Sequence Kernels for Clustering and Visualizing Near Duplicate Video Segments

Werner Bailer

DIGITAL – Institute for Information and Communication Technologies
JOANNEUM RESEARCH Forschungsgesellschaft mbH, Graz, Austria
werner.bailer@joanneum.at

Abstract. Organizing and visualizing video collections containing a high number of near duplicates is an important problem in film and video post-production. While kernels for matching sequences of feature vectors have been used e.g. for classification of video segments, kernel-based methods have not yet been applied to matching near duplicate video segments. In this paper we survey the application of six sequence-based kernels to clustering near duplicate video segments using kernel k-means and hierarchical clustering, and the application of kernel PCA for generating content visualizations for browsing. Evaluation on the TRECVID 2007 BBC rushes data set shows that the results of the kernel based methods are comparable to other approaches for matching near duplicates, eliminating differences between dynamic time warping and string matching. These results show that hierarchical clustering outperforms kernel k-means. We also show that well-arranged visualizations of both single- and multi-view content sets can be obtained using kernel PCA.

1 Introduction

In this paper we consider the problem of organizing and visualizing video collections containing a high number of near duplicates. Such collections exist for example in the film and video production process, where a large amount of raw material is shot, and a small fraction of it is selected for use in post-production. The material is highly redundant, containing often many takes of the same scene, which are similar, but differ in small details. A substantial amount of literature on the problem of matching and detecting near duplicate video segments exists (for an overview see e.g. [2,16], also fostered by two iterations of the TRECVID [23] rushes summarization task.

Kernels for matching sequences of feature vectors have been proposed and applied to feature sequences from videos for problems such as classifying events or person trajectories. As several of the works on near duplicate detection use sequence-based similarity measures, it seems promising to apply such kernels to collections of near duplicate video segments. Although this seems a logical step, a recent paper [13] seems to be the only work that mentions the use of a sequence-based kernel in a video summarization system.

The rest of this paper is organized as follows. In the remainder of this section we briefly discuss related work on sequence-based kernels and the application of

K. Schoeffmann et al. (Eds.): MMM 2012, LNCS 7131, pp. 383–394, 2012.

kernel k-means and kernel PCA to video content. Section 2 discusses several kernels for sequences of feature vectors and describe their application to clustering video segments using kernel k-means and hierarchical clustering as well as using kernel PCA for projection to a 2-dimensional space for visualization. Section 3 reports experimental results and Section 4 concludes the paper.

Several approaches for sequence matching based on the idea of the pyramid match kernel have been proposed. The original pyramid match kernel [9,11] partitions the feature space in each of the dimensions of the input feature vector. Its efficiency advantage is based on avoiding explicit distance calculations, but only counting elements that end up in the same bin of the pyramid. This assumes that the L_1 distance can be applied to the feature vectors, and no specific distance functions can be used. The vocabulary guided pyramid matching approach proposed in [10] addresses this problem, as it uses a clustering step to construct the pyramid, supporting arbitrary distance measures. The approach has been extended to spatio-temporal matching in [4], using sets of clustered SIFT and optical flow features as local descriptors. Their approach is similar to spatial pyramid matching proposed in [12], which applies the pyramid matching only to the image space (i.e., subdividing an image into a spatial pyramid, and counting features of the same type in each of the bins), but uses clustering in the feature space (i.e., the common bag of words approach).

Another temporal matching method based on the pyramid match kernel is described in [24,25]. Temporally constrained hierarchical agglomerative clustering is performed to build a structure of temporal segments. The pyramid match approach is applied to the decision values of different SVMs instead of the features. The similarity between segments is determined using the earth mover's distance and the pyramid match kernel is applied to the similarities on the different hierarchy levels. This approach explicitly assumes that the temporal order of the individual subclips is irrelevant (as is e.g. the case for news stories). Then the temporal order within the clips is aligned using linear programming.

Kernels for sequences based on dynamic time warping (DTW) [14] have been proposed. The dynamic time alignment kernel (DTAK) proposed in [22] is one of them. Instead of only considering the kernel values along the optimal DTW alignment, the time series alignment kernel proposed in [5] considers the values along all possible paths in DTW alignment.

