# ENERGY BASED FAST EVENT RETRIEVAL IN VIDEO WITH TEMPORAL MATCH KERNEL

*Junfu Pu*[⋆] *Yusuke Matsui*[†] *Fan Yang*[‡†] *Shin'ichi Satoh*[†‡]

[⋆] University of Science and Technology of China, Hefei, Anhui, China
[†] National Institute of Informatics, Tokyo, Japan
[‡] The University of Tokyo, Tokyo, Japan

## ABSTRACT

We propose a fast event retrieval method in video databases based on temporal match kernel. The basic idea of the baseline is to maximize the score function for all possible relative timestamps. However, considering the stability and computational cost, we simplify the similarity score by calculating the energy of the score function. In this way, maximizing the score function which is time-consuming and sensitive to noise is avoided. We derive the simplified energy formulation by using Parseval's theorem, which describes the unitarity of Fourier transform. Further, by our formulation, the problem becomes a simple nearest neighbor search, and we can use product quantization (PQ) to accelerate the computation. We evaluate our method on EVVE dataset for event retrieval. The experimental results show that our approach is much faster than the baseline, with a higher mAP as well. Comparison with state of the art demonstrates the efficacy of our approach.

***Index Terms***— Event retrieval, temporal match kernel

## 1. INTRODUCTION

This paper introduces an approach for fast content-based search in large video databases, which is fundamentally related to video copy detection [1, 2] and particular event retrieval [3]. Previous research shows this topic is difficult due to some intrinsic characteristics of videos and high computational complexity. In order to reduce computational cost, the common choices are twofold, pooling a representative video descriptor out of a frame descriptor set, or selectively choosing keyframes to extract features, which stand for BOF method [4] and keyframe method [5, 6] respectively. However, one big drawback of these approaches is that the temporal relations between frames will be lost.

More recently, circulant temporal encoding (CTE) [3] and temporal embedding (TE) [7] provide new possibilities to keep temporal continuity in search while the complexity is relatively reasonable. These approaches take benefit of compact frame descriptors such as Fisher vector [8] and MVLAD feature vectors [9] in EVVE dataset [3]. With dimensionality reduction, these feature vectors could lead to lower cost.

In CTE, a filter is applied on each video, or frame sequence, to avoid self-similarity. When a video is queried by itself, the response should be standardized to a Dirac delta output. However, the timestamp on each frame must be fixed implicitly. While in TE, each frame is embedded with a timestamp, which could be handily controlled through explicit feature maps. Whereas, TE is still not efficient enough for evaluating similarity score on every time offset.

In this paper, we propose a fast event retrieval method based on TE [7]. We generalize a formulation to accelerate the current algorithm. Instead of maximizing the score function of temporal match kernel directly as done in TE, we use the energy of the function as a similarity. The energy is easily calculated due to the property of Fourier series. By our formulation, the problem becomes a nearest neighbor search. Hence, the search can be further accelerated by PQ. The experimental results on EVVE dataset demonstrate that our approach achieves the state-of-the-art performance.

## 2. BACKGROUND: TEMPORAL MATCH KERNEL

In this section, we briefly introduce explicit feature maps [10, 11]. The idea is to use the inner product in the embedded space as the approximation of the kernel in the original space. With explicit feature maps, Poullot et al. [7] define a class of temporal kernel between frame descriptors for event retrieval.

### 2.1. Matching with Time Offset

Considering two temporal sequences $\mathbf{x} = (\boldsymbol{x}_0, ..., \boldsymbol{x}_t, ...)$ and $\mathbf{y} = (\boldsymbol{y}_0, ..., \boldsymbol{y}_{t'}, ...)$ with a time offset $\Delta$, $\boldsymbol{x}_i \in \mathbb{R}^D$ is the descriptor for the $i$th frame in video $\mathbf{x}$. The descriptor is 4096D VLAD feature in this paper. We get a kernel defined with $\mathbf{x}$, $\mathbf{y}$, and $\Delta$ according to [7] as

