THE UNIVERSITY OF TOKYO

MASTER THESIS

Research on Advanced Temporal Matching Kernel toward High Performance Video Event Retrieval

映像イベント検索の高精度化のための時間的照合カーネルの改良に関する研究

Author:
Fan YANG

Supervisor:
Prof. Shin'ichi SATOH

A thesis submitted in fulfillment of the requirements for the degree of Master of Engineering

in the

Information and Communication Engineering
Graduate School of Information Science and Technology

Student ID: 48-166464

February 5, 2018

THE UNIVERSITY OF TOKYO

Abstract

Information and Communication Engineering
Graduate School of Information Science and Technology

Master of Engineering

Research on Advanced Temporal Matching Kernel toward High Performance Video Event Retrieval

by Fan YANG

This research is addressing the problem of content-based video retrieval in a large video database. Most of previous approaches for solving this task are based on bag of features. However, they are incapable of localizing the content of interest within retrieved videos, while temporal information embedded methods are competent on it. The frustrating fact is that the latter group of methods showed inferior retrieval performance. In this paper, we propose novel methods to improve the unsatisfactory performance of temporal matching kernel.

The proposed methods are aiming at alleviating the affects caused by irrelevant yet similar frames in a video comparison. In detail, we propose an embedded stability-sensitive filter for penalizing the contribution made by similar frames scattered in irrelevant contexts. Moreover, we propose a burst-survive temporal matching kernel for the same purpose but with less computational cost. Furthermore, in order to obtain better localization performance, we propose a multi-period method based on Fibonacci numbers.

Acknowledgements

First of all, I would like to express my sincere gratitude to my supervisor Dr. Shin'ichi Satoh for his meticulous guidance and genial attitude to all members in the laboratory. This research will hardly ripen without his invaluable comments and suggestions. I deeply appreciate the wonderful opportunities granted by him to participate several academic conferences.

I am very much thankful to the senior members, Dr. Yusuke Matsui and Mr. Ryota Hinami. They selflessly shared many valuable experiences both in life and study. Their passion and achievements always motivate me to be better than current me. I am also grateful to the previous members Mr. Shunsuke Tsukatani, Mr. Sébastien Pollout and Mr. Junfu Pu. Without your work and contribution, I may miss this interesting research topic and would not obtain so much inspiration.

Thanks also to my classmate Mr. Norihito Yanai for staying up late together in the laboratory. And of course, thank so much to our secretary Mrs. Yumi Awano for the support and seriously, the reimbursement. I acknowledge every other member for the harmonious ambience in our Lab.

Finally, I would like to say thank you to my family. Thank you for your trust and support all the way.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Recent years, along with the exponential growth of video sources, the auxiliary analytic tools have been widely developed and studied. Content-based video retrieval (CBVR), as a genre among the video analyses, is a specific task to search for semantically similar videos in a large video database, given a query video. As reported in [13, 21], CBVR addresses various tasks such as particular event analysis, copy detection and video surveillance, etc. The broad range of applications motivates the interests of researchers worldwide. The annual Text REtrieval Conference VIDeo retrieval evaluation (TRECVID), sponsored by the National Institute of Standards and Technology, provides a large-scale test collection of videos. The fierce competition in TRECVID's CBVR tasks shows not only the researchers' enthusiasm but also the significance of CBVR itself.

Most of existing CBVR methods are based on bag of features (BoF) technique. Since the redundant information are readily constricted by clustering, those BoF based methods are extolled by their remarkable retrieval quality. However, other than the retrieval results depicting which video clips contain the query, or the content of interest, sometimes we still want the information where the query appears in retrieved videos.

Unfortunately, to localize the query is a task of high computational cost. Traditionally, this task is solved by using classic Hough voting scheme [23, 8], whose computational complexity is quadratic to the length of the video. The temporal information embedded methods were proposed to tackle the computational issue. By embedding temporal information together with frame descriptors into the video descriptors, the relative temporal offsets or displacements between videos become by-products in CBVR with an acceptable computational complexity.

However, the retrieval performance of temporal information embedded methods are generally inferior to BoF based ones. Worse still, since temporal information are embedded for a better computational performance, any modification on the method should do no harm to the original embedding framework.

1.2 Research Purpose

Consider if you are handling a CBVR task by naked eyes, you may watch the query video first, and remember the scenes in your mind. Next, you watch the videos in database for comparison, your decision is possibly made by such a scheme: instead of focusing on a single frame, you actually compare the context of the current frame with contents of the query, if the context is relevant, you may check the following frames and find if there is a continuously similar pattern; if the context is irrelevant, you ignore the current frame in this comparison. Such an intuitive method is noteworthy because it depicts a critical principle in the video retrieval: the similarity is determined by the context rather than the frames one by one. For example, there are many video clips you took for your girlfriend in the smart phone. These videos are related to different events, *e.g.* eating in a restaurant, playing in the park, etc. However, your girlfriend shows up in every video. So if you compare the videos by each frame, the contribution of your girlfriend's face will be predominant and makes all videos similar. Instead, by taking the context, several neighborhood frames, of each frame into consideration, we are able to rule out some irrelevant yet similar frames. For instance, your girlfriend's face showing in an indoor scene are considered irrelevant to the same face but showing in an outdoor scene. In real video retrieval cases, there also are many similar contents, *e.g.* human bodies, buildings, cars, trees and so on, which may also cause irrelevant yet similar frames. Since such irrelevant frames always result in misleading contribution to the video-wise similarity, we call the frame-wise similarities between those irrelevant frames as **noise**.

Unfortunately, it is not easy to alleviate the influence of noise. To enable the offset localization, a video descriptor must contain information of frames at all positions. It means that all frames are evenly weighted, and none of them are allowed to be dropped during the construction of video descriptors. Thus, in the searching stage, even some frames are irrelevant to the query in its context, we can hardly drop them since they are already embedded into the video descriptors. The contribution of these frames is still collected, and serves like noise.

This research is aiming at improving the retrieval performance by eliminating the noise caused by irrelevant frames. The relatively new temporal matching kernel [21] is chosen as the baseline. Even though our main target is to improve the retrieval and localization performance, since we are dealing with a retrieval task, we still have to control the computational complexity on a practical level.

In this paper, we propose two generic methods to circumvent the "noise" problem: the stability-sensitive filter and burst-survive temporal matching kernel. Furthermore, we propose a brand-new multi-period method for enhancing the localization performance. We show that combining the burst-survive temporal matching kernel and the new multi-period method generates significant improvement on the retrieval performance.

1.3 Paper Organization

- Chapter 1:
  Introduction to the thesis topic. The purpose is briefly described.
- Chapter 2:
  We introduce the related works, some of them are prerequisites of the following discussion. TE on which our proposed methods based is demonstrated in detail.
- Chapter 3:
  A stability-sensitive filter is proposed for better retrieval performance, including an approximation for reducing computational cost.
- Chapter 4:
  A novel method called BURST as well as its theory is described in this chapter. The difference between the asymmetric and symmetric schemes is explained.
- Chapter 5:
  We visualize the multi-period strategy for ease of understanding. A method for choosing periods based on the golden ratio is proposed.
- Chapter 6:
  We show detailed settings of our experiments and the experimental results in this chapter. Some analyses to the results can be found here.
- Chapter 7:
  The conclusion and future works are described.

Chapter 2

Related Work

2.1 Content-based Video Retrieval

The problem we are addressing is the Content-Based Video Retrieval (CBVR). It is abstracted from a practical application that we want to search videos by videos, which is conceptually similar to the Content-Based Image Retrieval (CBIR). Supposing that you obtained a video clip in which a lovely girl shows up, you want more but you have no extra information about her. Then you may need to conduct CBVR. CBVR is a task of searching relevant videos according to the similarity of contents, given a query video. The most relevant videos are expected to have contents similar to the query. Now that we retrieved relevant videos, we still wonder if our searching system can directly locate the content of interest in the retrieved videos. Hopefully our system is able to tell us that the contents in query appear in a retrieved video from a certain time. For instance, a commercial is set as a query to be searched in a large TV database. The final results are expected to provide the videos that contains the commercial, and optionally locate where the commercial is, *i.e.* the relative offset.

However, to implement such a system is not easy. Since videos contain rich information, *e.g.* subtitles, audio, and visual content, which can be used for retrieval, researchers made various attempts. Some of them concerned the textual and speech information inside videos. Such kind of video retrieval systems extract textual data by using Optical Character Recognition (OCR) on keyframes or automatic speech recognition (ASR) on audio [5, 32]. Other researchers insist in constructing visual descriptors from visual contents. At the very beginning, they tried to extract low level image-based information, such as color, texture, and shape, to construct global frame descriptors [1, 24]. In later researches, local descriptors, particularly SIFT and SURF are used in many Bag of Features (BoF) methods [26, 27, 7]. Video Google [26] is frequently cited as a predecessor to build a visual vocabulary by using descriptors extracted from frames. Some researchers took advantage of videos' hierarchical structures like chapters, scenes, and shots, whereby redundancies can be removed for producing compact video descriptors [8, 29, 16]. More recently, the BoF family has been developed rapidly, most contributed by the development of image descriptors. Jégou *et al.* proposed compact frame descriptors, including VLAD and MVLAD, that achieve good performance in retrieval [14, 15]. The modern Convolutional Neural Networks (CNN)

provide better image descriptors than those hand-made previously, and also greatly facilitate the progress of CBVR.

The methods in the BoF family are superior in creating compact video descriptors and have good retrieval performance. However, they are incapable of localization. Instead, an additional temporal matching strategy such as local alignment [33] or voting techniques[1] are required to ensure temporal consistency. Since the computational complexity of such a temporal matching strategy is quadratic in the length of video, localizing the relative offsets afterwards is extremely time consuming. A better choice for localization is such kind of methods who are initially designed with ability of localization. We describe these methods in section 2.3.

2.2 Image Feature Extraction

In a content-based video retrieval task, the target is to search for videos sharing similar contents. Focusing merely on the visual contents, the first requirement is a measurement on similarity according to contents. As we know, a video is basically a collection of images, or frames. To measure the similarity between videos' contents, one prerequisite is to measure the similarity between frames. More particularly, for each frame, the features are extracted and compared.

In addition, it should be mentioned that since videos have rich content, the data amount is quickly a burden. To perform video analyses on a practical level, compact frame descriptors are usually required [13]. Traditional image descriptors are hand-made by local descriptors such as SIFT [18] or SURF [3]. The Fisher vector [20] and its variants VLAD [2] and Multi-VLAD [2, 14], designed to be both distinctive and robust, are common choices. Recent years, the development of deep convolutional neural networks (DCNN) lead to another kind of promising visual descriptors for images. Those image descriptors based on the activations within DCNN have been proved to yield state-of-the-art performance in visual recognition [11, 22, 34].

In following sections, we describe some of the typical image descriptors.

2.2.1 VLAD and Multi-VLAD

Jégou *et al.* introduced VLAD for large scale object instance retrieval, given a query image [2]. Starting from local descriptors such as SIFT, they apply a vector quantization for clustering, and then record the difference from the cluster center. As it is derived from SIFT, VLAD is also invariant to in-plane rotation and somewhat tolerant to other transformations. Although VLAD is reported to perform better than conventional Bag of Words (BoW) descriptor, it still suffers from high memory occupation, especially for very large image datasets.

[1] For example, the Hough transform or RANSAC.

[2] Abbreviation for "Vector of Locally Aggregated Descriptors".

Multi-VLAD (MVLAD) is an advanced version of VLAD, which is literally multiple VLADs but aggregated through a Gaussian Mixture Model (GMM). Moreover, whitening and dimension reduction are applied afterwards. Previous works show that the retrieval performance is improved for small objects by using MVLADs [14].

We employ MVLADs in some experiments for evaluating our proposed methods against other state-of-the-arts where MVLADs are used.

2.2.2 DCNN Features

Deep Convolutional Neural Networks (DCNN) have shown their proficiency in a number of tasks such as image classification. This inspires an application where the CNN features are extracted as a universal representation.

Previous researches reported that features extracted from DCNN layers showed a significant increase in performance over traditional approaches [6, 19]. The recent work [10] also proved the superiority of DCNN features especially those extracted from the ResNet [12] outperform MVLADs in a video retrieval task. Similar to these applications, we also apply DCNN features in our experiments.

2.3 Video Retrieval Methods

In this section, we briefly introduce the existing video retrieval methods along with their pros and cons. These methods can be roughly put into three categories: 1) per-frame matching methods [8], 2) Bag of Features/Frames (BoF) methods [9, 10], 3) temporal encoding/embedding methods [23, 21].

Although some retrieval technique such as inverted index are used for acceleration, the intrinsic computational complexity limits the use of per-frame matching methods for large datasets. While these methods are useful when precise, on frame level exactly, localization is demanded. BoF methods come to another extremity. They focus on extracting unique features from videos by using certain clustering approaches, *e.g.* k-means, Sign of Stable Components (SSC) [9], counting grid [10], etc. These features extracted for each video are then merged together for generating a extremely compact video representation with little redundancy. Since videos are represented by compact descriptors, the computation can be very fast, and the retrieval performance is good as well. However, because all temporal information are lost during the irreversible pooling, BoF methods are incapable of localizing the relative offset.

The choice of temporal encoding/embedding methods is a plausible compromise between the above two extremities. Taking advantage of the frequency domain, these methods are not only capable in video retrieval, but also in temporal localization. As the word "compromise" literally denotes, the computation cost of temporal encoding/embedding methods is between that of other two kinds of methods, but is feasible enough though. The problematical issue lies in the performance. Although previous works [23, 21]

showed state-of-the-art performance, there is still a plenty of room for improvement with respect to BoF methods.

Both Circulant Temporal Encoding (CTE) and Temporal matching kernel with Explicit feature maps (TE) typify such methods in the third category. However, they are different in several aspects:

- CTE uses FFT on temporal sequences while TE uses Fourier series instead. Consequently, CTE requires zero padding while TE does not. Meanwhile, the embedded space of CTE is a complex space due to properties of FFT, which makes the computation more complicated.
- In CTE, the comparison function for timestamps is fixed as a Dirac delta, which can be customized in TE.
- A filter is applied in CTE for reducing self-similarities, which greatly improved the performance of CTE. While there is no counterpart for TE.

For simplicity and expandability, we choose TE as our baseline.