The authors of [26] use the Levenshtein distance between sequences of clustered local descriptors for classification of still images. Recently, a kernel for matching sequences of histograms of visual words has been proposed [3]. The authors consider different similarity measures between the histograms and use them instead of symbol equality in the Needleman-Wunsch distance. The result of sequence matching is then plugged into a Gaussian kernel. In [1] a kernel based on longest common subsequence (LCSS) matching of sequences has been proposed. An arbitrary kernel can be plugged in to determine the similarity between two elements of the sequences, and the kernel value is determined as the normalized sum of the similarities along the backtracked longest common sequences.

While methods such as kernel k-means and kernel PCA have been used for features derived from video sequences (e.g., pedestrian trajectories [18]), there is little work applying these methods to matching and organizing near duplicate video content. An approach for unsupervised summarization of rushes video is proposed in [13]. It uses a technique called constrained aligned cluster analysis, for both segmentation of the input video and clustering, which is based on kernel k-means and the dynamic time alignment kernel (DTAK) [22]. Unfortunately the authors do not provide objective evaluation results for clustering repeated takes, but only an example for one video.

The contributions of this paper are the following. As only the DTAK kernel has been applied the clustering near duplicate video content, we consider also other types of sequence-based kernels in order to compare their applicability to this problem. As the distance function based on string matching clearly outperforms the one based on DTW in the experiments reported in [2], we are interested whether the same holds for the kernels based on each of these approaches. In addition to using kernel k-means for clustering, we also investigate the use of hierarchical clustering based on the kernel matrix of the sequences. Finally, we use kernel PCA to project a collection of video segments to a 2-dimensional space for visualization purposes based on the similarity of the sequences over time. To the best of our knowledge, this has not been proposed before.

2 Kernel-Based Clustering and Visualization

In this work we aim at finding similar video segments S from a set originating from a single video or a collection of videos. A segment is represented by samples s_i (e.g., frames, key frames). Each of these samples is described by a feature vector x_i, consisting of arbitrary features of this sample (or a temporal segment around this sample). In order to represent the video segment, we concatenate the individual feature vectors to form a feature vector $X = (x_1, \ldots, x_m)$ of the segment. Clearly, not every segment has the same length and/or consists of the same number of samples, thus the lengths of the feature vectors of different segments will differ. We thus need to be able to determine the similarity between such feature vectors having different lengths.

In this section, we first analyze some kernels, which can be applied to the problem of matching such feature vectors. We then discuss how these kernels can be applied to clustering near duplicate video segments using kernel k-means and hierarchical clustering, as well as to projecting video sequences to a 2-dimensional space for visualization.

2.1 Candidate Sequence Kernels

In the following, we review six kernels for sequences of feature vectors with varying lengths and harmonize the formulations of the kernel functions. We denote

as $X = (x_1, \ldots, x_m)$ and $Y = (y_1, \ldots, y_n)$ the sequences of feature vectors of two segments. The term sequence denotes a possibly non-contiguous subsequence. As we intend to support arbitrary ground distances between the feature vectors of the input samples, we use a kernel for matching the feature vectors of elements of the feature sequences, denoted as $\kappa_f(x_i, y_j)$.

Earth Mover's Distance. An advantage of the EMD is that it can applied to different ground distances [19]. We use EMD in a similar way as applied in [24], but we do not use the proposed temporal alignment (TPAM), as it actually does sequence alignment, which is similar to the methods discussed below. We define a kernel using the EMD, replacing the ground distance d_{ij} with $\kappa_f(x_i, y_j)$, as

$$\kappa_{\mathrm{EMD}}(X, Y) = -\frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\widehat{f_{ij}}(-\kappa_f(x_i, y_j))}{\sum_{i=1}^{m}\sum_{j=1}^{n}\widehat{f_{ij}}}, \tag{1}$$

where $\widehat{f_{ij}}$ is the optimal flow determined as

$$\begin{gathered}\widehat{f_{ij}} = -\arg\min_{f_{ij}} \sum_{i=1}^{m}\sum_{j=1}^{n}(-\kappa_f(x_i, y_j))f_{ij}, \\ \sum_{j=1}^{n} f_{ij} \leq w_{x_i}, 1 \leq i \leq m, \text{ and} \\ \sum_{i=1}^{m} f_{ij} \leq w_{y_j}, 1 \leq j \leq n.\end{gathered} \tag{2}$$

The the weights of samples x_i and y_j are chosen as $w_{x_i} = 1/m$ and $w_{y_j} = 1/n$ respectively. Under this condition the formulation is equivalent to the Earth Mover's Similarity proposed in [17].