$$\begin{aligned}\mathcal{K}_\Delta(\mathbf{x}, \mathbf{y}) &\propto \sum_{t=0}^{\infty} \boldsymbol{x}_t^T \boldsymbol{y}_{t+\Delta} \\ &= \underbrace{\big(\sum_{t=0}^{\infty} \boldsymbol{x}_t \otimes \boldsymbol{\varphi}(t)\big)^T}_{\boldsymbol{\psi}_0(\mathbf{x})} \underbrace{\big(\sum_{t'=0}^{\infty} \boldsymbol{y}_{t'} \otimes \boldsymbol{\varphi}(t' + \Delta)\big)}_{\boldsymbol{\psi}_\Delta(\mathbf{y})},\end{aligned} \tag{1}$$

where $\boldsymbol{\varphi}(t)$ is

$$\boldsymbol{\varphi}(t) = \begin{bmatrix} \sqrt{a_0} \\ \sqrt{a_1}\cos(\frac{2\pi}{T}t) \\ \sqrt{a_1}\sin(\frac{2\pi}{T}t) \\ \vdots \\ \sqrt{a_m}\cos(\frac{2\pi}{T}mt) \\ \sqrt{a_m}\sin(\frac{2\pi}{T}mt) \end{bmatrix}. \tag{2}$$

The kernel $\mathcal{K}_\Delta(\mathbf{x}, \mathbf{y})$ measures the similarity between $\mathbf{x}$ and $\mathbf{y}$ with $\Delta$ time offset. In formulae (1), $\boldsymbol{\psi}_0(\mathbf{x})$ is the representation of $\mathbf{x}$, with a dimension of $(2m+1) \times D$, where $m$ is the number of frequencies remained in Fourier approximation. For convenience, we write $\boldsymbol{\psi}_0(\mathbf{x})$ as

$$\boldsymbol{\psi}_0(\mathbf{x}) = \left[\mathbf{V}_0^T, \mathbf{V}_{1,c}^T, \mathbf{V}_{1,s}^T, ..., \mathbf{V}_{m,c}^T, \mathbf{V}_{m,s}^T\right]^T, \tag{3}$$

where

$$\begin{aligned} \mathbf{V}_0 &= a_0 \sum_{t=0}^{\infty} \boldsymbol{x}_t \in \mathbb{R}^D, \\ \mathbf{V}_{i,c} &= a_i \sum_{t=0}^{\infty} \boldsymbol{x}_t \cos\left(\frac{2\pi}{T}it\right) \in \mathbb{R}^D, \\ \mathbf{V}_{i,s} &= a_i \sum_{t=0}^{\infty} \boldsymbol{x}_t \sin\left(\frac{2\pi}{T}it\right) \in \mathbb{R}^D. \end{aligned} \tag{4}$$

Generally, we should have to calculate $\boldsymbol{\psi}_\Delta(\mathbf{y})$ for all different $\Delta$, which is very time consuming and unnecessary. In fact, thanks to the good property of trigonometric function and with the definition in equations (3) and (4), we get another form for $\mathcal{K}_\Delta(\mathbf{x}, \mathbf{y})$ as

$$\begin{aligned} \mathcal{K}_{\mathbf{x},\mathbf{y}}(\Delta) &= \langle \mathbf{V}_0^{(\mathbf{x})}, \mathbf{V}_0^{(\mathbf{y})} \rangle \\ &+ \sum_{n=1}^{m} \cos(n\Delta) \left( \langle \mathbf{V}_{n,c}^{(\mathbf{x})}, \mathbf{V}_{n,c}^{(\mathbf{y})} \rangle + \langle \mathbf{V}_{n,s}^{(\mathbf{x})}, \mathbf{V}_{n,s}^{(\mathbf{y})} \rangle \right) \\ &+ \sum_{n=1}^{m} \sin(n\Delta) \left( -\langle \mathbf{V}_{n,c}^{(\mathbf{x})}, \mathbf{V}_{n,s}^{(\mathbf{y})} \rangle + \langle \mathbf{V}_{n,s}^{(\mathbf{x})}, \mathbf{V}_{n,c}^{(\mathbf{y})} \rangle \right). \end{aligned} \tag{5}$$

The kernel $\mathcal{K}_\Delta(\mathbf{x}, \mathbf{y})$ is parameterized by $\Delta$ for two fixing videos $\mathbf{x}$ and $\mathbf{y}$. When $\Delta$ varying from $-\pi$ to $\pi$, we get the value of $\mathcal{K}_\Delta(\mathbf{x}, \mathbf{y})$ for different time offset $\Delta$. By plotting the curve, a peak for matching pair can be found as shown in Fig. 1. Further more, the corresponding $\Delta$ for peak value is the matching time offset. Otherwise, if $\mathbf{x}$ and $\mathbf{y}$ are unrelated videos, there are no matching peak in the curve.