2.3.1 Temporal Matching Kernel with Explicit Feature Maps

As mentioned in Section 2.2, CBVR requires a feature extraction to videos at the frame level. Once features of frames are extracted, a video is represented as a sequence of feature vectors known as frame descriptors. Consider a comparison between two videos, whose frame sequences are denoted as $\mathbf{x} = [\boldsymbol{x}_0, \dots, \boldsymbol{x}_t, \dots]$ and $\mathbf{y} = [\boldsymbol{y}_0, \dots, \boldsymbol{y}_t, \dots]$, where $\boldsymbol{x}_t, \boldsymbol{y}_t \in \mathbb{R}^d$ are d-dimensional frame descriptors. Under the constraint that frame descriptors are L2 normalized, the similarity between two frames is evaluated by inner product of their corresponding descriptors. Moreover, following the assumption proposed in [23], the sum of similarities between the frame descriptors reflects the similarity of the sequences or videos. The metric is denoted as

$$\mathcal{K}_\Delta(\mathbf{x}, \mathbf{y}) = \sum_{t=0}^{\infty} s_t^\Delta = \sum_{t=0}^{\infty} \langle \boldsymbol{x}_t, \boldsymbol{y}_{t+\Delta} \rangle, \tag{2.1}$$

where $\langle \cdot, \cdot \rangle$ stands for the operator of inner product, s_t^Δ represents the similarity between t-th frame in video $\mathbf{x}$ and $(t+\Delta)$-th frame in video $\mathbf{y}$.

In addition, [23] also pointed out that this assumption is not well satisfied in practice, mainly due to the self-similarity[3] of videos themselves. From our perspective, we argue the assumption itself is not robust to the noise as discussed in section 1.2.

TE is basically an algorithm to accelerate the computation of Eq. 2.1 by using explicit feature map proposed in [31]. To achieve its purpose, Eq. 2.1 is

[3]The property that videos are self-similar in time.

deformed as

$$\mathcal{K}_\Delta(\mathbf{x},\mathbf{y}) \propto \sum_{t=0}^{\infty}\sum_{t'=0}^{\infty} \boldsymbol{x}_t^{\mathrm{T}}\boldsymbol{y}_{t'}k(t,t'+\Delta) = \underbrace{\left(\sum_{t=0}^{\infty}\boldsymbol{x}_t\otimes\boldsymbol{\varphi}(t)\right)^{\mathrm{T}}}_{\boldsymbol{\psi}_0(\boldsymbol{x})}\underbrace{\left(\sum_{t'=0}^{\infty}\boldsymbol{y}_{t'}\otimes\boldsymbol{\varphi}(t'+\Delta)\right)}_{\boldsymbol{\psi}_\Delta(\boldsymbol{y})}, \tag{2.2}$$

where $\otimes$ denotes the operator of Kronecker product, $k(t,t')$ is a temporal similarity kernel approximated by using explicit feature map such that

$$k(t,t') \approx \varphi(t)^{\mathrm{T}}\varphi(t') = \begin{bmatrix} \sqrt{a_0} \\ \sqrt{a_1}\cos(\frac{2\pi}{T}t) \\ \sqrt{a_1}\sin(\frac{2\pi}{T}t) \\ \vdots \\ \sqrt{a_m}\cos(\frac{2\pi}{T}mt) \\ \sqrt{a_m}\sin(\frac{2\pi}{T}mt) \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \sqrt{a_0} \\ \sqrt{a_1}\cos(\frac{2\pi}{T}t') \\ \sqrt{a_1}\sin(\frac{2\pi}{T}t') \\ \vdots \\ \sqrt{a_m}\cos(\frac{2\pi}{T}mt') \\ \sqrt{a_m}\sin(\frac{2\pi}{T}mt') \end{bmatrix}, \tag{2.3}$$

where $\{a_i\}$ are the coefficients of Fourier series[4], m is the number of frequencies we take, and T is the period.

As some of you might be aware now, the similarity computation between videos is handed over to independent video descriptors $\{\boldsymbol{\psi}_\Delta(\cdot)\}$. A more attractive fact is that for all shifts $\{\Delta\}$, the inner product of video descriptors is merely conducted once.

Taking $\boldsymbol{\psi}_0(\mathbf{x})$ as an example, we obtain

$$\boldsymbol{\psi}_0(\mathbf{x}) = [\mathbf{D}^{\mathrm{T}}, \mathbf{C}_1^{\mathrm{T}}, \mathbf{S}_1^{\mathrm{T}}, \ldots, \mathbf{C}_m^{\mathrm{T}}, \mathbf{S}_m^{\mathrm{T}}]^{\mathrm{T}}, \tag{2.4}$$

where

$$\begin{aligned} \mathbf{D} &= \sqrt{a_0}\sum_{t=0}^{\infty}\boldsymbol{x}_t, \\ \mathbf{C}_i &= \sqrt{a_i}\sum_{t=0}^{\infty}\boldsymbol{x}_t\cos\left(\frac{2\pi}{T}it\right), \\ \mathbf{S}_i &= \sqrt{a_i}\sum_{t=0}^{\infty}\boldsymbol{x}_t\sin\left(\frac{2\pi}{T}it\right) \end{aligned} \tag{2.5}$$

are referred to as the Direct Current (DC) component, cosine components, and sine components respectively. The latter two types are generalized as Alternating Current (AC) components.

By performing trigonometric transformations, it could be quickly derived that

$$\begin{aligned} \mathcal{K}_\Delta(\mathbf{x},\mathbf{y}) \propto\ & \langle \mathbf{D}^{(\mathbf{x})}, \mathbf{D}^{(\mathbf{y})} \rangle \\ & + \sum_{i=1}^{m}\cos\left(\frac{2\pi}{T}i\Delta\right)\left(\langle \mathbf{C}_i^{(\mathbf{x})}, \mathbf{C}_i^{(\mathbf{y})} \rangle + \langle \mathbf{S}_i^{(\mathbf{x})}, \mathbf{S}_i^{(\mathbf{y})} \rangle\right) \\ & + \sum_{i=1}^{m}\sin\left(\frac{2\pi}{T}i\Delta\right)\left(\langle \mathbf{C}_i^{(\mathbf{x})}, \mathbf{S}_i^{(\mathbf{y})} \rangle - \langle \mathbf{S}_i^{(\mathbf{x})}, \mathbf{C}_i^{(\mathbf{y})} \rangle\right). \end{aligned} \tag{2.6}$$

[4]For instance, if we set $k(t,t')$ as a Dirac delta function, $a_0 = a_1 = \cdots = 1$.

It is clear to see that DC components only provide a baseline, on which AC components draw the fluctuation with regard to different shifts $\{\Delta\}$.

In video searching stage, we are going to return the most relevant videos to a query from a large database, together with their relative time offsets. Let the query video be $\mathbf{q}$ and the video collection of database be $\{\mathbf{d}_i\}$. For each video in database, the similarity score is measured as

$$S_{\mathbf{q},\mathbf{d}_i} = \max_{\Delta} \mathcal{K}_\Delta(\mathbf{q}, \mathbf{d}_i), \tag{2.7}$$

while the relative time offset $\Delta_{\mathbf{q},\mathbf{d_i}}$ between videos $\mathbf{q}$ and $\mathbf{d}_i$ satisfies

$$\Delta_{\mathbf{q},\mathbf{d}_i} = \arg\max_{\Delta} \mathcal{K}_\Delta(\mathbf{q}, \mathbf{d}_i). \tag{2.8}$$

Once the video-wise similarity scores are obtained, we only have to sort them in descending order. The top ranked ones are best-matching videos.

2.3.2 Multiple Periods

As shown in Eq. 2.3, a period T is directly related to the explicitly expanded temporal kernel. One crucial criterion of choosing the period T is that such a T should guarantee the uniqueness of the time offset $\Delta_{\mathbf{q},\mathbf{d}_i}$. As we know, $\Delta_{\mathbf{q},\mathbf{d}_i}$ can be any value that shows the number of shifted frames between compared videos, *i.e.* $\Delta_{\mathbf{q},\mathbf{d}_i} \in [-\mathbf{q}.\text{length} + 1, \mathbf{d}_i.\text{length}]$. Moreover, since $\varphi(t) = \varphi(t + T)$, for a period T, $\mathcal{K}_\Delta(\mathbf{q}, \mathbf{d}_i)$ is evaluated over a period $\Delta \in [0, T)$. In one word, the period T should ensure that $[0, T)$ is not narrow than $[-\mathbf{q}.\text{length} + 1, \mathbf{d}_i.\text{length}]$. However, [21] controversially chooses T to be larger than the number of frames of the longest video in database for simplicity.

Unfortunately, the larger T is, the lower the resolution [5] $\frac{2\pi}{T}$ will be, which may lead to an inaccurate offset, or even miss it. In addition, a long period requires higher frequencies in Fourier series for offset localization, and thus need more dimensions in timestamps $\{\varphi(t)\}$ for keeping the resolution on an endurable level, which inevitably lead to lengthy video descriptors. After realizing the problem incurred by a large T, Poullot *et al.* [21] proposed an approach to improve the localization accuracy while keeping the video representation in a reasonable size. Their idea is to use multiple short periods instead of a single long period to modulate video descriptors.

Consider a set of short periods $\{T_j\}$. For each period T_j, the potential candidates of real time offset between a query video $\mathbf{q}$ and a video $\mathbf{d}_i$ in the database are $\{\Delta^{T_j}_{\mathbf{q},\mathbf{d}_i} + k_j T_j\}$, where k_j is an integer. To disambiguate the offset, they set these multiple periods as prime numbers. And choose the most possible offset amongst all potentials. However, we argue that prime periods are not suitable in practice. The details are introduced in Chapter 5.

[5] We call it resolution because the smaller it is, there will be less discrepancy between timestamps $\varphi(t)$ and $\varphi(t + \triangle t)$ where $\triangle t$ is a constant standing for an increment on t.

2.4 Query Expansion

Query expansion is a re-ranking method for event retrieval. There are two widely used methods presented by Douze *et al.* [9]: Average Query Expansion (AQE) and Difference of Neighborhood (DoN). DoN is reported to perform better than AQE both theoretically and practically.

More importantly, since TE provides relative time offset between a query and each video in database, TE has a typical functionality to check temporal consistency among the query and videos in shortlist of retrieved videos[6]. Consistency check is very effective to select relevant videos from the shortlist for query expansion. This technique is similar to geometric consistency check for object retrieval from images [4]. Consider a query video $\mathbf{q}$ and two database videos $\mathbf{d}_1, \mathbf{d}_2$. If they are perfectly consistent, their time offsets should satisfy $\Delta_{\mathbf{q},\mathbf{d}_1} + \Delta_{\mathbf{d}_1,\mathbf{d}_2} = \Delta_{\mathbf{q},\mathbf{d}_2}$. A small temporal tolerance ε is added for robustness; the constraint is then loosened to

$$|\Delta_{\mathbf{q},\mathbf{d}_1} + \Delta_{\mathbf{d}_1,\mathbf{d}_2} - \Delta_{\mathbf{q},\mathbf{d}_2}| \leq \varepsilon, \tag{2.9}$$

where ε is in units of frame. We set its value according to the frame rate of the videos. In our experiments, we found 10 for 5 fps and 50 for 15 fps are reasonable choices.

[6] A small collection of top-ranked retrieved videos corresponding to a query.

Chapter 3

Temporal Matching Kernel with Embedded Filter

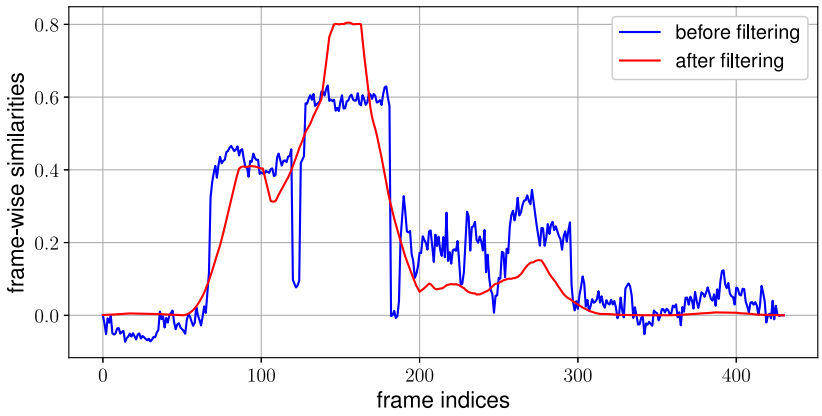
Before introducing the details of proposed methods, we briefly crystallize the reason why TE or other temporal information embedded methods perform unsatisfactorily.

As a method that is capable of localization, the video descriptors of TE is constructed by all frames in the videos. In a comparison between two videos, those irrelevant yet sometimes accidentally similar frames may result in noisy frame-wise similarities, *i.e.* frames without temporal continuity are in correspondence instead. Usually matching videos have consecutive frames in correspondence, the above situation is by no means what we expect. We argue that existing temporal information embedded methods are not robust to the noise caused by irrelevant frames.

As discussed in section 1.2, instead of a single frame, it is more reasonable to judge relevance by the context, the ambient frames of current one. Since a pair of irrelevant yet similar frames lead to a isolated high frame-wise similarity while the contexts around them are not relevant. Those frames result in non-consecutive high frame-wise similarities which are in a shape of white noise. This naturally give us the hint to apply a low-pass filter on the frame-wise similarities. A low-pass filter is expected to be effective since it smooths the noise with high frequencies. Following this intuition, we propose a stability-sensitive filter, to the frame-wise similarities. We further develop a technique to embed our filter into frame descriptors without deforming TE's well-designed framework.

We take two examples below for better understanding to our purpose. Fig. 3.1 shows an example of true match case where consecutive similar contents exist in two relevant videos. The similar contents between pre-aligned videos result in a section of stable high frame-wise similarities in Fig. 3.1 (B) (blue line), which are usually expected. On the other hand, the remaining contents do not show any continuously similar pattern, but still contain some similar features, such as human bodies, buildings, etc. As plotted in Fig. 3.1 (B), the frames in irrelevant contexts (different scenes) causes lots of noisy frame-wise similarities, where some of them are non-consecutive high values. Since TE defines the video-wise similarity as a sum of frame-wise similarities (Eq. 2.1), it is incapable of distinguishing if the contribution is from consecutive similarities or non-consecutive ones. Our proposed

(A) Frame sequences of two videos in a true match case.



(B) Frame-wise similarities in a true match case before and after filtering. The scales are adjusted for comparison.

FIGURE 3.1: A true match case: videos are relevant due to consecutive similar contents.

stability-sensitive filter is applied for sieving noise from frame-wise similarities. Fig. 3.1 (B) (red line) shows the effectiveness of our filter: stable sections are emphasized while noise is panelized.

