] Levenshtein, V.I. (1966) Binary Codes of Correcting Deletions, Insertions, and Reversals. *Soviet Physics Doklady*, **10**, 707-710.

[4] Sakoe, H. and Chiba, S. (1978) Dynamic Programming Algorithm Optimization for Spoken Word Recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **26**, 43-49. https://doi.org/10.1109/TASSP.1978.1163055

[5] Needleman, S.B. and Wunsch, C.D. (1970) A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins. *Journal of Molecular Biology*, **48**, 443-453. https://doi.org/10.1016/0022-2836(70)90057-4

[6] Smith, T.F. and Waterman, M.S. (1981) Identification of Common Molecular Subsequences. *Journal of Molecular Biology*, **147**, 195-197. https://doi.org/10.1016/0022-2836(81)90087-5

[7] Messing, D.S., van Beek, P. and Errico, J.H. (2001) The MPEG-7 Colour Structure Descriptor: Image Description Using Colour and Local Spatial Information. *Proceedings* 2001 *International Conference on Image Processing*, Thessaloniki, Greece, **1**, 670-673. https://doi.org/10.1109/ICIP.2001.959134

[8] Paschalakis, S., Iwamoto, K., Brasnett, P., Sprlian, N., Oami, R., Nomura, T., Yamada, A. and Bober, M. (2012) The MPEG-7 Video Signature Tools for Content Identification. *IEEE Transactions on Circuits and Systems for Video Technology*, **22**, 1050-1063. https://doi.org/10.1109/TCSVT.2012.2189791

[9] ISO/IEC 15938-3:2002, Information Technology—Multimedia Content Description Interface—Part 3: Visual.

[10] Horie, T., Shikano, A., Iwase, H. and Matsuyama, Y. (2015) Learning Algorithms and Frame Signatures for Video Similarity Ranking. *Lecture Notes in Computer Science*, **9489**, 147-157. https://doi.org/10.1007/978-3-319-26532-2_17

[11] Frey, B.J. and Dueck, D. (2007) Clustering by Passing Messages between Data Points. *Science*, **315**, 972-976. https://doi.org/10.1126/science.1136800

[12] Horie, T., Moriwaki, M, Yokote, R., Ninomiya, S., Shikano, A. and Matsuyama, Y. (2014) Similar-Video Retrieval via Learned Exemplars and Time-Warped Alignment. *Lecture Notes in Computer Science*, **8836**, 85-94. https://doi.org/10.1007/978-3-319-12643-2_11

[13] Matsuyama, Y. (1996) Harmonic Competition: A Self-Organizing Multiple Criteria Optimization. *IEEE Transactions on Neural Networks*, **7**, 652-668. https://doi.org/10.1109/72.501723

[14] Matsuyama, Y., Shikano, A., Iwase, H. and Horie, T. (2015) Order-Aware Exemplars for Structuring Video Sets: Clustering, Aligned Matching and Retrieval by Similarity. *Proceedings of the International Joint Conference on Neural Networks*, Killarney, Ireland, 1-10. https://doi.org/10.1109/IJCNN.2015.7280423

[15] Equitz, W.H. (1989) A New Vector Quantization Clustering Algorithm. *IEEE Transactions on Acoustics*, *Speech*, *and Signal Processing*, **37**, 1568-1575. https://doi.org/10.1109/29.35395

[16] Ward, J.H. (1963) Hierarchical Grouping to Optimize an Objective Function. *Journal of the American Statistical Association*, **58**, 236-244. https://doi.org/10.1080/01621459.1963.10500845

[17] Yahoo! Webscope (2014) Yahoo! Webscope dataset YFCC-100M. http://labs.yahoo.com/Academic_Relations

[18] Thomee, B., Shamma, D.A., Friedland, G., Elizalde, B., Ni, K., Poland, D., Borth, D. and Li, L. (2016) YFCC100M: The New Data in Multimedia Research. *Communications of the ACM*, **59**, 64-73. https://doi.org/10.1145/2812802

[19] Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. (2014) Generative Adversarial Nets. *Proceedings of the 27th International Conference on Neural Information Processing Systems*, Montreal, Canada, **2**, 2672-2680.