Temporal Pyramid Match. In order not to constrain the choice of distances in the feature space, pyramid matching can only be applied to the temporal domain, in a similar way as proposed for spatial [12] or spatio-temporal [4] pyramid matching. As we do not perform clustering of the feature vectors in advance, we define a threshold θ to determine whether two feature vectors match or not. The temporal pyramid match kernel is then defined as

$$\kappa_{\mathrm{TPM}}(X, Y) = \sum_{l=1}^{L} \frac{1}{2^{L-l+1}} \Gamma^l + \frac{1}{2^L}\Gamma^0, \tag{3}$$

where L is the number of pyramid levels ($L = \lceil \log_2 \max(|X|, |Y|) \rceil$) and Γ^l is the number of elements matching on level l, i.e., the number of elements falling into the same temporal bin on level l for which $\kappa_f(x_i, y_j) \geq \theta$.

Dynamic Time Alignment. The dynamic time alignment kernel (DTAK) proposed in [22] is based on the dynamic time warping (DTW) approach for sequence alignment [14]. DTW tries to align the samples of the sequences so that the temporal order is kept, but the distance (i.e., the sum of the distances of aligned elements) is globally minimized. Each sample of one sequence is aligned with one or more samples from the other sequence. Let $\psi_x(k)$ be the alignment

function, with $1 \leq \psi_x(k) \leq \psi_x(k+1) \leq |X|$. In addition, a local continuity constraint γ can be defined, s.t. $\psi_x(k+1) - \psi_x(k) \leq \gamma$. Then DTAK is defined as

$$\kappa_{\text{DTAK}}(X,Y) = \max_{\psi_x,\psi_y} \frac{1}{\sum_{k=1}^{N} m(k)} m(k) \kappa_f(x_{\psi_x(k)}, y_{\psi_y(k)}), \tag{4}$$

where $N = \max(|X|,|Y|)$ and $m(k)$ is a weighting coefficient. The kernel can be defined recursively and efficiently implemented using dynamic programming.

Weighted All Subsequences. The all subsequences kernel [21] is defined as $\kappa_{\text{ASS}}(X,Y) = \sum_{\sigma \in \Sigma^*} \phi_\sigma(X)\phi_\sigma(Y)$, where σ denotes a sequence from the possible set of sequences Σ^*, and $\phi_\sigma(X)$ counts the number of times σ occurs as a subsequence of X. Clearly, $\phi_\sigma(X)\phi_\sigma(Y)$ is only non-zero, if σ is a subsequence of both X and Y. Thus a dynamic programming approach can be applied to determine the set of *common* subsequences. The approach is based on the observation that the kernel can be defined recursively. This kernel assumes sequences of discrete values, which does not generally hold for feature vectors. Thus we introduce a threshold θ and consider elements in the sequence as matching, iff $\kappa_f(x_i, y_j) \geq \theta$. The kernel is then defined as

$$\begin{array}{l} \kappa_{\text{ASS}}(X, \emptyset) = 1, \\ \kappa_{\text{ASS}}((x_1, \ldots, x_{n-1}), Y) = \kappa_{\text{ASS}}((x_1, \ldots, x_{n-2}), Y) + \\ \sum_{k:\kappa_f(x_{n-1}, y_k) \geq \theta} \kappa_{\text{ASS}}((x_1, \ldots, x_{n-2}), (y_1, \ldots, y_{k-1})), \end{array} \tag{5}$$

where $\emptyset$ denotes the empty sequence. The kernel value is normalized by the possible maximum number of common sequences of X and Y. In addition, we want to weight the result by the similarities of the matching elements. This can be done by summing $\kappa_f(x_i, y_j)$ for all elements for which $\kappa_f(x_i, y_j) \geq \theta$ and normalizing.