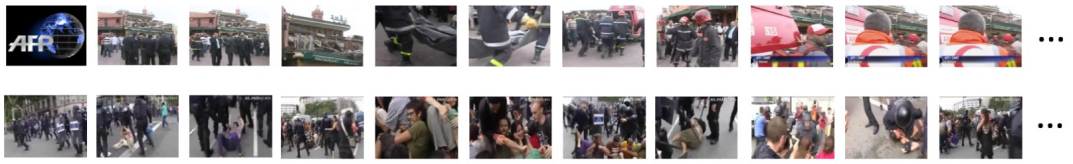
For further interpretation, a false match case is illustrated in Fig. 3.2. The frame sequences in Fig. 3.2 (A) are extracted from videos belonging two irrelevant events. However, they are judged to be relevant[1] because frame-wise similarities displayed in Fig. 3.2 (B) (blue line) are summed up to a falsely high similarity score. These frame-wise similarities, apparently, are noisy ones what we never expected. Fortunately, by applying our filter, the contribution from noise is almost eliminated as Fig. 3.2 (red line) shows.

3.1 Filter Analysis

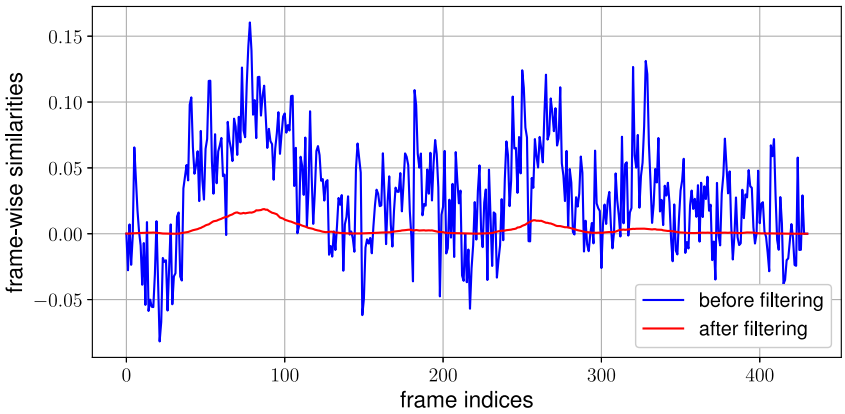
Regarding frame-wise similarities as a temporal sequence, a reasonable choice to penalize noise would be to conduct convolution with a linear filter, e.g. a Gaussian window. However, since frame-wise similarities are aggregated as a video-wise similarity score afterwards following Eq. 2.1, whatever linear filter is used the aggregated score will remain unchanged.

For ease of exposition, we denote frame-wise similarities as a temporal sequence $[s_0^\Delta, \cdots, s_t^\Delta, \cdots]$, where s has the same definition with that in Eq. 2.1.

[1] Videos in database are sorted by their similarity scores corresponding to the query, top ranked ones are inferred as relevant videos.



(A) Frame sequences of two videos in a false match case.



(B) Frame-wise similarities in a false match case before and after filtering. The scales are adjusted for comparison.

FIGURE 3.2: A false match case: videos are irrelevant while the video-wise similarity score is falsely high.

By introducing a linear filter $\{h_\cdot\}$, whose width is $w = 2l + 1$, we obtain

$$\hat{\mathcal{K}}_\Delta(\mathbf{x}, \mathbf{y}) = \sum_{t=0}^{\infty} \sum_{k=-l}^{l} h_k s_{t+k}^\Delta \approx \sum_{k=-l}^{l} h_k \sum_{t=0}^{\infty} s_t^\Delta \propto \mathcal{K}_\Delta(\mathbf{x}, \mathbf{y}) = \sum_{t=0}^{\infty} s_t^\Delta. \tag{3.1}$$

Thus, linear filters are ineffective due to the existence of a sum function over frame-wise similarities. Accordingly, we decide to switch to non-linear filters. We choose a higher-order monomial kernel as the filter instead of other commonly used non-linear filters. The reason is further demonstrated in section 3.3. Intuitively, we combine a linear filter and higher-order monomial kernel $(\cdot)^p, p > 1$ to compose the stability-sensitive filter as bellow.

$$\tilde{\mathcal{K}}_\Delta(\mathbf{x}, \mathbf{y}) \propto \sum_{t=0}^{\infty} \left(\sum_{k=-l}^{l} h_k s_{t+k}^\Delta \right)^p, \tag{3.2}$$

where p is the exponent of filter. The settings in Fig. 3.1 (B) and Fig. 3.2 (B) are $p = 2, w = 37$.

Note that the filter is applied on frame-wise similarities between pre-aligned frame sequences, which may need a two-step operation: 1) obtain relative temporal offsets for alignment by using current TE, 2) apply filter on frame-wise similarities measured between aligned videos. However, we still prefer to embed the filter into TE in order to complete the computation

of video-wise similarity in a single step. We show the pipeline of embedding the filter into TE in Fig. 3.3, our filter works on frame-wise similarities (dashed arrow) in a embedded way (solid arrow).

As shown in Eq. 2.2, TE measure the video-wise similarity by computing inner product between two independent video descriptors. This arouses a problem that how can we embed a non-linear filter into a linear computation. In next section, we formulate the technique to embed our filter into TE.

3.2 Stability-Sensitive Filter

Inspired by the idea of explicit feature map, we explicitly expand our stability-sensitive filter into an embedded space. Similar to the definitions in section 2.3.1, we show an example for a comparison between videos $\mathbf{x}$ and $\mathbf{y}$. The frame-wise similarities when there is a shift Δ between $\mathbf{x}, \mathbf{y}$ are denoted as a temporal sequences $[s_0^\Delta, \cdots, s_t^\Delta, \cdots]$. The stability-sensitive filter is applied in form of

$$\begin{aligned}\tilde{s}_t^\Delta &= \left(\sum_{k=-l}^{l} h_k s_{t+k}^\Delta\right)^p = \left(\sum_{k=-l}^{l} \langle \sqrt{h_k}\boldsymbol{x}_{t+k}, \sqrt{h_k}\boldsymbol{y}_{t+\Delta+k}\rangle\right)^p \\ &= \left(\begin{bmatrix} \sqrt{h_{-l}}(\boldsymbol{x}_{t-l}) \\ \vdots \\ \sqrt{h_l}(\boldsymbol{x}_{t+l}) \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \sqrt{h_{-l}}(\boldsymbol{y}_{t+\Delta-l}) \\ \vdots \\ \sqrt{h_l}(\boldsymbol{y}_{t+\Delta+l}) \end{bmatrix}\right)^p = \left(\langle \mathbf{U}_t, \mathbf{V}_{t+\Delta}\rangle\right)^p,\end{aligned} \tag{3.3}$$

where

$$\begin{aligned}\mathbf{U}_t &= \left[\sqrt{h_{-l}}(\boldsymbol{x}_{t-l})^{\mathrm{T}} \cdots \sqrt{h_0}(\boldsymbol{x}_t)^{\mathrm{T}} \cdots \sqrt{h_l}(\boldsymbol{x}_{t+l})^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{R}^{wd}, \\ \mathbf{V}_{t'} &= \left[\sqrt{h_{-l}}(\boldsymbol{y}_{t'-l})^{\mathrm{T}} \cdots \sqrt{h_0}(\boldsymbol{y}_{t'})^{\mathrm{T}} \cdots \sqrt{h_l}(\boldsymbol{y}_{t'+l})^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{R}^{wd}.\end{aligned} \tag{3.4}$$

For embedding our filter into linear TE, we represent $\left(\langle \mathbf{U}_t, \mathbf{V}_{t+\Delta}\rangle\right)^p$ in a linear form, i.e. $\boldsymbol{\phi}_p(\mathbf{U}_t)^{\mathrm{T}}\boldsymbol{\phi}_p(\mathbf{V}_{t+\Delta})$. [30] has showed the technique for expanding a higher-order monomial kernel explicitly. Following that, when $p = 2$, $\tilde{s}_t^\Delta$ is expanded as

$$\begin{aligned}\tilde{s}_t^\Delta &= \left(\langle \mathbf{U}_t, \mathbf{V}_{t+\Delta}\rangle\right)^2 = \left(\sum_{i=1}^{wd} u_i v_i\right)\left(\sum_{j=1}^{wd} u_j v_j\right) \\ &= \sum_{i=1}^{wd}\sum_{j=1}^{wd}(u_i u_j)(v_i v_j) = \boldsymbol{\phi}_2(\mathbf{U}_t)^{\mathrm{T}}\boldsymbol{\phi}_2(\mathbf{V}_{t+\Delta}),\end{aligned} \tag{3.5}$$

where $\{u_\cdot\}$ and $\{v_\cdot\}$ correspond to elements in $\mathbf{U}_t$ and $\mathbf{V}_{t+\Delta}$ respectively. Specifically, $\boldsymbol{\phi}_2(\mathbf{U}_t)$ refers to

$$\boldsymbol{\phi}_2(\mathbf{U}_t) = \left[u_1u_1, \cdots, u_{wd}u_{wd}, \sqrt{2}u_1u_2, \sqrt{2}u_1u_3, \cdots, \sqrt{2}u_{wd-1}u_{wd}\right]^{\mathrm{T}}. \tag{3.6}$$

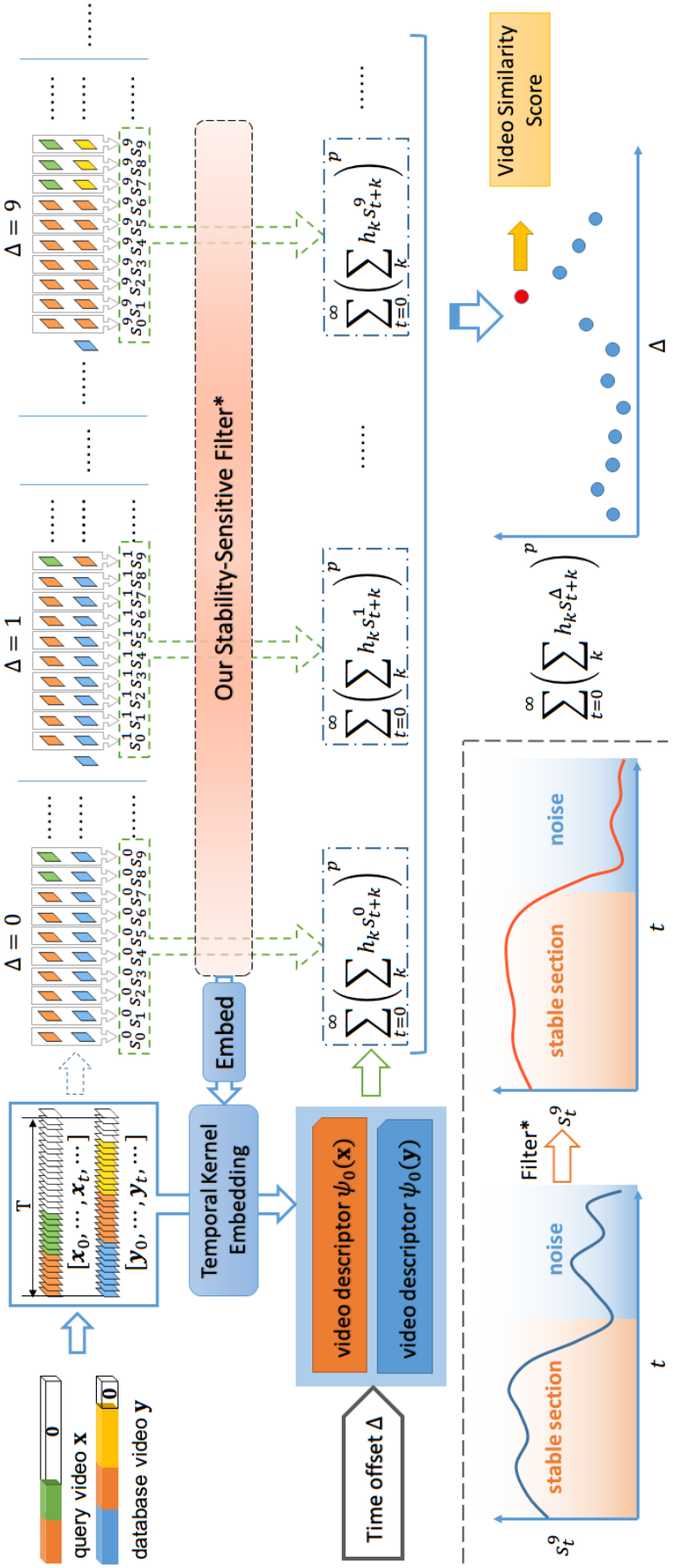


FIGURE 3.3: Overview of TE with embedded stability-sensitive filter.

In the same way, for any value of p, $\mathbf{U}_t$ can be expanded into $\boldsymbol{\phi}_p(\mathbf{U}_t)$, with a dimension of $\binom{wd+p-1}{p}$. Similar to Eq. 2.2, the linear version of Eq. 3.3 is

$$\begin{aligned}\tilde{\mathcal{K}}_\Delta(\mathbf{x},\mathbf{y}) &\propto \sum_{t=0}^{\infty} \boldsymbol{\phi}_p(\mathbf{U}_t)^{\mathrm{T}} \boldsymbol{\phi}_p(\mathbf{V}_{t+\Delta}) \\ &= \sum_{t=0}^{\infty}\sum_{t'=0}^{\infty} \boldsymbol{\phi}_p(\mathbf{U}_t)^{\mathrm{T}} \boldsymbol{\phi}_p(\mathbf{V}_{t'}) k(t, t'+\Delta) \\ &= \underbrace{\left(\sum_{t=0}^{\infty} \boldsymbol{\phi}_p(\mathbf{U}_t) \otimes \boldsymbol{\varphi}(t)\right)}_{\tilde{\boldsymbol{\psi}}_0(\mathbf{x})}^{\mathrm{T}} \underbrace{\left(\sum_{t'=0}^{\infty} \boldsymbol{\phi}_p(\mathbf{V}_{t'}) \otimes \boldsymbol{\varphi}(t'+\Delta)\right)}_{\tilde{\boldsymbol{\psi}}_\Delta(\mathbf{y})}.\end{aligned} \tag{3.7}$$

3.3 Approximation

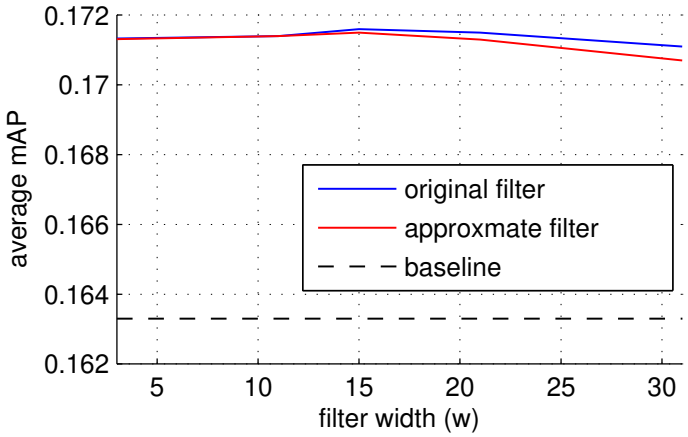


FIGURE 3.4: Average mAPs of TE embedded with original filter versus TE embedded with approximate filter on a subset of EVVE dataset over the filter width w when $d = 16$.

