In Proc. 14th International MultiMedia Modeling Conference (MMM'08), Kyoto, Japan, 2008.

Similarity Search in Multimedia Time Series Data using Amplitude-Level Features

Johannes Aßfalg, Hans-Peter Kriegel, Peer Kröger, Peter Kunath, Alexey Pryakhin, Matthias Renz

Institute for Informatics, Ludwig-Maximilians-Universität München, Germany
{assfalg,kriegel,kroegerp,kunath,pryakhin,renz}@dbs.ifi.lmu.de

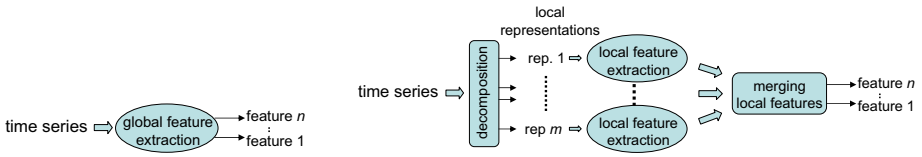
Abstract. Effective similarity search in multi-media time series such as video or audio sequences is important for content-based multi-media retrieval applications. We propose a framework that extracts a sequence of local features from large multi-media time series that reflect the characteristics of the complex structured time series more accurately than global features. In addition, we propose a set of suitable local features that can be derived by our framework. These features are scanned from a time series amplitude-levelwise and are called amplitude-level features. Our experimental evaluation shows that our method models the intuitive similarity of multi-media time series better than existing techniques.

1 Introduction

Time series data is a prevalent data type in multi-media applications such as video or audio content analysis. Videos are usually modeled as sequences of features extracted for each picture of the video stream. Analogously, audio content is also modeled as a series of features extracted continuously from the audio stream. Similarity search in such time series data is very important for multi-media applications such as query-by-humming, plagiarism detection, or content-based audio and video retrieval.

The challenge for similarity search in time series data is twofold. First, the adequate modeling of the intuitive similarity notion between time series is important for the accuracy of the search. For that purpose, several distance measures for time series have been defined recently, each of which works fine under specific assumptions and in different scenarios (e.g. the Euclidian distance or Dynamic Time Warping (DTW)). Most of them apply features comprising quantitative information of the time series. However, in particular for complex structured time series, features comprising quantitative information are often too susceptible to noise, outliers, and other interfering variables. Second, since time series are usually very large, containing several thousands of values per sequence, the comparison of two time series can be very expensive, particularly when using distance measures that require the access to the raw time series data (i.e. the entire sequence of time series values). For example, for a audio sequence we can derive 300 features per second. Thus, a 3 minute audio sequence is represented by a time series of length 54,000. Generally, shape-based similarity measures like the DTW are very expensive, and are usually not applicable for multi-media data.

In this paper, we propose a novel framework for shape-based similarity search on multi-media time series that addresses both mentioned problems. Our approach allows



(a) Global (b) Local

Fig. 1. Global vs. local feature extraction.

to incorporate the relevant features that model qualitative characteristics of the time series and, thus, is able to represent the similarity of complex time series more accurately than recent approaches. In addition, the runtime complexity of our method is independent of the length of the time series which is important for very long multi-media time series. Figure 1 depicts our novel approach of local feature extraction (cf. Figure 1(b)) in comparison to the traditional global feature extraction strategies [1] (cf. Figure 1(a)). The traditional global approach extracts a set of n one-dimensional features representing the global characteristics of the time series. The resulting n features are used to build an n-dimensional feature vector and the Euclidean distance is used to measure the similarity between the derived features. In contrast, our approach is based on a decomposition of the complex structured time series into a reasonable set of more simple structured components which we call local representations. Then, we extract a set of local features of different types from these local representations. Subsequently, we merge the local features of each type into a feature vector. As a result, we obtain a set of n feature vectors. The main advantage of our strategy is that we dissect the complex feature extraction problem into a set of small subproblems which can be solved more easily. In order to reduce the complexity of the similarity measure based on the resulting features, we can subsequently compress the results using standard dimensionality reduction techniques. In this paper, we focus on one-dimensional time series. However, our approach can easily be adapted to the multi-dimensional case by extracting features for each dimension.

The rest of the paper is organized as follows. In Section 2, we survey related work. In Section 3, we present our feature extraction framework. A set of feature types reflecting the characteristics of time series in Section 4. Section 5 comprises the experimental results. The paper is concluded in Section 6.