Longest Common Subsequence. Kernels based on the longest common subsequence (LCSS) algorithm have been proposed in [3,1]. The kernel described in [1] already allows plugging in any kernel for measuring the distance between the feature vectors of the samples of the two sequences, and includes the similarities in the result of the kernel. The kernel uses a recursive definition of LCSS and a threshold θ to decide if two feature vectors are considered as matching.

$$\text{LCSS}(X,Y) = \begin{cases} 0, & \text{if } |X| = 0 \vee |Y| = 0, \\ \kappa_f(x_{|X|}, y_{|Y|}) + & \\ \quad \text{LCSS}(\text{Head}(X), \text{Head}(Y)), & \text{if } \kappa_f(x_{|X|}, y_{|Y|}) \geq \theta, \\ \max(\text{LCSS}(\text{Head}(X), Y), & \\ \quad \text{LCSS}(X, \text{Head}(Y))) & \text{otherwise,} \end{cases} \tag{6}$$

where θ is a threshold to consider two feature vectors as matching and $\text{Head}(X) = (x_1, \ldots, x_{|X|-1})$. The kernel function to determine the length of the single longest common subsequence is given as $\kappa_{\text{LCSS}} = \text{LCSS}(X,Y)$. Similarity weighting can be achieved by performing backtracking of the longest sequence, summing the values of $\kappa_f(\cdot)$ of the matches and normalizing.

All Longest Common Subsequences. In [1] the authors propose to consider all subsequences ending in the last element of either of the two sequences:

$$\kappa_{\mathrm{ALCSS}}(X,Y) = \sum_{i=m}^{1} \mathrm{LCSS}((x_1,\ldots,x_i),Y) + \sum_{j=n-1}^{1} \mathrm{LCSS}(X,(y_1,\ldots,y_j)). \tag{7}$$

This requires backtracking of all sequences ending in the last element of either X or Y. The result of the kernel function is normalized to account for sequences of different lengths.

2.2 Kernel-Based Clustering

In this section we discuss the application of the kernels reviewed above for clustering collections of near duplicate video segments.

Kernel *k*-Means. The basic idea of kernel k-means is to apply the well-known k-means algorithm to data points mapped into a high-dimensional feature space. As with other kernel methods, the kernel trick allows performing the required calculations (distance to cluster center, update of cluster center) only by dot products of the mapped data points, thus avoiding the explicit construction of the high-dimensional feature space. In each iteration, the updated cluster index j' of a feature vector X (assuming equal weights for all feature vectors) is given as [6]

$$j'(X) = \mathrm{argmin}_j \left(-2 \sum_{Y \in C_j} \kappa(X,Y) + \sum_{Y,Z \in C_j} \kappa(Y,Z) \right), \tag{8}$$

where C_j is the set of feature vectors in cluster j. An approach based on kernel k-means using the DTAK kernel has been proposed in [13]. Here we generalize this approach and plug in the different types of sequence kernels discussed above.

An issue with k-means is of course the question of the optimal number of clusters. As this question is independent of the use of sequence-based kernels, we do not discuss it here, but refer the reader to the literature.

Hierarchical Clustering. Hierarchical clustering is a common technique to build a cluster structure out of a similarity matrix. Here we use the kernel matrix K of the video segments of the collection as input, i.e., the elements of K are $k_{ij} = \kappa(X_i, Y_j)$. We use the clustering algorithm proposed in [2] for clustering different takes of the scene. It is based on single-linkage clustering, but has an additional constraint to first cluster takes or assign them to scenes before merging scenes. Instead of the number of clusters, this algorithm has a minimum distance parameter which determines when to stop clustering. As several of the kernels use a similarity threshold, we use this threshold θ as the cutoff distance for clustering. This means, that feature vectors can be clustered, if they contain at least one element for which $\kappa_f(\cdot) > \theta$.

2.3 Kernel-Based Visualization

Principal component analysis (PCA) is a well-known method to apply an orthogonal linear transform which projects data into a coordinate system spanned by the principal components. The dimensions of the coordinate system are ordered by decreasing variance of the data. Thus a small number of principal components often approximates the data quite well. Projection of data using PCA to a plane is commonly used for visualization purposes.

In [20] the kernel PCA is introduced, which applies the idea of the PCA to data transformed to a high-dimensional space, using the kernel trick to avoid explicit construction of this space. Instead, the projection to the space spanned by the k first principal components can be determined as

$$P_k(X) = \left(\sum_{i=1}^{l} \alpha_i^j \kappa(X, Y_i) \right)_{j=1}^{k}, \tag{9}$$

where l is the size of the kernel matrix (i.e., in our case, the number of video segments in the collection) and $\alpha^j = (1/\lambda_j)v_j$ is defined from the eigenvectors v_j and eigenvalues λ_j of the kernel matrix. The kernel matrix K contains the mutual kernel values between the video segments of the collection as input, i.e. the elements of K are $k_{ij} = \kappa(X_i, Y_j)$. As the PCA is defined on centered data, a similar step is required for the kernel matrix: $\tilde{K} = K - \frac{1}{l}\mathbf{1}K - \frac{1}{l}K\mathbf{1} + \frac{1}{l^2}\mathbf{1}K\mathbf{1}$, where $\mathbf{1}$ is a matrix of size $l \times l$ with all elements 1.