Noticing that both wd and p have a strong bearing on the dimension of $\boldsymbol{\phi}_p(\cdot)$ and $\tilde{\boldsymbol{\psi}}(\cdot)$, we devise following approximation to the original filter in Eq. 3.3.

$$\tilde{s}_t^{*\Delta} = \left(\langle \mathbf{U}_t^*, \mathbf{V}_{t+\Delta}^* \rangle\right)^p = \left(\left\langle \sum_{k=-l}^{l} \sqrt{h_k}\boldsymbol{x}_{t+k}, \sum_{k=-l}^{l} \sqrt{h_k}\boldsymbol{y}_{t+\Delta+k} \right\rangle\right)^p, \tag{3.8}$$

where $\mathbf{U}_t^*, \mathbf{V}_{t+\Delta}^* \in \mathbb{R}^d$. Consequently, the dimension of $\boldsymbol{\phi}_p(\mathbf{U}_t^*)$ becomes $\binom{d+p-1}{p}$, *i.e.* the width w is no longer related to the dimension and can be adjusted freely. However, the dimension is still quadratic in d, the dimension of frame descriptors. A dimension reduction is needed when d is large. This also lead to the rule of choosing filter. Other widely used kernels such as exponential kernel are not feasible because the explicitly expanded vectors have infinite dimensions.

Finally, TE with embedded approximate stability-sensitive filter is formulated as follows.

$$\tilde{\mathcal{K}}^*_\Delta(\mathbf{x}, \mathbf{y}) \propto \tilde{\boldsymbol{\psi}}^*_0(\mathbf{x})^T \tilde{\boldsymbol{\psi}}^*_\Delta(\mathbf{y}), \tag{3.9}$$

where

$$\begin{aligned} \tilde{\boldsymbol{\psi}}^*_0(\mathbf{x}) &= \sum_{t=0}^{\infty} \boldsymbol{\phi}_p \left(\sum_{k=-l}^{l} \boldsymbol{x}_{t+k} \right) \otimes \boldsymbol{\varphi}(t) \\ \tilde{\boldsymbol{\psi}}^*_\Delta(\mathbf{y}) &= \sum_{t'=0}^{\infty} \boldsymbol{\phi}_p \left(\sum_{k=-l}^{l} \boldsymbol{y}_{t'+k} \right) \otimes \boldsymbol{\varphi}(t'+\Delta). \end{aligned} \tag{3.10}$$

We particularly use $p = 2$ throughout the thesis.

To confirm the influence of approximation, we performed a video retrieval task on a subset of EVVE (see section 6.1.1) wherein videos are shorter than 5,000 frames and chose $d = 16$ for computational reason. The results are shown in Fig. 3.4, it can be observed that the approximation only brings trivial performance loss when the width w grows large. Such a loss is acceptable considering the benefit that the computational cost is greatly reduced through approximation. We use the approximate version unless otherwise stated.

3.4 Pros and Cons

Rather than comparing frames one by one, our stability-sensitive filter is able to compare frames while considering their contexts. Only those similar frames with concurrently similar neighborhoods contribute to the video-wise similarity. The effectiveness of the stability-sensitive filter is proved by the performance in a video retrieval task. The average mAP (mean Average Precision) gained 2% improvement by using proposed filter, *i.e.* 35.3% versus 33.3% (see Tab. 6.2).

However, even we apply the approximation, the dimension of $\boldsymbol{\phi}_2(\mathbf{U}^*_t)$ is still quadratic in d, the dimension of frame descriptors. Thus, although our proposed filter is able to alleviate the problem caused by noise, the computational cost is hindering its application. In next section, we show a totally different method for the same purpose with a low computational cost.

Chapter 4

Burst-Survive Temporal Matching Kernel

We have introduced two kinds of match cases in Fig. 3.1 and Fig. 3.2. When we dig into the results of video retrieval, we always find that false matches were responsible for poor performance of TE. These cases are caused by irrelevant videos yet shared scattered similar frames. The falsely matched videos indeed share similar visual elements such as faces, streets, banners, etc. The only difference between true and false matches is the distribution of similar frames. Particularly, in true match cases, similar frames are normally concentrated on specific time spans; while in false match cases, similar frames are scattered globally in the videos. We proposed stability-sensitive filter in Chapter 3 to circumvent false match issues. Filtering noise inside frame-wise similarities is indeed an effective way to achieve better retrieval performance. However, with embedded stability-sensitive filter, the dimension of video descriptors grows inevitably and lead to higher computational cost.

Inspired by the interleaving technique for error correction with respect to burst errors, we devise a novel method to focus on **bursts**, *i.e.* the pattern where continuously high frame-wise similarities last for a short time span. Since the method statistically rules out contributions from noisy frame-wise similarities, only contributions from bursts survive in other words, we name it as BURst-Survive Temporal matching kernel (BURST). BURST provides a similar effect as the stability-sensitive filter, while the concomitant computational burden is trivial.

The intuition is straightforward. Since the frame-wise similarities in a false match case is shaped like white noise, it can be inferred that the frame-wise similarities between a pair of shuffled videos in a false match will still be like white noise. However, in a true match case, the similar frames are aligned continuously along the time and lead to the bursts. When we deliberately shuffle the videos in a true match, the alignment between frames will disappear and the frame-wise similarities between shuffled videos become white noise. The proposed BURST method is designed following this phenomenon. More details are discussed in next section.

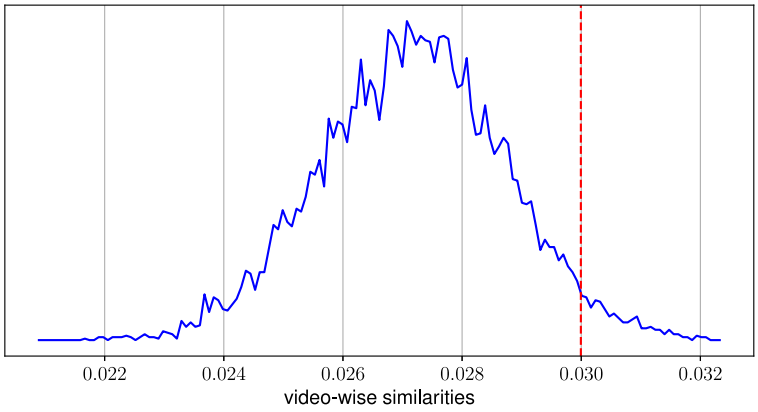


FIGURE 4.1: Probability density approximated by video-wise similarities $\{\mathcal{K}_\Delta(\mathbf{x}, \hat{\mathbf{y}})\}$ obtained from 10,000 times shuffle. The red line shows the original video-wise similarity $\mathcal{K}_\Delta(\mathbf{x}, \mathbf{y})$ without shuffle.

4.1 Shuffle Strategy Analysis

Before consider real cases in video retrieval, we would like to show a toy model for ease of understanding. Say we have two arrays of same length n containing random binary values, *e.g.* $\mathbf{a} = [1, 1, 0, 1, \cdots]$, $\mathbf{b} = [0, 1, 1, 0, \cdots]$. We can measure their similarity by simply computing

$$\mathcal{K}(\mathbf{a}, \mathbf{b}) = \sum_{t=0}^{n} \delta_{a_t, b_t} \tag{4.1}$$

where δ is a Kronecker's delta function, while a_t stands for t-th value in $\mathbf{a}$ as well as b_t in $\mathbf{b}$. Supposing that each binary value obeys the Bernoulli distribution,*i.e.* $a_t \sim B(1, p_a), b_t \sim B(1, p_b)$[1], the distribution of δ_{a_t, b_t} is then determined as $B(1, p)$, where $p = p_a p_b + (1 - p_a)(1 - p_b)$. Furthermore, $\mathcal{K}(\mathbf{a}, \mathbf{b})$ follows a binomial distribution $B(n, p)$. As far as we know, $B(n, p)$ is approximately a normal distribution when n is large enough[2]. Thus, the value $\mathcal{K}(\mathbf{a}, \mathbf{b})$ is most likely distributed near the mean, the expected value. Now we shuffle array $\mathbf{b}$ to $\hat{\mathbf{b}}$, since the distributions of each value are unchanged, there is a high probability that $\mathcal{K}(\mathbf{a}, \hat{\mathbf{b}})$ is a very close to $\mathcal{K}(\mathbf{a}, \mathbf{b})$. In other words, $\mathcal{K}(\mathbf{a}, \mathbf{b}) - \mathcal{K}(\mathbf{a}, \hat{\mathbf{b}}) \approx 0$.

Let's move on to another situation, given arrays $\mathbf{a}, \mathbf{b}$ with the same definitions above. However, $\mathbf{a}$ and $\mathbf{b}$ are observed to have a continuously similar or same section, *i.e.* a **burst**. That means $\mathcal{K}(\mathbf{a}, \mathbf{b})$ is initially a large value (far larger than the expected value), due to the contribution of the burst. In a statistic view, $\mathcal{K}(\mathbf{a}, \mathbf{b})$ is less likely to be far from the mean, which also means the burst is kind of a rare case in return. Obviously, if we deliberately shuffle $\mathbf{b}$ to $\hat{\mathbf{b}}$, we can hardly expect the rare bursts appear again, or $\mathcal{K}(\mathbf{a}, \hat{\mathbf{b}})$ is still as

[1]Here, $B(n, p)$ stands for a binomial distribution by convention. $n \in \mathbb{N}$ is the number of trials, $p \in [0, 1]$ is the probability of getting success in each trial.

[2]In real cases, the number of frames in a video is usually large.

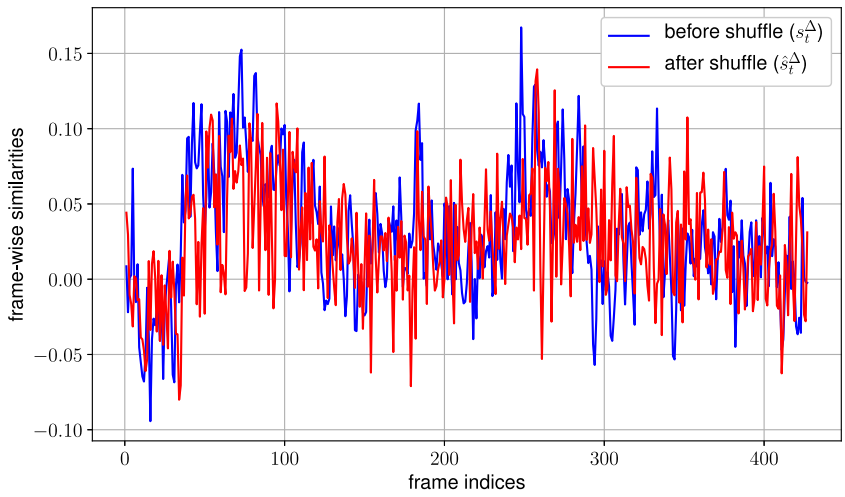


FIGURE 4.2: A comparison between frame-wise similarities $\{s_t^\Delta\}$ (before shuffle) and $\{\tilde{s}_t^\Delta\}$ (after shuffle) in a false match case.

large as $\mathcal{K}(\mathbf{a}, \mathbf{b})$. As a result, it is totally plausible that $\mathcal{K}(\mathbf{a}, \mathbf{b}) - \mathcal{K}(\mathbf{a}, \hat{\mathbf{b}}) > 0$, and is even large enough.

We only have to do few changes on the toy model for real cases. The binary values in arrays are displaced by frame descriptors, and the Kronecker's delta in Eq. 4.1 are switched to a frame-wise similarity measurement as in Eq. 2.1. Taking videos $\mathbf{x}$ and $\mathbf{y}$ as example again, frame-wise similarities are $[s_0^\Delta, \cdots, s_t^\Delta, \cdots]$ where $s_t^\Delta = \boldsymbol{x}_t^T \boldsymbol{y}_{t+\Delta}$. As we know, noise among frame-wise similarities is caused by haphazard similar frames in irrelevant videos. Thus, in a false match case where $\{s_t^\Delta\}$ are merely composed of noise, even if we shuffle the order of frames in $\mathbf{y}$, the pattern of frame-wise similarities is still noise-like (Fig. 4.2). Denoting shuffled $\mathbf{y}$ as $\hat{\mathbf{y}}$ and its frame sequences as $[\hat{\boldsymbol{y}}_0, \cdots, \hat{\boldsymbol{y}}_t, \cdots]$, frame-wise similarities between $\mathbf{x}$ and $\hat{\mathbf{y}}$ are $[\hat{s}_0^\Delta, \cdots, \hat{s}_t^\Delta, \cdots]$, where $\hat{s}_t^\Delta = \boldsymbol{x}_t^T \hat{\boldsymbol{y}}_{t+\Delta}$. Since probabilistically no bursts appear in similarities before and after shuffle, we can expect the same conclusion in the toy model for false match cases:

$$\mathcal{K}_\Delta(\mathbf{x}, \mathbf{y}) - \mathcal{K}_\Delta(\mathbf{x}, \hat{\mathbf{y}}) \approx 0. \tag{4.2}$$

To substantiate our assumption, we test on several false match cases. Firstly, a video pair is obtained from a false match case. We say the two videos are $\mathbf{x}$ and $\mathbf{y}$ in conformity with the above statement. For proving that Eq. 4.2 satisfies probabilistically, we shuffle video $\mathbf{y}$ 10,000 times by different permutations, and evaluate the video-wise similarity following Eq. 2.1 between $\mathbf{x}$ and the shuffled $\mathbf{y}$. The 10,000 video-wise similarities are then used for plotting Fig. 4.1, a probability density graph.