### 2.2. Similarity Score Calculation

Similarity measurement is one of the most important problem in retrieval task. Given two video sequences $\mathbf{x}$ and $\mathbf{y}$, the first step is to obtain $\boldsymbol{\psi}_0(\mathbf{x})$ and $\boldsymbol{\psi}_0(y)$ according to equation (3) and (4). After that, sample $\Delta$ in $[-\pi, \pi]$ and calculate $\mathcal{K}_{\mathbf{x},\mathbf{y}}(\Delta)$ for each $\Delta$. The similarity score $S(\mathbf{x}, \mathbf{y})$ between $\mathbf{x}$, $\mathbf{y}$, and the corresponding matching time offset $t_m$ are defined as

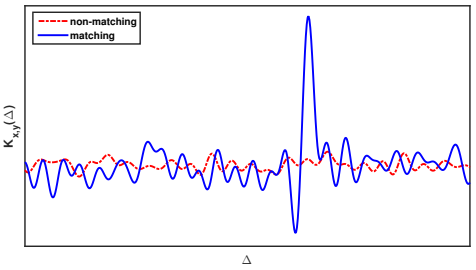


**Fig. 1**: Illustration of $\mathcal{K}_{\mathbf{x},\mathbf{y}}(\Delta)$ for matching pair and non-matching pair [7].

$$S(\mathbf{x}, \mathbf{y}) = \max_{\Delta} \mathcal{K}_{\mathbf{x},\mathbf{y}}(\Delta), \tag{6}$$

$$t_m = \arg\max_{\Delta} \mathcal{K}_{\mathbf{x},\mathbf{y}}(\Delta). \tag{7}$$

This work was originally proposed in [7]. Although it works well in video retrieval and event detection task, it has a high computation complexity and time consuming thanks to the calculation of $S(\mathbf{x}, \mathbf{y})$. In order to calculate $S(\mathbf{x}, \mathbf{y})$, we have to discretize $\Delta$, e.g. $\Delta_i = -\pi + \frac{2\pi}{P}i$ $(i = 1, 2, ..., P)$, where $P$ is the sampling number. The computational cost of $S(\mathbf{x}, \mathbf{y})$ is $O(mDK) + O(K \log K)$, where $D$ is the dimension of $\mathbf{V}_{i,c}$, $K$ is the number of candidates. $O(K \log K)$ is the computational cost for sorting algorithm. Experimentally, $mD \gg K$, so the computational cost is simplified as $O(mDK)$. In addition, the sampling interval for $\Delta$ may have an uncertain influence. And the score $S(\mathbf{x}, \mathbf{y})$ is sensitive to noise, which is not robust while matching. To address these problems, we optimize the formulation of similarity score.

## 3. OUR METHOD

This section introduces the details for matching with energy in our method, with stability and few time-consuming. Further, we use product quantization [12] for speedup.

### 3.1. Matching with Energy

As is introduced in section 2.1, there is a spike in $\mathcal{K}_{\mathbf{x},\mathbf{y}}(\Delta)$ for matching pair. Otherwise, $\mathcal{K}_{\mathbf{x},\mathbf{y}}(\Delta)$ is small, and there is no distinct spike on the curve for unrelated pair, shown in Figure 1. The purpose is to find the peak of this plot, and decide the highest peak among all videos. However, the direct computation of $S(\mathbf{x}, \mathbf{y})$ takes time, with a computational cost of $O(mDK)$. Hence, we do not try to find the peak. Instead, we focus on the energy (area under the curve) of this plot.