We aim at using this approach for projecting a collection of video sequences to a low-dimensional space (as an alternative to applying multidimensional scaling to a similarity matrix between the segments). Using the sequence kernels discussed above, the kernel PCA is expected to yield a projection of the data, in which near duplicates are close in the projected space. Such a representation is useful for video browsing and interactive search in video collections containing near duplicate segments.

3 Results

In the following we present results of experiments for clustering repeated takes and visualizing collections of unedited video material.

3.1 Clustering Repeated Takes

The proposed clustering algorithms using the different kernels have been evaluated on a subset of the TRECVID 2007 BBC rushes test data set (the same subset as used e.g. in [2,8]). The subset consists of six randomly selected videos out of this data set (in total 3 hours, for more details see [2]), using the ground truth provided by NHK [15]. In order to avoid side effects from different shot segmentations, the results are based on the ground truth shots. Every 10th frame

Table 1. Mean and median of the frame based F1 measure of clustering results with hierarchical clustering and kernel k-means (with different choices of k) for the six sequence kernels

clustering	**hierarchical**		**kernel k-means**		**kernel k-means**		**kernel k-means**	
methods			k ground truth		k as hierarch.		best $k \in [3;15]$	
	mean	**median**	**mean**	**median**	**mean**	**median**	**mean**	**median**
ALCS	0.50696	0.48305	0.40378	0.41629	0.39730	0.39898	0.49800	0.51472
LCS	0.58775	0.57874	0.45652	0.38854	0.38106	0.37901	0.62065	0.65442
ASS	0.43178	0.41910	0.42903	0.40174	0.37684	0.36764	0.49262	0.47499
EMD	0.49595	0.51785	0.43934	0.44547	0.43277	0.43945	0.51365	0.50151
TPM	0.55394	0.54737	0.40873	0.37693	0.38878	0.37401	0.48079	0.49547
DTAK	0.60153	0.58055	0.48300	0.44685	0.48173	0.47998	0.59541	0.60995

of the videos is used in the feature sequence, and from each frame we extract a feature vector consisting of the MPEG-7 ColorLayout (cl) descriptor (DC and the first two AC coefficients of each channel), the MPEG-7 EdgeHistogram (eh) descriptor and a scalar visual activity (va) value. The kernel function between individual feature vectors is defined as $\kappa_f((dc_X, eh_X, va_X)^T, (dc_Y, eh_Y, va_Y)^T) = \kappa_{MPEG-7}((dc_X, eh_X)^T, (dc_Y, eh_Y)^T)\ \kappa_{RBF}(va_X, va_Y)$, where κ_{MPEG-7} is the MPEG-7 kernel proposed in [7] (with equal weighting of both MPEG-7 features). For the sequence kernels that need a similarity threshold, we set $\theta = 0.03$.

In Table 1 we report the mean and median F1 measure for take clustering. The F1 values are calculated from the frame precision/recall measure proposed for evaluating clustering of repeated takes in [8]. In general, the results are comparable to those of other clustering approaches. An interesting result is that precision and recall are more balanced than in clustering results reported in the literature for clustering with other distance functions. In contrast to the results reported for string matching and DTW based distance functions [2], the LCS or ALCS kernels do not outperform the DTAK kernel. The reason seems to lie in the properties of the kernel $\kappa_f(\cdot)$ between individual feature vectors, which is in the form of $\exp(-\text{distance}(\cdot))$, and thus better distinguishes well matching subsequences than a linear distance measure.

From the string matching kernels, LCS performs better than those considering more than one subsequence (ALCS, ASS). Also EMD and TPM, which do not enforce an ordered sequence, perform similarly well. For TPM, it seems that the temporal tolerance introduced by the pyramid match is sufficient to cover the timing differences and insertions between different takes. The optimal sequence found by EMD often contains many samples in the correct temporal order.