As can be easily seen, the probability density shows a bell-shaped curve which is apparently the characteristic of a normal distribution. The original video similarity before shuffle is very close to the mean as we conjectured. For further demonstration, we show the frame-wise similarities before and

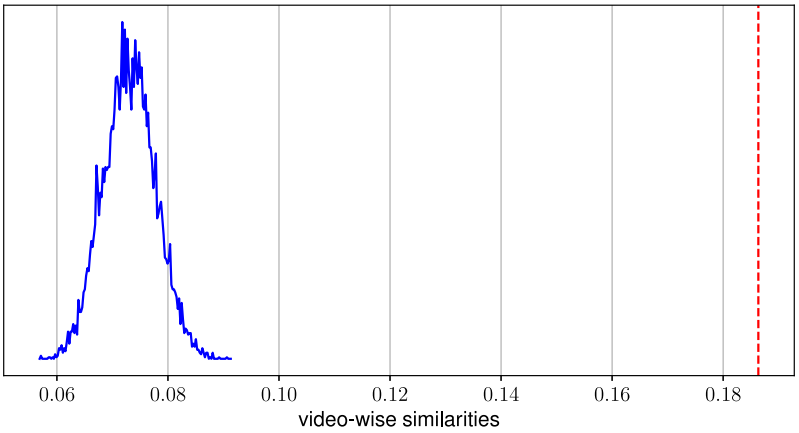


FIGURE 4.3: Probability density approximated by video-wise similarities $\{\mathcal{K}_\Delta(\mathbf{x}, \hat{\mathbf{y}})\}$ obtained from 10,000 times shuffle. The red line shows the original video-wise similarity $\mathcal{K}_\Delta(\mathbf{x}, \mathbf{y})$ without shuffle.

after shuffle for clearer comparison in Fig. 4.2. Both $\{s_t^\Delta\}$ and $\{\tilde{s}_t^\Delta\}$ are noisy which make Eq. 4.2 reasonably satisfied.

In a real match case, since the rare situation that two videos have consecutive similar frames will be crumbled after a deliberate shuffle, the original video-wise similarity is far away from the expected similarity after shuffle. We conduct the same test on some true match cases, the distribution of video-wise similarities is shown in Fig. 4.3.

Similarly, we show the frame-wise similarities before and after shuffle for the true match case in Fig. 4.4. It can be observed that the contribution from the burst, a section of continuously high frame-wise similarities caused by consecutive similar frames, is strong enough to survive in a subtraction such that

$$\mathcal{K}_\Delta(\mathbf{x}, \mathbf{y}) - \mathcal{K}_\Delta(\mathbf{x}, \hat{\mathbf{y}}) > 0. \tag{4.3}$$

Nevertheless, all our analyses begin with an assumption that the elements in compared sequences are randomly distributed to some extent. More specifically, we suppose the elements are generated by information sources having a relatively high entropy. For the videos where frames are time-varying, our shuffle strategy works well; while for other videos who are almost stationary over time, the shuffle has even no effect. We should pay attention to this situation in the design of an advanced temporal matching kernel.

To summarize the above theory, we conclude three points below.

1. In a comparison of two sequences, we can differentiate if there are bursts by using the shuffle strategy;

2. The shuffle strategy works well when the sequences are long enough according to our assumption;

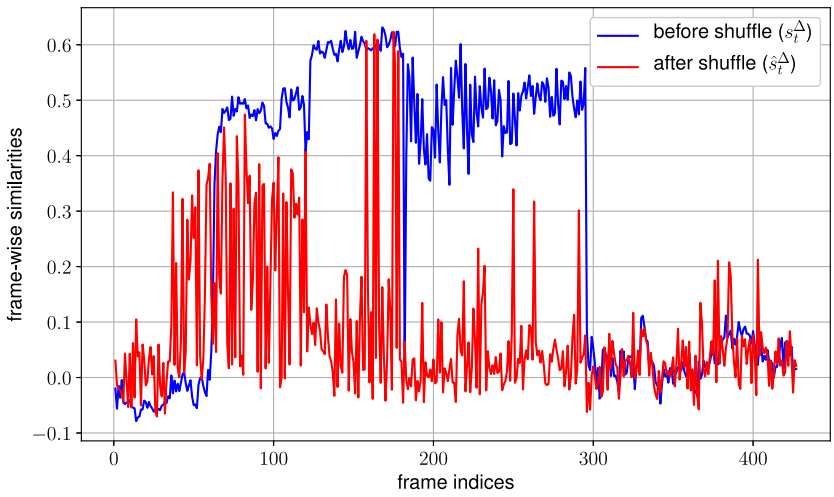


FIGURE 4.4: A comparison between frame-wise similarities $\{s_t^\Delta\}$ (before shuffle) and $\{\tilde{s}_t^\Delta\}$ (after shuffle) in a true match case.

3. The shuffle strategy has virtually no effect when either sequence is almost stationary.

We design our novel burst-survive temporal matching kernel following these points.

4.2 Burst-Survive Temporal Matching Kernel

Based on the theory proposed in section 4.1, the design of the BURst-Survive Temporal matching kernel (BURST) is rather simple. With the same definition used in Eq. 2.1, we extend Eq. 4.2 and Eq. 4.3 as follows,

$$\begin{aligned}\mathcal{K}_\Delta(\mathbf{x}, \mathbf{y} - \hat{\mathbf{y}}) = \mathcal{K}_\Delta(\mathbf{x}, \mathbf{y}) - \mathcal{K}_\Delta(\mathbf{x}, \hat{\mathbf{y}}) = \sum_{t=0}^{\infty} s_t^\Delta - \hat{s}_t^\Delta = \sum_{t=0}^{\infty} \langle \boldsymbol{x}_t, \boldsymbol{y}_{t+\Delta} - \hat{\boldsymbol{y}}_{t+\Delta} \rangle \\ \approx \Bigg(\underbrace{\sum_{t=0}^{\infty} \boldsymbol{x}_t \otimes \boldsymbol{\varphi}(t)}_{\boldsymbol{\psi}_0(\mathbf{x})} \Bigg)^T \Bigg(\underbrace{\sum_{t'=0}^{\infty} (\boldsymbol{y}_{t'} - \hat{\boldsymbol{y}}_{t'}) \otimes \boldsymbol{\varphi}(t' + \Delta)}_{\boldsymbol{\psi}_\Delta(\mathbf{y} - \hat{\mathbf{y}})} \Bigg).\end{aligned} \tag{4.4}$$

Notice that since we maintain the form of TE, we can still enjoy the benefits of TE's framework. As explained in Eq. 2.6, DC components provide a baseline on which a fluctuation is drawn by AC components on each shift Δ for localizing the offset between videos. In other words, DC components are mainly responsible for the retrieval performance while AC components are in charge of the localization performance. However, when the frame descriptors of $\mathbf{y}$ are subtracted by their shuffled ones to construct a burst-survive video descriptor $\boldsymbol{\psi}_\Delta(\boldsymbol{y} - \hat{\boldsymbol{y}})$, the DC component (Eq. 2.5) in $\boldsymbol{\psi}_\Delta(\boldsymbol{y} - \hat{\boldsymbol{y}})$ will be

a zero vector because it is proportional to the sum aggregation of $\{\boldsymbol{y} - \hat{\boldsymbol{y}}\}$. This will directly result in a huge drop on the retrieval performance. Thus, after obtaining AC components through shuffle strategy, we have to supplement a DC component to $\boldsymbol{\psi}_\Delta(\boldsymbol{y} - \hat{\boldsymbol{y}})$. Since we mainly focus on the localization functionality in our proposed method, we only employ a naive sum aggregation for computing the DC component. For even better retrieval performance, a more exquisite BoF feature, such as the counting grid aggregation [10] can also be applied.

More importantly, by designating DC component to the sum-aggregated frame descriptors or other BoF descriptors, we can circumvent the problem caused by near-stationary videos mentioned in section 4.1.

It should also be noticed that the shuffle strategy is only needed to be performed on one video in a pair of videos for comparison. We call it an asymmetric scheme. One may be curious about the symmetric scheme though, it is defined as

$$\begin{aligned}\mathcal{K}_\Delta(\mathbf{x} - \hat{\mathbf{x}}, \mathbf{y} - \hat{\mathbf{y}}) &= \mathcal{K}_\Delta(\mathbf{x}, \mathbf{y}) - \mathcal{K}_\Delta(\mathbf{x}, \hat{\mathbf{y}}) - \mathcal{K}_\Delta(\hat{\mathbf{x}}, \mathbf{y}) + \mathcal{K}_\Delta(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \\ &\approx \underbrace{\left(\sum_{t=0}^{\infty} (\boldsymbol{x}_t - \hat{\boldsymbol{x}}_t) \otimes \boldsymbol{\varphi}(t) \right)}_{\boldsymbol{\psi}_0(\mathbf{x} - \hat{\mathbf{x}})}^{T} \underbrace{\left(\sum_{t'=0}^{\infty} (\boldsymbol{y}_{t'} - \hat{\boldsymbol{y}}_{t'}) \otimes \boldsymbol{\varphi}(t' + \Delta) \right)}_{\boldsymbol{\psi}_\Delta(\mathbf{y} - \hat{\mathbf{y}})}.\end{aligned} \tag{4.5}$$

Although the asymmetric scheme is already effective for our purpose, however, we actually use a symmetric scheme in practice. The reason is twofold. First, we have no idea which video in comparison is longer than the other, and shuffling a short video is not always effective enough, while a symmetric scheme can always ensure that the long video is shuffled. Second, a symmetric scheme provides convenience in the implementation.

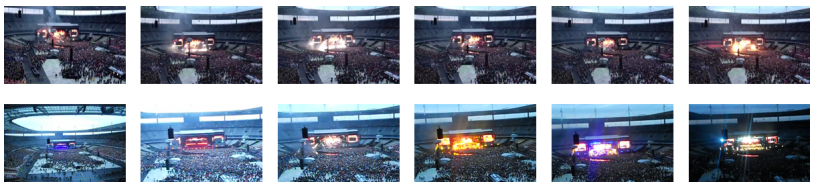
Similar to Eq. 2.4, the burst-survive video descriptors are obtained as bellow.

$$\hat{\boldsymbol{\psi}}_0(\mathbf{x}) = [\hat{\mathbf{D}}^\mathrm{T}, \hat{\mathbf{C}}_1^\mathrm{T}, \hat{\mathbf{S}}_1^\mathrm{T}, \ldots, \hat{\mathbf{C}}_m^\mathrm{T}, \hat{\mathbf{S}}_m^\mathrm{T}]^\mathrm{T}, \tag{4.6}$$

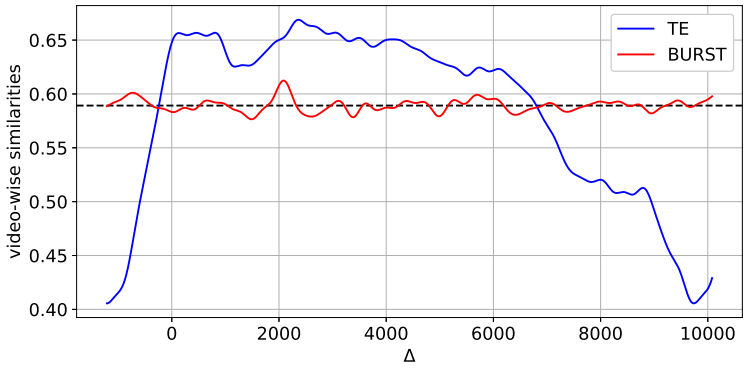
where

$$\begin{aligned}\hat{\mathbf{D}} &\approx \sqrt{a_0} \sum_{t=0}^{\infty} \boldsymbol{x}_t, \\ \hat{\mathbf{C}}_i &\approx \sqrt{a_i} \sum_{t=0}^{\infty} (\boldsymbol{x}_t - \hat{\boldsymbol{x}}_t) \cos\left(\frac{2\pi}{T} i t\right), \\ \hat{\mathbf{S}}_i &\approx \sqrt{a_i} \sum_{t=0}^{\infty} (\boldsymbol{x}_t - \hat{\boldsymbol{x}}_t) \sin\left(\frac{2\pi}{T} i t\right).\end{aligned} \tag{4.7}$$

The DC component and AC components are separately L2 normalized. Moreover, we multiply a coefficient λ on the DC component to balance the contributions of DC and AC components. λ is empirically set as 0.4 in our experiments.



(A) Frame sequences of two near-stationary videos, where frames have little changes over time.



(B) Video-wise similarities obtained from a comparison between two near-stationary videos.

FIGURE 4.5: An example of the comparison between near-stationary videos.

4.3 Effectiveness of BURST

The burst-survive temporal matching kernel is designed for eliminating the noise in frame-wise similarities which seems to lead to false matches. We arbitrarily choose several true matches and false matches obtained from TE's results, and compare the results of TE and BURST to show the effectiveness of BURST.

Fig. 4.6 shows the results. The black lines show the inner products of DC components obtained by sum aggregation with a L2 normalization applied afterwards. Based on them, AC components from TE and BURST draw curves over Δ, which are used for localizing the offsets. It can be observed from the top two figures that by using BURST, video-wise similarities remain high enough where its value is mainly contributed from bursts. Another two figures in the bottom are false match cases. Their video-wise similarities are circumscribed a lot, since BURST are capable of "killing" noise as discussed in Eq. 4.2.

For near-stationary videos, the AC components in the BURST are less useful though. Fig. 4.5 shows that videos are almost unchanged over time, our assumption for the shuffle strategy is not satisfied in this case. Since almost each frame in video $\mathbf{x}$ is the same with its shuffled version $\hat{\mathbf{x}}$, $\boldsymbol{x}_t - \hat{\boldsymbol{x}}_t \approx \mathbf{0}$ which makes AC components are approximately zero vectors according to Eq. 4.7. Fortunately, the supplemented DC component set for such situation ensures the video similarity score is still a reasonable value (the peak value of the red line). The effectiveness of BURST is further proved in Chapter 6.