Suppose we have a query video $\mathbf{x}$ and two target video $\mathbf{y}_1, \mathbf{y}_2$. We empirically found that, if $\mathbf{y}_1$ is similar to $\mathbf{x}$ than $\mathbf{y}_2$ (the peak value of $\mathcal{K}_{\mathbf{x},\mathbf{y}_1}(\Delta)$ is higher than that of $\mathcal{K}_{\mathbf{x},\mathbf{y}_1}(\Delta)$), the energy of $\mathcal{K}_{\mathbf{x},\mathbf{y}_1}(\Delta)$ is also usually higher than that of $\mathcal{K}_{\mathbf{x},\mathbf{y}_2}(\Delta)$:

$$E(\mathcal{K}_{\mathbf{x},\mathbf{y}_1}) > E(\mathcal{K}_{\mathbf{x},\mathbf{y}_2}) \quad \text{if } S(\mathbf{x}, \mathbf{y}_1) > S(\mathbf{x}, \mathbf{y}_2). \tag{8}$$

From this empirical fact, we modify the score function from $S(\mathbf{x}, \mathbf{y})$ (finding the peak value of the plot) to the energy (the area under the plot) as

$$\widetilde{S}(\mathbf{x}, \mathbf{y}) = E(\mathcal{K}_{\mathbf{x},\mathbf{y}}(\Delta)). \tag{9}$$

Noticing that $\mathcal{K}_{\mathbf{x},\mathbf{y}}(\Delta)$ in formula (5) is the Fourier series. For convenience, denote the Fourier series of $f(x)$ as

$$f(x) = \frac{1}{2}c_0 + \sum_{n=1}^{m} c_n \cos(nx) + \sum_{n=1}^{m} s_n \sin(nx), \tag{10}$$

where $c_i (i = 0, 1, 2, ..., m)$ and $s_i (i = 1, 2, ..., m)$ are the Fourier coefficients kept for Fourier approximation. The energy of $f(x)$ is

$$E(f(x)) = \int_{-\infty}^{\infty} [f(x)]^2 \, dx. \tag{11}$$

According to the Parseval's Theorem [13],

$$\frac{1}{2\pi} \int_{-\infty}^{\infty} [f(x)]^2 \, dx = \sum_{n=1}^{n} \left(c_n^2 + s_n^2\right) + c_0^2, \tag{12}$$

formula (11) is written as

$$E(f(x)) \propto \sum_{n=1}^{m} \left(c_n^2 + s_n^2\right) + c_0^2. \tag{13}$$

In this case, we only focus on the variance, so we just ignore the DC coefficient. Hence, we get

$$E(f(x)) \approx \sum_{n=1}^{m} \left(c_n^2 + s_n^2\right). \tag{14}$$

Using formulae (3) (4) (5) (9), we get

$$c_0 = \langle \mathbf{V}_0^{(\mathbf{x})}, \mathbf{V}_0^{(\mathbf{y})} \rangle \quad \text{(ignored)}, \tag{15}$$

$$c_n = \langle \mathbf{V}_{n,c}^{(\mathbf{x})}, \mathbf{V}_{n,c}^{(\mathbf{y})} \rangle + \langle \mathbf{V}_{n,s}^{(\mathbf{x})}, \mathbf{V}_{n,s}^{(\mathbf{y})} \rangle, \tag{16}$$

$$s_n = -\langle \mathbf{V}_{n,c}^{(\mathbf{x})}, \mathbf{V}_{n,s}^{(\mathbf{y})} \rangle + \langle \mathbf{V}_{n,s}^{(\mathbf{x})}, \mathbf{V}_{n,c}^{(\mathbf{y})} \rangle, \tag{17}$$

where $n = 1, 2, ..., m$. The final form of the energy $\widetilde{S}(\mathbf{x}, \mathbf{y})$ for $\mathcal{K}_{\mathbf{x},\mathbf{y}}(\Delta)$ is

$$\begin{aligned} \widetilde{S}(\mathbf{x}, \mathbf{y}) &= E(\mathcal{K}_{\mathbf{x},\mathbf{y}}(\Delta)) \\ &\approx \sum_{n=1}^{m} \left[ \left( \mathbf{V}_{n,c}^{(\mathbf{x})}, \mathbf{V}_{n,c}^{(\mathbf{y})} \rangle + \langle \mathbf{V}_{n,s}^{(\mathbf{x})}, \mathbf{V}_{n,s}^{(\mathbf{y})} \right)^2 \right. \\ &\quad \left. + \left( \mathbf{V}_{n,c}^{(\mathbf{x})}, \mathbf{V}_{n,s}^{(\mathbf{y})} \rangle + \langle \mathbf{V}_{n,s}^{(\mathbf{x})}, \mathbf{V}_{n,c}^{(\mathbf{y})} \right)^2 \right]. \end{aligned} \tag{18}$$