2 Related Work

Modeling multimedia objects by time series. Recently, modeling multimedia data (e.g., audio streams, video sequences) as time series attracted more and more attention in the field of content-based multimedia object retrieval. In [2], a template matching method is presented. This approach measures similarity of videos by applying the time warping distance on sequences of low-level feature vectors. Another efficient scheme for video

sequence matching was suggested in [3]. This method applies a time-series-based description of visual features from every video frame to capture the visual similarity w.r.t. the order of their appearance in the query video. The authors of [4] propose a clustering process and a weighted similarity measure between so-called key frames calculated for each shot in order to capture the dynamics of visual contents for matching. This key-frame based representation of a video can be viewed as a multi-dimensional time series in order to retain the temporal order of events.

In order to compare two multimedia objects represented by time series, a proper distance measure is required. The most prominent similarity measure for time series is the Euclidean distance. In the past years, the Dynamic Time Warping (DTW) [5] which is conceptually similar to sequence alignment has become as prominent as the Euclidean distance. Contrary to the Euclidean distance which is a measure for the similarity in time, DTW is a measure for the similarity in shape.

Feature extraction for time series. For long time series usually structure level similarity measures based on global features or model parameter extraction are used [6–9]. A similarity model for time series that considers the characteristics of the time series was recently proposed in [1]. A set of global features including periodicity, self-similarity, skewness, kurtosis among others are used to compute the similarity between the time series. Some of the features are generated from the raw time series data as well as from trend and seasonally adjusted time series. The authors focused on clustering as a special application of similarity search and showed that a small set of global features can be sufficient to achieve an adequate clustering quality. However, this approach is successful only as long as adequate features that reflect the time series characteristics can be identified. Unfortunately, long time series often feature very complex structures which cannot sufficiently be reflected by a single global feature, e.g. modeling the periodicity of a long time series with only one value may be too coarse in most cases.

In the multimedia community, publications about global features can be grouped in two main categories. Approaches belonging to the first category calculate features in the so-called frequency domain. The following feature transformations of this category are well known in the audio and signal processing area: Relative Spectral Predictive Linear Coding, Pitch [10], Spectral Flux, Mel Frequency Cepstral Coefficients, Bark Frequency Cepstral Coefficients [11], and coefficients calculated by basic time-frequency transformations (e.g., DCT, FFT, CWT, DWT). The second category consists of techniques that extract features in the so-called time domain. This category consists of the following features: Linear Predictive Coding coefficients[12], Zero Crossing Rate Periodicity Histogram [10], Sone and Short Time Energy [13], Length of High Amplitude Sequence, Length of Low Amplitude Sequence, or Area of High Amplitude [14].

In contrast to the existing features working in the time-based domain, the features proposed in this paper are calculated over the whole amplitude spectrum. This fact allows us capturing time-domain properties over the whole available amplitude range. Moreover, we suggest an automatical method for the combination of the derived features which results in a significant improvement of effectiveness.

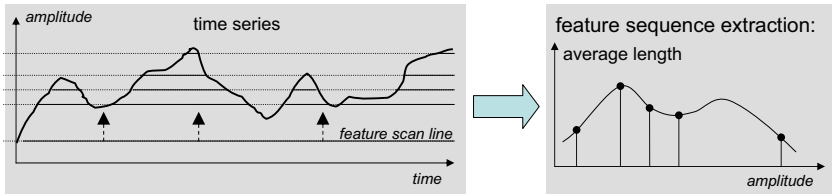


Fig. 2. Amplitude-level-wise feature extraction.

3 Feature Sequence Extraction

As discussed above, traditional similarity measures like the Euclidean distance or DTW that work on the raw time series are not appropriate for multi-media applications. Rather, it is more suitable to extract features from qualitative representations of time series. Here, we propose a qualitative representation of time series that relies on sequences of intervals each representing all time slots at which the value of the original time series is above a given amplitude level.

3.1 Amplitude-Level-Wise Feature Extraction

Using only features according to one single amplitude level to describe a time series might be much too coarse to get satisfying results. Rather, we use several amplitude levels in order to capture the entire shape of the time series and extract features for each amplitude value. Instead of one feature value we get a sequence of feature values, called *feature sequence*, as illustrated in Figure 2. However, it is not very appropriate to consider all possible amplitude values for the feature extraction due to two reasons: First, the number of amplitude levels is infinite. Second, close amplitude values are likely to contain similar information, so that the feature sequence will contain a lot of redundant information.