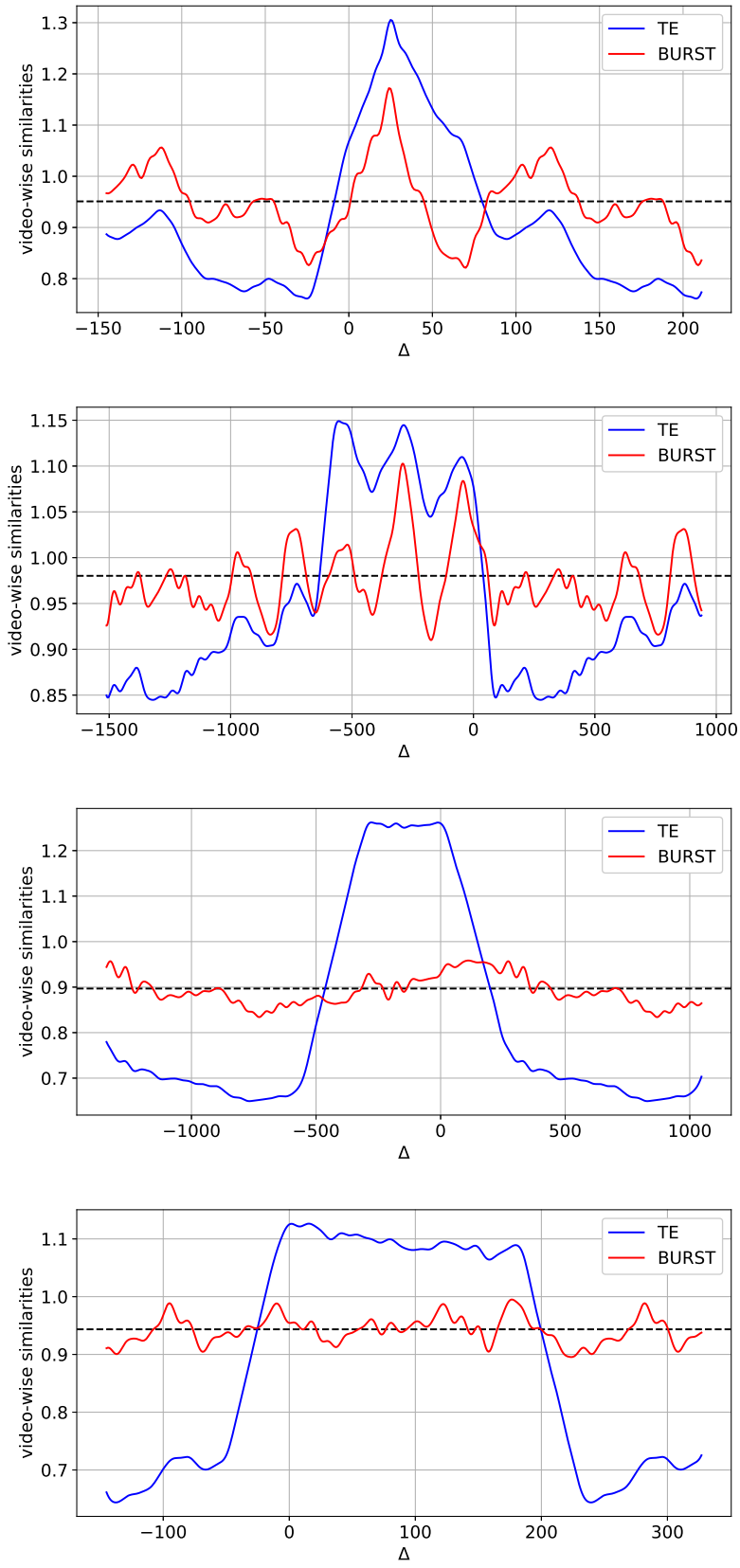


FIGURE 4.6: Results of TE and BURST. The top two are true matches, while bottom ones are false matches. Black line shows the inner products of sum-aggregated descriptors (DC components).

Chapter 5

Golden Ratio Based Multi-Period Strategy

5.1 Multi-Period Strategy

Given two videos **x** and **y**, we usually map each temporal position of a frame to a unique value inside a period. A toy model is visualized in Fig. 5.1. For simplicity, we measure frame-wise similarities by colors, same colors output 1, and 0 otherwise. In this example, the query video **x** comprises 4 frames, and a retrieved video has 16 frames. When we set the period as a relatively large value, $T = 16$ in this case, we are able to locate a sole peak among video-wise similarities (the red dot), whose value is the video similarity score while its position represents the offset, $\Delta^T_{\mathbf{x},\mathbf{y}} = 9$ here.

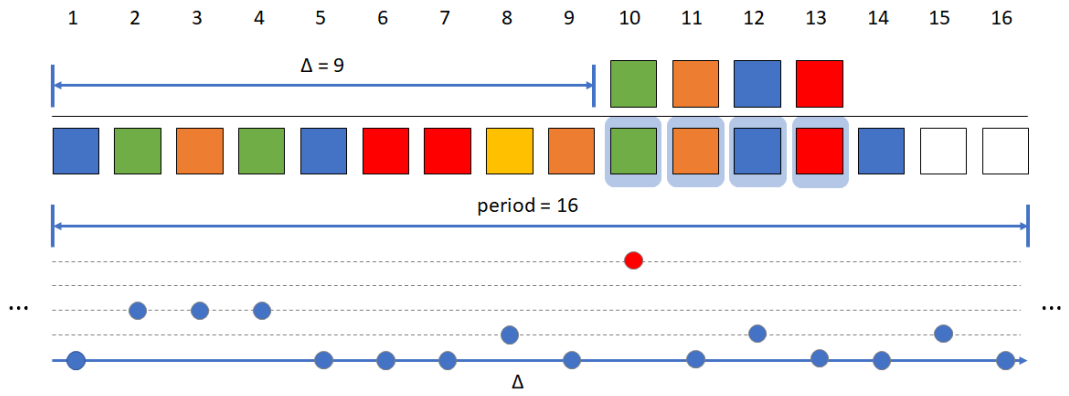


FIGURE 5.1: An example showing the localization of the relative offset between two toy videos, by using a single period.

A single long period works theoretically, however, there are some practical issues. Firstly, a long period directly leads to a low resolution for the offset's localization. Moreover, the only way to increase the resolution is to employ more dimensions in the temporal vector $\boldsymbol{\varphi}(t)$ for exploiting higher frequencies. Since more dimensions lead to higher computational cost, it is definitely not a wise idea.

As introduced in section 2.3.2, a multi-period strategy is proposed for sidestepping the problems caused by a single long period. The idea is quite straightforward. Now that a short period provides high resolution but is

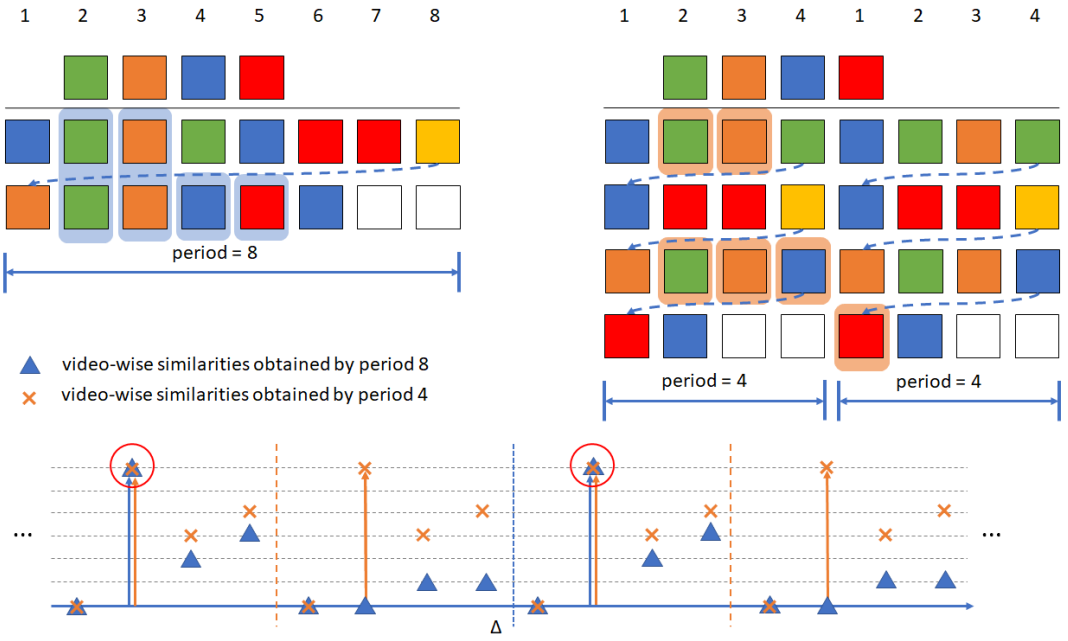


FIGURE 5.2: An example showing the localization of the relative offset between two toy videos, by using two short periods who are not relatively prime.

unable to exactly locate the offset, we can combine the offset candidates obtained by several short periods to construct a real offset. For instance, we choose two periods $T_1 = 4, T_2 = 8$, and show the results in Fig. 5.2.

As we know, $\boldsymbol{\varphi}(t)$ is a periodic function whose period is T. Thus, a t-th frame has the same timestamp with $(t+kT)$-th frame since $\boldsymbol{\varphi}(t) = \boldsymbol{\varphi}(t+kT)$, where k can be any integer. After embedding, we evaluate the video-wise similarities over $[0, T)$, and the peak is denoted as $\Delta^T_{\mathbf{x},\mathbf{y}} \in [0, T)$. The potential candidates of the real offset are $\{\Delta^T_{\mathbf{x},\mathbf{y}} + kT\}$ correspondingly. In Fig. 5.2, those candidates are $\{\Delta^8_{\mathbf{x},\mathbf{y}} + 8k = 8k + 1\}$ and $\{\Delta^4_{\mathbf{x},\mathbf{y}} + 4l = 4l + 1\}$, where $k, l \in \mathbb{Z}$. As a result, there are two possible potentials at $\{1, 9\}$ (denoted by red circles). Unfortunately, we cannot discriminate which one is the real offset. Some of you may already find the problem, we cannot obtain a sole peak in the concerned range if we choose short periods whose Greatest Common Divisor (GCD) is not small enough.

A plausible solution proposed by Poullot *et al.* [21] is to choose periods who are relatively prime. For example, we set periods as $T_1 = 5, T_2 = 7$ in Fig. 5.3. Now the offset candidates become $\{\Delta^7_{\mathbf{x},\mathbf{y}} + 7k = 7k + 2\}$ and $\{\Delta^5_{\mathbf{x},\mathbf{y}} + 5l = 5l + 4\}$, where $k, l \in \mathbb{Z}$. The real offset 9 is then constructed by these potentials, and is unique in $[-\mathbf{x}.\text{length} + 1, \mathbf{y}.\text{length}]$ because the Least Common Multiple (LCM) of 5, 7 is greater than $\mathbf{x}.\text{length} + \mathbf{y}.\text{length}$.

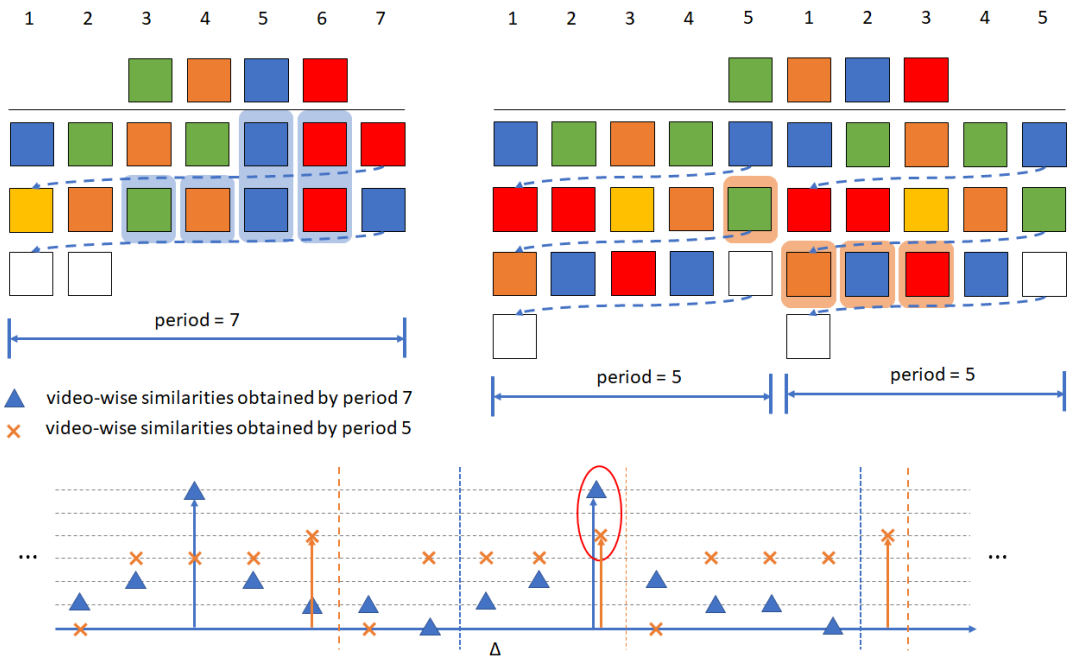


FIGURE 5.3: An example showing the localization of the relative offset between two toy videos, by using two short periods who are relatively prime.

5.2 Design with Golden Ratio

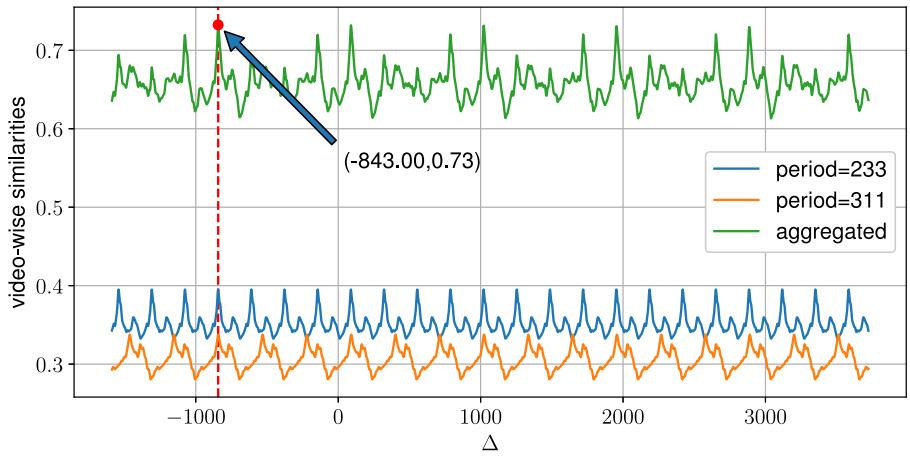
However, we find that this solution is debatable, especially in practice. We select video $\mathbf{x}^1$ as a query and $\mathbf{y}^2$ as a compared video, their relative offset is confirmed to be around -145 frames (15 fps). We conduct offset localization experiments by using three pairs of periods, $(233, 311)$, $(233, 367)$ and $(233, 377)$. The offset are obtained by two steps: 1) evaluate video-wise similarities over the shift Δ for each periods, 2) aggregate these video-wise similarities to an overall similarities and locate the peak. Fig. 5.4 shows the results. As everyone knows, $233, 311, 367$ are all primes. However, neither $(233, 311)$ nor $(233, 367)$ gives correct answer, which is totally contrary to our expectations since primes are indeed effective in the toy model. The reason is sort of ridiculous. Because $\frac{233}{311} \approx \frac{3}{4}$ and $\frac{233}{367} \approx \frac{2}{3}$, even they are primes, their approximated LCMs are merely 933 and 700 respectively, which are too small to disambiguate offset candidates. Interestingly though, the pair $(233, 377)$ successfully locates an accurate offset. The secret behind this setting is the golden ratio. A golden ratio is the number which is most hardly to be approximated by a fraction. Thus we can simply choose periods who have a golden ratio proportion. Furthermore, a well-known fact is that the ratio between two adjacent numbers in the Fibonacci sequence asymptotically approaches the golden ratio. To unveil the truth, 233 and 377 are exactly a pair of adjacent numbers selected from the Fibonacci sequence. Since the

[1] http://www.youtube.com/watch?v=1MuBD439BY4.
[2] http://www.youtube.com/watch?v=PkT2WJQ_KdQ.