Given a query video, we need to go through the candidates in the database and calculate the energy $\widetilde{S}(\mathbf{x}, \mathbf{y})$ for each database video. The candidates are ordered by decreasing with this score. Using the energy $\widetilde{S}(\mathbf{x}, \mathbf{y})$, the algorithm performs well not only on the stability and robustness, but less computational complexity as well. There are three advantages with the similarity score in formula (9):

1. The energy ($\widetilde{S}(\mathbf{x}, \mathbf{y})$) is more stable than the maximum of the function ($S(\mathbf{x}, \mathbf{y})$) because the maximum is sensitive to noise.
2. Computing $S(\mathbf{x}, \mathbf{y})$ is time-consuming. We have to sample $\Delta$ and calculate $\mathcal{K}_{\mathbf{x},\mathbf{y}}(\Delta)$ for each $\Delta$. On the other hand, once we get the Fourier series of a function, it is easy to calculate the energy of the function by the Parseval's Theorem. Thus, the computational complexity becomes efficient. Fortunately, $\mathcal{K}_{\mathbf{x},\mathbf{y}}(\Delta)$ in (5) is the standard form of Fourier series. Given $\mathbf{x}$ and $\mathbf{y}$, we are able to calculate the coefficients in (5).
3. By our formulation, the problem becomes a simple nearest neighbor search. Hence we can further accelerate the computation using the approximate nearest neighbor method such as PQ.

With the similar idea, the score can be generalized as

$$S^{(p)}(\mathbf{x}, \mathbf{y}) = \sqrt[p]{\sum_{n=1}^{m} (c_n^2 + s_n^2)^p}, \tag{19}$$

where $p$ is integer and $p \geq 3$. $S^{(p)}(\mathbf{x}, \mathbf{y})$ is parameterized by $p$. When we use the extreme value $\infty$ of $p$, we get

$$\begin{aligned} S^{(\infty)}(\mathbf{x}, \mathbf{y}) &= \lim_{p \to \infty} \frac{1}{M} \sqrt[p]{\sum_{i=1}^{m} (c_n^2 + s_n^2)^p} \\ &= \max_n \{(c_n^2 + s_n^2)\}. \end{aligned} \tag{20}$$

### 3.2. Algorithm Speedup with PQ

The most time consuming part in the algorithm is to calculate the similarity score. Considering the score $\widetilde{S}(\mathbf{x}, \mathbf{y})$ in formula (18), we can speed up this process with product quantization (PQ) [12]. The basic idea of product quantization is to decompose the space into several product of low-dimensional subspace and to quantize each subspace separately. In this case, PQ is implemented to calculate similarity (actually inner product of high dimensional vector).

We use the coarse quantizer to implement an inverted file structure as an array of lists $\mathcal{L}_1, ..., \mathcal{L}_k$. $\mathcal{L}_i$ associated with the centroid stores the set $\{\boldsymbol{y} \in \mathcal{Y} : \boldsymbol{q}_c(\boldsymbol{y}) = \boldsymbol{c}_i\}$, where $\boldsymbol{c}_i$ is the centroid obtained by K-means clustering algorithm [14]. The inverted list $\mathcal{L}_i$ contains a vector identifier and the encoded vector. When we implement K-means algorithm to generate the codebook for coarse quantizer, the similarity metric used here is $\widetilde{S}(\mathbf{x}, \mathbf{y})$. As for the codebooks for subquantizers, the $j$th codebook $\boldsymbol{c}_{j*}$ is generated from $\left\{\mathbf{V}_{j,c}^{(\mathbf{x}_i)} : i \in \{1, \ldots, N\}\right\}$ and $\left\{\mathbf{V}_{j,s}^{(\mathbf{x}_i)} : i \in \{1, \ldots, N\}\right\}$ by K-means, where $N$ is the number of database videos.

Searching steps with product quantization in our task are as follow:

(i) Quantize query $q$ to its $w$ nearest neighbors in the codebook with energy $\widetilde{S}(\mathbf{x}, \mathbf{y})$.