From the clustering methods, hierarchical clustering seems to be the better choice. It yields better results than kernel k-means with the same number of clusters or the number of clusters from the ground truth. We conclude that the reason for this is that the hierarchical clustering algorithm used includes a specific constraint for the take clustering problem. The best kernel k-means results slightly outperform hierarchical clustering result in terms of median, but not in terms of mean.

Fig. 1. Projection of segments from video MRS025913 using kernel PCA (2 principal components) with ALCS kernel (top) and DTAK kernel (bottom)

3.2 Kernel PCA for Visualization

We perform visualization experiments on two data sets: on the videos from the TRECVID 2007 BBC rushes set used for the clustering experiments and on a set of multi-view test material from the 2020 3D Media project[1] (3 views, about 6 minutes). Kernel PCA has been applied and the data has been projected to the plane spanned by the first two principal components. Each video segment is visualized by its first, center and last key frame.

Figure 1 shows the results for one video from the BBC rushes set, using the ALCS and the DTAK kernel. In both visualizations, the proximity is related to the similarity of the sequences. It is difficult to find objective criteria for assessing the quality of the visualizations, especially as the difference in clustering performance between the two kernels is quite small. However, the data seems to be organized more clearly in the visualization produced using the ALCS kernel.

Figure 2 shows the visualization of the multi-view content set using the LCS kernel. The different views are rather spread along the horizontal axis, while the different takes are on the vertical axis (note e.g. the shot with the calibration board being close to the bottom in both cases). Only takes from the one shot with structured light experiments (the dark frames in the middle) are outliers and not well fit into the projection space.

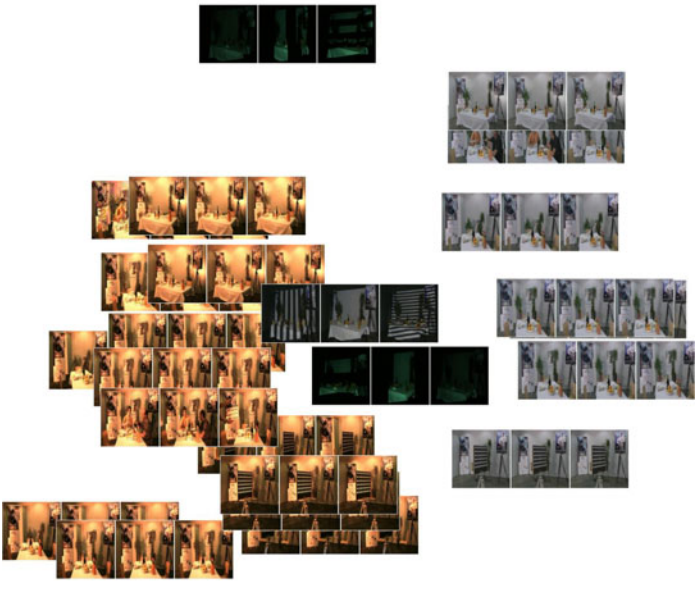

Fig. 2. Projection of segments from the 2020 3D Media multi-view set using kernel PCA (2 principal components) with LCS kernel

[1] http://www.20203dmedia.eu/

4 Conclusion

In this paper we have analyzed the application of six sequence-based kernels for clustering and visualizing collections of near duplicate video segments.

In contrast to previous work using kernel k-means clustering in summarization, we have compared the performance of different kernels and have also used hierarchical clustering on the kernel matrix. No strong differences in the performance of the different kernels have been observed. However, we see that kernels that determine a single best matching sequence perform slightly better than those that weight the results from several matching sequences. Differences in clustering results observed between string matching and dynamic time warping distances are not evident between kernels based on these paradigms. Our results show that hierarchical clustering outperforms kernel k-means in most cases.

We have also shown that meaningful visualizations for interactive browsing and presentation of summaries can be generated using kernel PCA to project the data into a plane. Once the kernel matrix has been calculated, both clustering and visualization can be performed very efficiently. Despite the similar performance of the kernels in clustering, the string matching based kernels (e.g., LCS, ALCS) produce visualizations with a more comprehensible organization of the data. The quality of the obtained visualization needs to be further evaluated in a user study.

Acknowledgments. The author would like to thank Felix Lee for the key frame plotting script, and Hannes Fassold for his support in implementing the kernel PCA. The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 215475, "2020 3D Media – Spatial Sound and Vision".