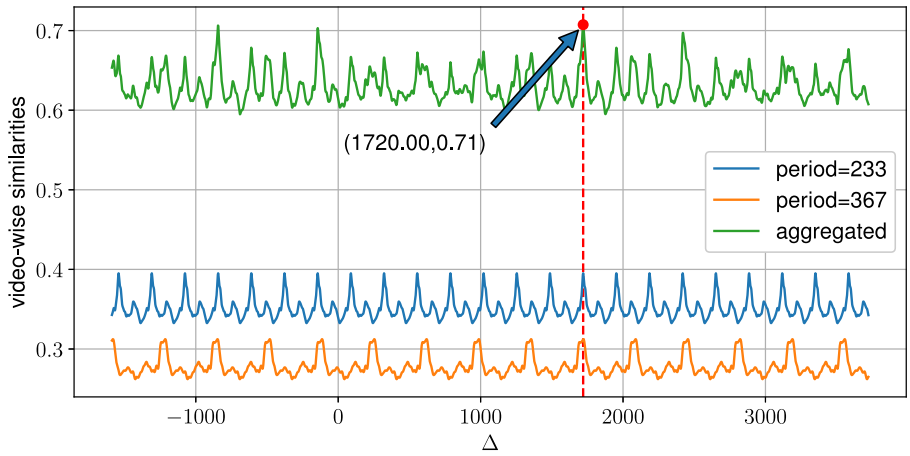
Fibonacci sequence is proved to be a strong divisibility sequence, *i.e.*

$$gcd(F_m, F_n) = F_{gcd(m,n)} \tag{5.1}$$

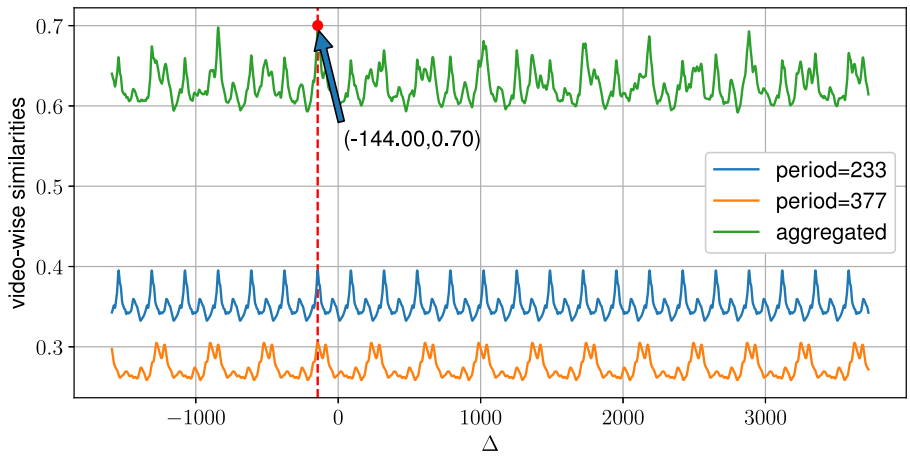
where $F_0 = 0, F_1 = 1, F_{n+1} = F_n + F_{n-1}$ following the general definition. The adjacent numbers in the Fibonacci sequence are always co-prime. It is safe to say, if only the product of the two adjacent numbers in the Fibonacci sequence is larger than **x**.length + **y**.length, we can choose them as a pair of periods.



(A)



(B)



(C)

FIGURE 5.4: An example showing the ability of localization under different period settings, only the periods (233, 377) lead to correct offset.

Chapter 6

Experiments and Results

6.1 Datasets and Evaluation Protocol

6.1.1 EVVE Dataset

The EVVE dataset[1] is an event retrieval dataset introduced in [23]. Instead of an event detection task, this dataset is dedicated to the retrieval of particular events. The whole dataset contains 620 query videos and 2,375 videos in database, categorized into 13 events. Videos were down-sampled at 15 fps, and each frame was processed to a 1,024-d MVLAD [14] descriptor by the provider. The shortest video only contains 12 frames, while the longest one contains 59,810 frames, which makes this dataset more challenging. We evaluated the performance in terms of the mean Average Precision (mAP) for each event and the average mAP for all events, following the standard protocol.

6.1.2 TV CBCD 2011 Dataset

To evaluate the offset localization performance, we used the content-based copy detection set of the TRECVID 2011[2] evaluation campaign [28]. It contains 1,608 queries 16,776 reference videos, extracted at 30 fps. The queries are programmatically created from 201 videos. Except some queries made by non-reference videos, there are 134 original queries having their correspondences in database, and each of them is processed into 8 queries by using different transformations.

We matched each transformed query video with its original reference video. The offsets estimated by TE and our proposed method are compared with the ground-truth.

6.2 Implementation Details

6.2.1 Frame-level descriptor

For EVVE dataset, we collected MVLAD descriptors for a fair competition with other state-of-the-arts who are evaluated with MVLADs. Moreover, we

[1] http://pascal.inrialpes.fr/data/evve/

[2] http://www-nlpir.nist.gov/projects/tv2011/#ccd

extract descriptors by using pre-trained CNN models for both EVVE and CBCD 2011. We finally adopt the last convolutional layer of a ResNet-50 after evaluating the baseline with descriptors extracted from AlexNet [17], VGG [25], and ResNet [12]. The dimension of raw frame descriptors extracted from ResNet-50 is 2,048.

6.2.2 PCA Whitening

PCA whitening is reported to provide significant benefits on an image or video retrieval task [14]. By taking videos in CBCD 2011 dataset as distractors for EVVE, we fit a PCA by using the sum-aggregated video descriptors of CBCD 2011. Before modulating the TE and BURST video descriptors, we apply the PCA on frame descriptors as a pre-process. The dimension of frame descriptors extracted from ResNet-50 is reduced to 1,024 after PCA.

6.2.3 Re-ranking by Query Expansion

Since both TE and BURST provide the ability of consistency check, we employ DoN for query expansion as mentioned in section 2.4. We set the number of retrieved videos in the short list (N_1) as 10, and the number of videos in the far list for subtraction (N_2) as 2000.

6.2.4 Computational Complexity

Thanks to the elegant framework of TE, each video is embedded as a single descriptor. Instead of a heavy process on videos, we only have to handle video descriptors.

When the dimension of frame descriptors is d and the number of frequencies we used in the temporal kernel (Eq. 2.3) is m, the dimension of a video descriptor becomes $d \times m$. The heaviest computation is the inner product between two video descriptors, whose complexity is $\mathcal{O}(d \times m)$. The offset localization is related to the period T, the computational complexity is $\mathcal{O}(T \times m)$. Thus, we desire a small T and m for better computational performance. The multi-period strategy satisfies this request.

6.3 Results

6.3.1 Results of Stability-Sensitive Filter

Embedding the stability-sensitive filter into TE enlarges the video descriptors and results in high computational cost. To conduct the retrieval task on a practical level, a dimension reduction is applied on the frame descriptors. Tab. 6.1 shows the average mAPs evaluated under different dimension d by using TE.

We use 256-d frame descriptors to embed the stability-sensitive filter. Another important parameter is w, the width of the filter. As section 3.3 points

TABLE 6.1: Results of TE with Different d

dimension d	1,024	256	128	64	32	16
avg-mAP	0.3341	0.3327	0.3182	0.3071	0.2886	0.2421

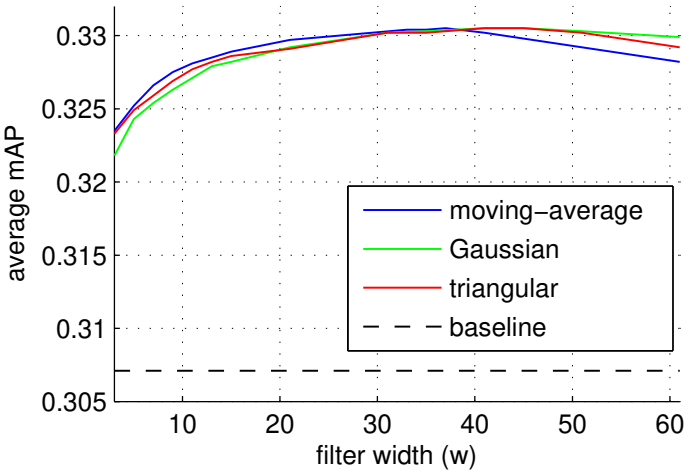


FIGURE 6.1: Average mAPs of TE with embedded stability-sensitive filter on EVVE dataset. $\{h_k\}$ is selected as a moving-average, Gaussian, and triangular filter ($d = 64$).

out, the original embedding (Eq. 3.3) is approximated by Eq. 3.8 for computational reason. The parameter w can be adjusted freely since it is no longer related to the dimension of video descriptors after approximation. We simply choose w that leads to best retrieval performance.

The stability-sensitive filter is defined as a combination of a linear filter $\{h_k\}$ and a monomial $(\cdot)^p$ (Eq. 3.2). Besides the monomial kernel fixed to be a square, we tried different types of the linear filter, *i.e.* moving average, Gaussian, and triangular filter, for seeking better performance. However, Fig. 6.1 shows that the type of the linear filter affects the stability [3] over w while has limited effect on precision. We simply choose moving-average filter because it achieves the peak mAP with smallest w. When w is around 37 frames, corresponding to 2.5 seconds of time (15 fps), we obtain the peak mAP. Hence, we set $w = 37$.

Tab. 6.2 compares Temporal matching kernel with Embedded Stability-Sensitive Filter (TESSF) and original TE on EVVE dataset. The results show that our method achieves better performance compared to the baseline of TE ($d = 256$), 33.27% versus 35.30%, and is even better than the baseline of TE ($d = 1,024$), 33.41%. Furthermore, we conducted query expansion by using DoN strategy. The results also show the effectiveness of our filter.

However, as discussed in section 3.3, the dimension of frame descriptors

[3]Moving average filter is slightly less stable compared to Gaussian and triangular filter.

TABLE 6.2: Performance of TE with embedded stability-sensitive filter compared with TE by using MVLADs on EVVE dataset, where $T = 65,537$.

	d	**avg-mAP**
TE	1,024	0.3341
TE	256	0.3327
TESSF	256	**0.3530**
TE (DoN)	1,024	0.4033
TESSF (DoN)	256	**0.4114**

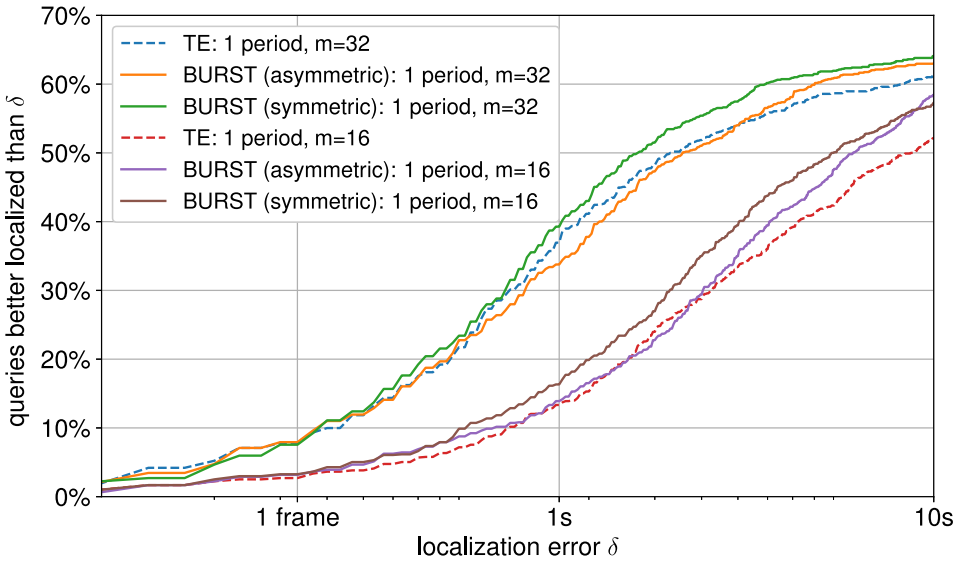


FIGURE 6.2: Localization accuracies of original TE, asymmetric and symmetric BURST with a single period, on CBCD 2011 dataset. m is the number of frequencies in timestamps (Eq. 2.3).

after embedding the stability-sensitive filter will be quadratic in d and consequently result in high computational cost. So even the stability-sensitive filter is proved to be effective for improving the performance, the computational burden carried by it can not be neglected in practice. In next section, we show that BURST is superior to TESSF in all aspects.

6.3.2 Results of BURST

We mainly demonstrate the effectiveness of BURST from three perspectives: retrieval performance, localization performance and computational complexity.

First of all, since BURST efficiently razes the misleading noise from false matches, the retrieval performance will be improved in theory. We evaluated

the performance of BURST by using the same period $T = 65,537$, and obtained the average mAP 31.42%. To our disappointment, BURST performed even worse than TE. Does it mean that BURST is useless? No, we just used it in a wrong way.

An intuitive explanation to the bad performance is that the power of BURST is only released when bursts can be found. However, bursts are easily missed due to the low resolution related to a single long period. A single long period is not suitable for the BURST method. Hence, a good retrieval performance is dependent on a good localization performance. Fig. 6.2 shows the localization accuracy of TE, asymmetric and symmetric BURST. The localization performance was evaluated by the percentage of queries better localized than an certain error (higher is better), in CBCD 2011 dataset. It is obvious that the more frequencies, corresponding to a greater m, we use for modulation, the better localization performance we will achieve. By using a certain number of frequencies, it is shown that the symmetric BURST generally has a better localization performance than the rest. This is reasonable because the offset obtained by BURST is exactly where the bursts shows, instead of where noise exists, while TE is unable to tell the difference.

Last but not least, the computational complexity of BURST never outgrows that of TE. The shuffle and subtraction (Eq. 4.7) are the only extra operations, which are totally negligible.

6.3.3 Results with Multiple Periods

Following our golden ratio based multi-period strategy, we evaluated the localization performance of symmetric BURST by using periods selected from the Fibonacci sequence adjacently. Fig. 6.3 shows that a proper pair of periods, *e.g.* $(233, 377)$ performs better than others. It should be noticed that the BURST modulated by a single long period ($m = 32$) is grossly worse than the BURST modulated by two short periods ($m = 16$ for each period). Generally speaking, the choice of periods is related to the length of videos. It is much safer to use more than two periods, *e.g.* $(233, 377, 610, 987)$. In our experiments, four periods indeed outperformed an adjacent pair.