(ii) Compute the squared distance $d_{squ}$ and dot product $d_{dot}$ for each subquantizer $j$ and each of its centroid $\boldsymbol{c}_{ji}$.

(iii) Using the subvector-to-centroid distance computed in step (ii), calculate the similarity score $\widetilde{S}(\mathbf{x}, \mathbf{y})$. It can be finished by looking up in the distance dictionary.

(iv) Order the candidates by decreasing similarity score $\widetilde{S}(\mathbf{x}, \mathbf{y})$.

## 4. EXPERIMENTS

In this section, we provide experimental results for event retrieval to evaluate our approach. The details of database are also introduced in this section.

### 4.1. Dataset

In our experiments, we use the EVVE dataset [3], which is a challenging dataset for event retrieval released by INRIA. The videos in this dataset are collected from YouTube, and annotated in 13 events. The EVVE contains 620 queries and 2375 database videos. Each frame is described with 1024-dimensional multi-VLAD. The details of the EVVE dataset and mVLAD descriptor are introduced in [3].

### 4.2. Experimental Setup and Performance

The method with the similarity score $S(\mathbf{x}, \mathbf{y})$ proposed in [7] is the baseline in our experiments, without any query expansion technic. In our experiments, the modulation period $T$ is set to be 65537. We keep $m = 32$ frequencies in Fourier approximation. As for the parameters in product quantization, there are 64 subquantizers with a dimension of 1024. K-means generate 512 centroids per subquantizer. When building the invert index, the number of coarse quantizers is 128.

First, consider the similarity $S^{(p)}$ in formula (19). Figure 2 gives the mAP for different $p$. As the figure shows, we have a better performance with the generalized formulation by increasing $p$. It works well in event retrieval task.

Table 1 shows the retrieval performance (mAP and time) on EVVE dataset with different methods proposed in this paper. The retrieval algorithm is implemented with Matlab and runs on the same machine, to make sense for time evaluation. We also provide the results with the extreme value of $p$ ($p \to \infty$), where $S^{(p)}(\mathbf{x}, \mathbf{y})$ is written as (20). The mAP using $S^{(\infty)}$ achieves 37.72%, with around 5% increasing compared to the baseline. What's more, with product quantization speedup, our method is 30× faster than the baseline. Table 2 reports our results as well as the state of the art, Circulant Temporal Encoding (CTE), Mean multi-VLAD (MMV) and (Stable Hyper-Pooling) SHP [4]. CTE and MMV are both proposed in [3] for event retrieval. These three method all use the VLAD frame descriptors. MMV is a video descriptor averaging the set of frame descriptors MVLADs. CTE encodes the frame descriptors of a video to jointly represent both appearance and temporal order. From the results, our method outperforms the state of the art, especially with a significant improvement to the baseline.