We propose a framework that extracts time series features in two steps: In a first step, we generate sequences of feature values by scanning the amplitudes of the corresponding time series with a reasonable high resolution. In order to improve the similarity search quality we suggest to extract several features from the interval sequences. As depicted in Figure 2 we use the feature scan line (fsl) to vertically scan the time series from bottom to top and retrieve at each (relevant) amplitude level τ a set of features called *Amplitude-Level Features* (ALFs). Examples of simple ALFs include the fraction of time series values that exceed each amplitude level (denoted by ALF_{ATQ} in the following) and the number of intervals of consecutive time slots above each amplitude level (denoted by ALF_{TIC} in the following).

As a result, we obtain a sequence of ALFs $\langle(\tau_{min}, f_{\tau_{min}}), \ldots (\tau_{max}, f_{\tau_{max}})\rangle$, where τ_{min} denotes the global minimum of all amplitudes of all time series and τ_{max} denotes the corresponding global maximum of all amplitudes. The resolution r of the amplitude scan (i.e. the length of the ALF sequence) is a user-defined parameter that influences the length of the resulting feature sequence as well as the accuracy of the representation. If we choose a high value for the resolution r, we will obviously obtain a more accurate

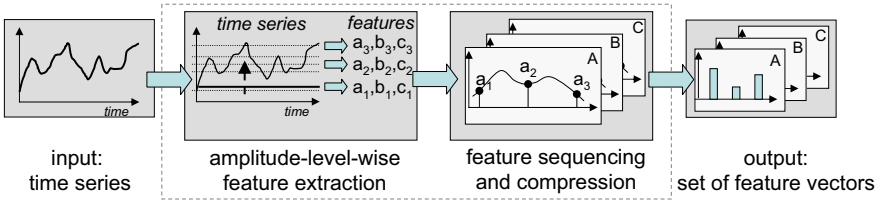


Fig. 3. Feature extraction framework.

description of the time series and may achieve better results. On the other hand, a high value for the resolution r results in larger space required to store the ALF sequences and in a lower query performance. In order to reduce the size of the extracted features and to decrease redundant information, we subsequently apply appropriate dimensionality reduction methods to reduce the large feature sequences to a smaller set of coefficients. These coefficients correspond to the feature vectors that are finally used to represent the time series and are used for the similarity search methods.

3.2 Feature Sequence Compression

Depending on the resolution r of the feature extraction method, the ALF sequences may usually have a more or less smooth shape. Assuming a high resolution, the features extracted from adjacent amplitude thresholds do not vary very much. For this reason, common dimensionality reduction techniques for time series like DFT, PAA, or Chebyshev applied to the ALF sequences lead to shorter ALF sequences while accurately approximating the original ALF sequence. Finally, for each feature we generate a dimensionality reduced ALF sequence in the form of a vector which can be indexed by any spatial index structure. This strategy helps to solve the performance problems while keeping the quality of the similarity measure.

The principle of our framework is depicted in Figure 3. The framework takes a time series as input and produces a set of feature vectors as output. It consists of the two steps, the amplitude-level-wise feature extraction and the feature sequence compression. In particular, for a given time series and a given feature A we extract a sequence of local feature values a_1, a_2, and a_3. Subsequently, the generated ALF sequence is compressed by means of standard dimensionality reduction techniques resulting in a feature vector. This step is repeated for each feature extraction method so that finally a set of feature vectors is returned. This set is afterwards used to measure the similarity between the time series objects. Obviously, the final set of feature vectors and the dimension of each feature depends on the number and type of derived features, the resolution r, and the applied dimensionality reduction techniques. The length of the input time series however, has no influence on the dimensions of the resulting feature vectors. So the time complexity of a query is constant with respect to the length of the input time series. In the following section, we will present some high quality ALFs and discuss how to compute the similarity of the resulting set of feature vectors.

3.3 Feature Sequence Combination

As mentioned above, we generate a set of feature vectors for each time series. As the combination of different feature vectors usually improves the search quality we apply a combination approach similar to the techniques described in [15] and [16]. In order to compute weights for the different representations, we initially chose a small subset of training objects from the database. For each query object, we then perform a k-NN query on our training set and aggregate the number of objects for each class in a confidence vector. The coefficients of this confidence vector reflect the frequency of each class in the k-NN sphere of the query object. After normalization, the weights are derived by computing the entropy for each representation. The idea of this method is that feature spaces yielding a pure k-NN sphere are more suitable than representations containing objects from a lot of classes. After having determined the weights, a standard combination method like *sum*, *product*, *min*, or *max* can be used to combine the distances according to the different feature representations.