References

1. Bailer, W.: A Feature Sequence Kernel for Video Concept Classification. In: Lee, K.-T., Tsai, W.-H., Liao, H.-Y.M., Chen, T., Hsieh, J.-W., Tseng, C.-C. (eds.) MMM 2011 Part I. LNCS, vol. 6523, pp. 359–369. Springer, Heidelberg (2011)
2. Bailer, W., Lee, F., Thallinger, G.: A distance measure for repeated takes of one scene. The Visual Computer 25(1), 53–68 (2009)
3. Ballan, L., Bertini, M., Del Bimbo, A., Serra, G.: Video event classification using string kernels. Multimedia Tools Appl. 48(1), 69–87 (2010)
4. Choi, J., Jeon, W.J., Lee, S.-C.: Spatio-temporal pyramid matching for sports videos. In: Proc. 1st ACM International Conference on Multimedia Information Retrieval, pp. 291–297. ACM, New York (2008)
5. Cuturi, M., Vert, J.-P., Birkenes, O., Matsui, T.: A kernel for time series based on global alignments. Computing Research Repository, abs/cs/0610033 (2006)
6. Dhillon, I.S., Guan, Y., Kulis, B.: Kernel k-means: spectral clustering and normalized cuts. In: KDD, pp. 551–556 (2004)
7. Djordjevic, D., Izquierdo, E.: Relevance feedback for image retrieval in structured multi-feature spaces. In: Proc. MobiCom (2006)

8. Dumont, E., Mérialdo, B.: Rushes video parsing using video sequence alignment. In: Proc. CBMI 2009 (June 2009)
9. Grauman, K., Darrell, T.: The pyramid match kernel: Discriminative classification with sets of image features. In: IEEE ICCV, vol. 2 (2005)
10. Grauman, K., Darrell, T.: Approximate correspondences in high dimensions. In: NIPS, pp. 505–512 (2006)
11. Grauman, K., Darrell, T.: The pyramid match kernel: Efficient learning with sets of features. J. Mach. Learn. Res. 8, 725–760 (2007)
12. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: CVPR (2006)
13. Liu, Y., Zhou, F., Liu, W., De La Torre, F., Liu, Y.: Unsupervised summarization of rushes videos. In: Proc. ACM Multimedia, pp. 751–754 (2010)
14. Myers, C.S., Rabiner, L.R.: A comparative study of several dynamic time-warping algorithms for connected word recognition. The Bell System Technical Journal 60(7), 1389–1409 (1981)
15. NHK Science & Technical Research Laboratories. Test modules for TRECVID activity. Use case scenario. Ver.1.2.0E (April 2008)
16. Over, P., Smeaton, A.F., Awad, G.: The TRECVID 2008 BBC rushes summarization evaluation. In: Proceedings of the 2nd ACM TRECVid Video Summarization Workshop, TVS 2008, pp. 1–20. ACM, New York (2008)
17. Rahimi, A., Kiran, R.: How earth mover's distance comprares two bags. Technical report, Intel Labs Berkeley (2007)
18. Ricci, E., Tobia, F., Zen, G.: Learning pedestrian trajectories with kernels. In: ICPR, pp. 149–152 (2010)
19. Rubner, Y., Tomasi, C., Guibas, L.J.: The earth mover's distance as a metric for image retrieval. Int. J. of Computer Vision 40(2), 99–121 (2000)
20. Schölkopf, B., Smola, A., Müller, K.-R.: Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation 10(5) (1998)
21. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge Univ. Press (2004)
22. Shimodaira, H., Noma, K.-I., Nakai, M., Sagayama, S.: Dynamic time-alignment kernel in support vector machine. In: NIPS (2001)
23. Smeaton, A.F., Over, P., Kraaij, W.: Evaluation campaigns and TRECVid. In: Proc. 8th ACM International Workshop on Multimedia Information Retrieval, pp. 321–330 (2006)
24. Xu, D., Chang, S.-F.: Visual event recognition in news video using kernel methods with multi-level temporal alignment. In: IEEE CVPR (2007)
25. Xu, D., Chang, S.-F.: Video event recognition using kernel methods with multilevel temporal alignment. IEEE Trans. Pattern Anal. Mach. Intell. 30 (2008)
26. Yeh, M.-C., Cheng, K.-T.: A string matching approach for visual retrieval and classification. In: Proc. 1st ACM International Conference on Multimedia Information Retrieval, pp. 52–58. ACM, New York (2008)