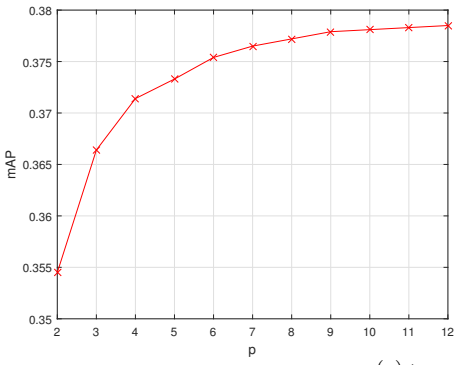


**Fig. 2**: The average mAP using $S^{(p)}(\mathbf{x}, \mathbf{y})$ for different $p$.

**Table 1**: Performance (mAP and time) on EVVE. The bold values show the best score.

| Event No. | Baseline | Ours | | |
|---|---|---|---|---|
| | | $\widetilde{S}$ | $\widetilde{S}$ +PQ | $S^{(\infty)}$ +PQ |
| #1 | 0.1521 | 0.2013 | 0.1985 | **0.2483** |
| #2 | 0.2424 | 0.2503 | **0.2621** | 0.2133 |
| #3 | **0.1186** | 0.1130 | 0.0651 | 0.0905 |
| #4 | 0.1370 | 0.1390 | 0.1419 | **0.1467** |
| #5 | 0.2486 | 0.2538 | **0.2675** | 0.2671 |
| #6 | 0.2913 | 0.3189 | 0.3511 | **0.3917** |
| #7 | **0.1856** | 0.1854 | 0.1177 | 0.1139 |
| #8 | 0.2004 | 0.2216 | 0.2128 | **0.2736** |
| #9 | 0.6119 | 0.6351 | 0.6276 | **0.6728** |
| #10 | 0.3737 | 0.4519 | 0.4913 | **0.5529** |
| #11 | 0.7979 | 0.7879 | **0.8584** | 0.8218 |
| #12 | 0.2295 | 0.3084 | 0.3224 | **0.4344** |
| #13 | 0.6187 | 0.6331 | **0.6915** | 0.6762 |
| ave-mAP | 0.3237 | 0.3461 | 0.3545 | **0.3772** |
| time | 9.31s | 1.74s | $\approx$ **0.3s** | $\approx$ **0.3s** |

**Table 2**: Comparison with state of the art.

| methods | state of the art | | | | Ours |
|---|---|---|---|---|---|
| | MMV | CTE | SHP | MMV+CTE | |
| ave-mAP | 0.334 | 0.352 | 0.363 | 0.376 | **0.377** |

## 5. CONCLUSION

In this paper, we make an improvement to the baseline, which makes it more efficient and less time-consuming. The similarity is associated with function energy, with a simplified formulation. Further, by our formulation, the approximate nearest neighbor method such as PQ is implemented to accelerate the computation. Experimental results shows that our method is much faster than the baseline, and outperforms the state of the art as well.

## 6. REFERENCES


[1] Matthijs Douze, 1, Cordelia Schmid, and Patrick Pérez, "Compact video description for copy detection with precise temporal alignment," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6311 LNCS, no. PART 1, pp. 522–535, 2010.

[2] Julien Law-To, Li Chen, Alexis Joly, Ivan Laptev, Olivier Buisson, Valerie Gouet-Brunet, Nozha Boujemaa, and Fred Stentiford, "Video copy detection: a comparative study," in *ACM International Conference on Image and Video Retrieval*, 2007, pp. 371–378.

[3] Jérôme Revaud, Matthijs Douze, Cordelia Schmid, and Hervé Jégou, "Event retrieval in large video collections with circulant temporal encoding," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2459–2466.

[4] Matthijs Douze, Jérôme Revaud, Cordelia Schmid, and Hervé Jégou, "Stable hyper-pooling and query expansion for event detection," in *IEEE International Conference on Computer Vision*, 2013, pp. 1825–1832.

[5] Alexandre Karpenko and Parham Aarabi, "Tiny videos: a large data set for nonparametric video retrieval and frame classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 618–630, 2011.

[6] Jingkuan Song, Yi Yang, Zi Huang, Heng Tao Shen, and Richang Hong, "Multiple feature hashing for real-time large scale near-duplicate video retrieval," in *ACM International Conference on Multimedia*, 2011, pp. 423–432.

[7] Sébastien Poullot, Shunsuke Tsukatani, Anh Phuong Nguyen, Hervé Jégou, and Shin'Ichi Satoh, "Temporal matching kernel with explicit feature maps," in *ACM International Conference on Multimedia*, 2015, pp. 381–390.

[8] Florent Perronnin and Christopher Dance, ," in *IEEE Conference on Computer Vision and Pattern Recognition*, jun 2007, pp. 1–8.

[9] Hervé Jégou and Ondrej Chum, "Negative Evidences and Co-occurences in Image Retrieval: The Benefit of PCA and Whitening," in *European Conference on Computer Vision*, pp. 774–787. 2012.

[10] Andrea Vedaldi and Andrew Zisserman, "Efficient additive kernels via explicit feature maps," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 480–492, 2012.

[11] Athanasios Papoulis, *Signal analysis*, vol. 191, McGraw-Hill New York, 1977.

[12] Herve Jegou, Matthijs Douze, and Cordelia Schmid, "Product quantization for nearest neighbor search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 117–128, 2011.

[13] Alan V Oppenheim and Ronald W Schafer, "Digital signal processing. 1975," *Englewood Cliffs, New York*.

[14] John A Hartigan and Manchek A Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.