For someone who may be curious about the performance of periods who are not adjacent Fibonacci numbers, we show the results of four different pairs of periods in Fig. 6.4. Except the pair $(233, 367)$ which is composed of prime numbers, the numbers in other pairs are all selected from the Fibonacci sequence. Actually, $233, 377, 610, 987$ are continuous numbers in the Fibonacci sequence. It can be seen that the adjacent pair $(233, 377)$ achieved the best performance. It should be again emphasized that the periods should be selected as adjacent numbers in the Fibonacci sequence.

Tab. 6.3 shows the results of the sum-aggregated video descriptors[4], TE, and BURST by using multiple periods. Same as we expected, BURST with multiple periods showed the best performance among them.

[4]The video descriptor created by summing up all frame descriptors with a L2 normalization afterwards.

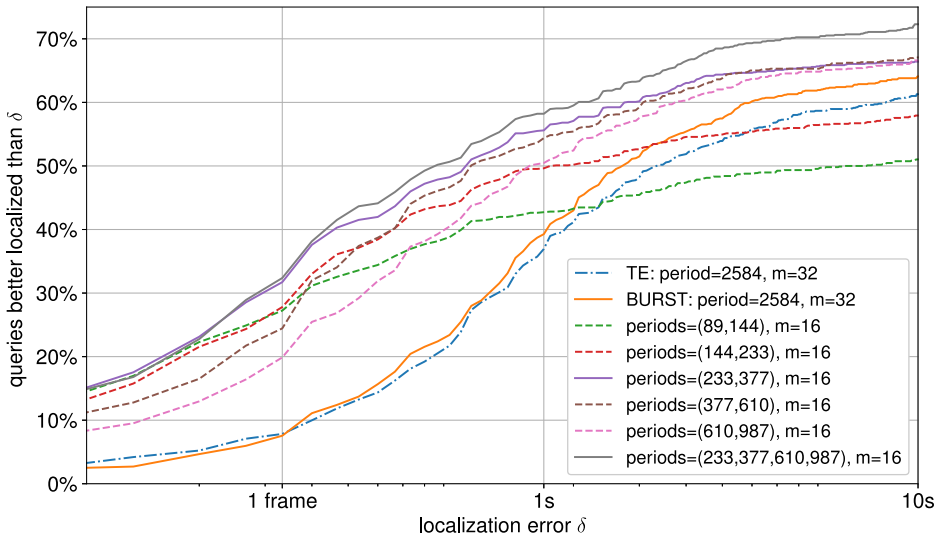


FIGURE 6.3: Localization accuracies of symmetric BURST with multiple periods, on CBCD 2011 dataset. m is the number of frequencies in timestamps (Eq. 2.3).

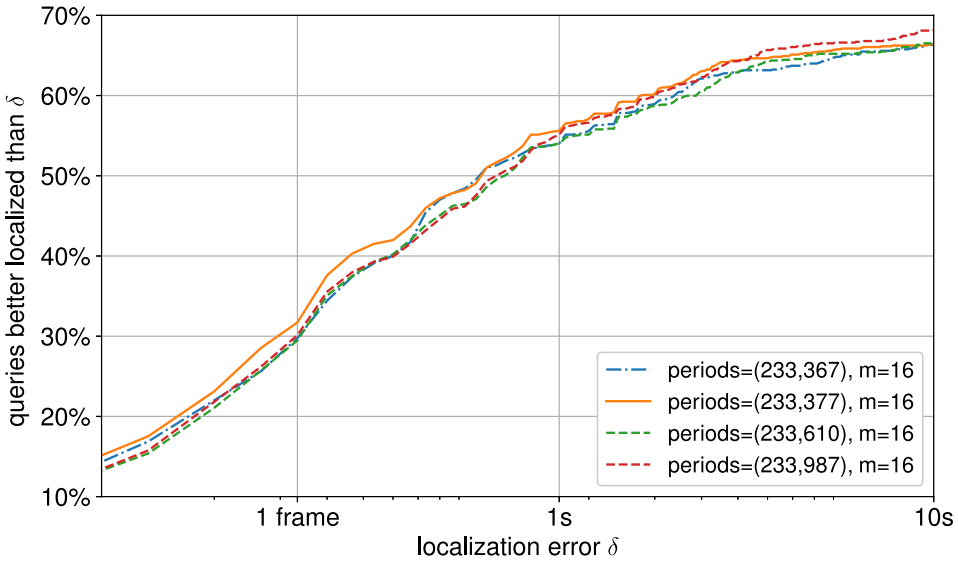


FIGURE 6.4: Localization accuracies of symmetric BURST with different period pairs, on CBCD 2011 dataset. m is the number of frequencies in timestamps (Eq. 2.3).

TABLE 6.3: Performance of BURST compared with baselines on EVVE by using multiple periods, where $m = 16$.

Method	Periods	Feature	fps	avg-mAP
Sum-aggregated	-	MVLAD	15	0.334
	-	ResNet-50	5	0.461
TE	(2027,3019,5003,7019)	MVLAD	15	0.332
	(144,233,377,610)	ResNet-50	5	0.461
TE (DoN)	(2027,3019,5003,7019)	MVLAD	15	0.413
	(144,233,377,610)	ResNet-50	5	0.558
BURST	(1597,2584,4181,6765)	MVLAD	15	**0.383**
	(144,233,377,610)	ResNet-50	5	**0.496**
BURST (DoN)	(1597,2584,4181,6765)	MVLAD	15	**0.453**
	(144,233,377,610)	ResNet-50	5	**0.573**

Comparison with other state-of-the-arts is shown in Tab. 6.4. Our method gives undoubtedly best retrieval performance among methods who are capable of offset localization. It should be pointed out we achieved +4.8% (38.3% versus 33.5%) improvement to our TE baseline by using MVLADs, and +3.5% (49.6% versus 46.1%) by using ResNet-50 features. And our temporal information embedded method BURST achieved even better performance than the typical BoF method SHP, which is a real breakthrough.

Even though the recent Counting Grid Aggregation (CGA) has updated the record of BoF methods, it does not provide localization functionality, and we found its computational complexity of learning the counting grid with EM algorithm is extremely high thus only viable on lots of GPUs. We are confident to allege that our BURST with multi-period strategy is outstanding among the state-of-the-arts due to its good performance in both retrieval and localization. Furthermore, by grafting other BoF descriptors into the DC component should be a feasible way to achieve better performance. On the other hand, the AC components in BURST can be annexed to any other video descriptors to provide localization ability and better retrieval performance.

TABLE 6.4: Retrieval performance of BURST compared with other state-of-the-art methods, the dimension d of frame descriptors is set as 1,024.

Method	Localization	Feature	avg-mAP	avg-mAP (DoN)
MMV [23]	✗	MVLAD	0.334	-
SHP [9]	✗	MVLAD	0.363	0.440
CTE [23]	✓	MVLAD	0.352	-
MMV+CTE	✓	MVLAD	0.376	-
TE	✓	MVLAD	0.335	0.413
BURST	✓	MVLAD	**0.383**	**0.453**
Sum-aggregated	✗	ResNet-50	0.467	0.551
CGA [10]	✗	ResNet-50	**0.512**	-
	✗	AlexNet+ResNet-50	**0.523**	**0.601**
TE	✓	ResNet-50	0.461	0.558
BURST	✓	ResNet-50	**0.496**	**0.573**

Chapter 7

Conclusion

7.1 Summary

For localizing relative time offsets, the information of each frame are embedded into the video descriptors. However, such a video descriptor contains too much redundant information will result in too much noise in frame-wise similarities, which is responsible for the false matches and the unsatisfactory retrieval performance. It seems to be a paradox that the functionality of temporal offset localization and a good retrieval performance cannot be obtained at the same time. In this thesis, we showed the possibility that there are generic methods for designing a temporal matching kernel with a good retrieval performance, and a good localization performance as well.

Starting from the purpose to rule out false matches, we focused on penalizing the noise in frame-wise similarities. Both stability-sensitive filter and burst-survive temporal matching kernel are designed for this purpose. The multi-period strategy is not directly designed for improving the retrieval performance, but achieves a better localization performance which improves the retrieval performance in return. We also showed the principle of choosing periods. The combination of BURST and the multi-period strategy achieved significant improvement on retrieval performance with trivial extra computational cost. We also showed that our proposed method is competitive with other state-of-the-arts.

7.2 Future Works

Sooth to say, the work proposed in this thesis is in its completed shape. I may not further dig into this method in the future, but I will be pleasant to apply this method into other applications. Although the temporal matching kernel is solving a 1d matching problem along the temporal axis, some of its idea is hopefully to be reused in a 2d problem, *e.g.* embedding the spatial information of objects on an image. I believe that our method should be compatible with semantic-spatial image search topics.

Bibliography

[1] Arnon Amir et al. "IBM research TRECVID-2003 video retrieval system". In: *NIST TRECVID-2003* 7.8 (2003), p. 36.

[2] Relja Arandjelovic and Andrew Zisserman. "All about VLAD". In: *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE. 2013, pp. 1578–1585.

[3] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. "Surf: Speeded up robust features". In: *European conference on computer vision*. Springer. 2006, pp. 404–417.

[4] Ondrej Chum et al. "Total recall: Automatic query expansion with a generative feature model for object retrieval". In: *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. IEEE. 2007, pp. 1–8.

[5] Jeffrey Dalton, James Allan, and Pranav Mirajkar. "Zero-shot video retrieval using content and concepts". In: *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. ACM. 2013, pp. 1857–1860.

[6] Jeff Donahue et al. "Decaf: A deep convolutional activation feature for generic visual recognition". In: *International conference on machine learning*. 2014, pp. 647–655.

[7] Matthijs Douze, Hervé Jégou, and Cordelia Schmid. "An image-based approach to video copy detection with spatio-temporal post-filtering". In: *IEEE Transactions on Multimedia* 12.4 (2010), pp. 257–266.

[8] Matthijs Douze et al. "Compact video description for copy detection with precise temporal alignment". In: *European Conference on Computer Vision*. Springer. 2010, pp. 522–535.

[9] Matthijs Douze et al. "Stable hyper-pooling and query expansion for event detection". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2013, pp. 1825–1832.

[10] Zhanning Gao et al. "ER3: A Unified Framework for Event Retrieval, Recognition and Recounting". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 2253–2262.

[11] Yunchao Gong et al. "Multi-scale orderless pooling of deep convolutional activation features". In: *European conference on computer vision*. Springer. 2014, pp. 392–407.

[12] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[13] Weiming Hu et al. "A survey on visual content-based video indexing and retrieval". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 41.6 (2011), pp. 797–819.

[14] Hervé Jégou and Ondřej Chum. "Negative evidences and co-occurences in image retrieval: The benefit of PCA and whitening". In: *Computer Vision–ECCV 2012*. Springer, 2012, pp. 774–787.

[15] Hervé Jégou et al. "Aggregating local descriptors into a compact image representation". In: *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE. 2010, pp. 3304–3311.

[16] Alexandre Karpenko and Parham Aarabi. "Tiny videos: A large data set for nonparametric video retrieval and frame classification". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.3 (2011), pp. 618–630.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[18] David G Lowe. "Object recognition from local scale-invariant features". In: *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*. Vol. 2. Ieee. 1999, pp. 1150–1157.

[19] Maxime Oquab et al. "Learning and transferring mid-level image representations using convolutional neural networks". In: *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE. 2014, pp. 1717–1724.

[20] Florent Perronnin and Christopher Dance. "Fisher kernels on visual vocabularies for image categorization". In: *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*. IEEE. 2007, pp. 1–8.

[21] Sébastien Poullot et al. "Temporal matching kernel with explicit feature maps". In: *Proceedings of the 23rd ACM international conference on Multimedia*. ACM. 2015, pp. 381–390.

[22] Ali Sharif Razavian et al. "CNN features off-the-shelf: an astounding baseline for recognition". In: *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*. IEEE. 2014, pp. 512–519.

[23] Jérôme Revaud et al. "Event retrieval in large video collections with circulant temporal encoding". In: *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE. 2013, pp. 2459–2466.

[24] Heng Tao Shen et al. "UQLIPS: a real-time near-duplicate video clip detection system". In: *Proceedings of the 33rd international conference on Very large data bases*. VLDB Endowment. 2007, pp. 1374–1377.

[25] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[26] Josef Sivic and Andrew Zisserman. "Video Google: A text retrieval approach to object matching in videos". In: *null*. IEEE. 2003, p. 1470.

[27] Josef Sivic and Andrew Zisserman. "Video Google: Efficient visual search of videos". In: *Toward category-level object recognition*. Springer, 2006, pp. 127–144.

[28] Alan F Smeaton, Paul Over, and Wessel Kraaij. "Evaluation campaigns and TRECVid". In: *Proceedings of the 8th ACM international workshop on Multimedia information retrieval*. ACM. 2006, pp. 321–330.

[29] Jingkuan Song et al. "Multiple feature hashing for real-time large scale near-duplicate video retrieval". In: *Proceedings of the 19th ACM international conference on Multimedia*. ACM. 2011, pp. 423–432.

[30] Giorgos Tolias, Teddy Furon, and Hervé Jégou. "Orientation covariant aggregation of local descriptors with embeddings". In: *European Conference on Computer Vision*. Springer. 2014, pp. 382–397.

[31] Andrea Vedaldi and Andrew Zisserman. "Efficient additive kernels via explicit feature maps". In: *IEEE transactions on pattern analysis and machine intelligence* 34.3 (2012), pp. 480–492.

[32] Haojin Yang and Christoph Meinel. "Content based lecture video retrieval using speech and video text information". In: *IEEE Transactions on Learning Technologies* 7.2 (2014), pp. 142–154.

[33] Mei-Chen Yeh and Kwang-Ting Cheng. "Video copy detection by fast sequence matching". In: *Proceedings of the ACM International Conference on Image and Video Retrieval*. ACM. 2009, p. 45.

[34] Shengxin Zha et al. "Exploiting Image-trained CNN Architectures for Unconstrained Video Classification". In: *Proceedings of the British Machine Vision Conference 2015*. British Machine Vision Association, 2015, pp. 60.1–60.13. arXiv: 1503.04144.

Publications

[1] Fan Yang, Sébastien Poullot, and Shin'ichi Satoh. " バースト検出能力を持つフィルタを埋め込んだ時間的な照合カーネルによる 映像検索 ", In: 画像の認識・理解シンポジウム (MIRU), 2017.

[2] Fan Yang, Sébastien Poullot, and Shin'ichi Satoh. "Temporal Matching Kernel with Embedded Stability-Sensitive Filter", In: IEEE International Symposium on Multimedia (ISM), pp. 278-283, 2017.

[3] Junfu Pu, Yusuke Matsui, Fan Yang and Shin'ichi Satoh. "Energy Based Fast Event Retrieval in Video with Temporal Match Kernel", In: International Conference on Image Processing (ICIP), 2017.