4 Amplitude-Level Features

The number and type of adequate features depend on the application and on the data. In the following, we propose a selection of features that mainly reflect shape characteristics of time series, and thus, might be suitable to common applications. Let us note that there may be other features that are sensible for special applications. However, we will show in our experiments that even the basic features described in the following already yield a rather accurate similarity model for shape-based similarity search in time series databases.

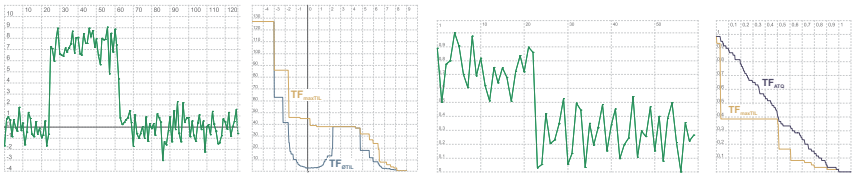
In the previous section, we already introduced two simple amplitude-level features, the Above Amplitude Level Quota (ATQ) that measures the fraction of time series values that exceed a given amplitude level τ and the Threshold Interval Count (TIC) that measures the relative number of amplitude level intervals. Both are easy to compute but reflect well the basic characteristic of time series. The resulting feature values obviously range from 0 to 1, where ALF_{ATQ} decreases monotonously for increasing values of τ. Noise has a stronger impact on ALF_{TIC} curve than on that of ALF_{ATQ} as a lot of noise can lead to a huge number of intervals. Similar to ALF_{ATQ} all chronological information is lost.

In the following, we present further adequate amplitude-level features. Here $ti_{\tau,X}$ denotes the qualitative representation of a time series X w.r.t. the amplitude level τ, i.e. the set of intervals where the value of X is above τ.

Threshold Interval Length (TIL). In contrast to the ALF_{TIC}, TIL tries to capture more complex characteristics of the intervals than just their existence. An obvious choice is the average and the maximal length of all intervals for a given amplitude level τ and a time series X, formally

$$ALF_{maxTIL}(X,\tau) = \max\{(u_j - l_j) : j \in ti_{\tau,X}\}$$

$$ALF_{\varnothing TIL}(X,\tau) = \frac{1}{|ti_{\tau,x}|}\sum_{j=0}^{M}(u_j - l_j)$$



(a) Example ALF_{maxTIL} and $ALF_{\varnothing\,TIL}$

(b) Robustness against noise for ALF_{maxTIL} compared to ALF_{ATQ}

Fig. 4. Examples of ALF sequences.

ALF_{maxTIL} and $ALF_{\varnothing\,TIL}$ show similar behavior in most cases, whereas the former always yields monotonously decreasing feature values and naturally is more robust against noise (cf. Figure 4(a)). The contained information basically indicates at which amplitudes the time series contain high/low frequent sections. The same observation could be made for ALF_{ATQ}, however, for noisy data $ALF_{\varnothing\,TIL}$ is more effective than ALF_{ATQ}(cf. Figure 4(b)).

Threshold Interval Distance (TID). A further reasonable feature is the distance between consecutive intervals. Generally, distances between intervals are ambiguous, so we have some options for the feature generation. We can build distances between the start points of the intervals only or between the end points only or we take both points into account. Here, we use the distance between the start point l_j of an interval and the start point l_{j+1} of the subsequent interval. We again consider both, the maximum and the average distance value:

$$ALF_{maxTID}(X,\tau) = \max\{(l_{j+1} - l_j) : j \in 1..ti_{\tau,X} - 1\}$$

$$ALF_{\varnothing\,TID}(X,\tau) = \frac{1}{|ti_{\tau,X}| - 1} \sum_{j=1}^{|ti_{\tau,X}|-1} (l_{j+1} - l_j)$$

This feature differs from the others in being not invariant against mirroring of the time series along the time axis. It is adequate to separate periodical signals having different frequencies. Like ALF_{TIL}, ALF_{maxTID} is more robust against noise than $ALF_{\varnothing\,TID}$.

Threshold Crossing Angle (TXA). This feature differs slightly from the previous as it does not take the interval sequences into account. Here, we consider the slopes of the time series that occur at the start and end points of the time intervals.

First, we define the slope angle $\mathrm{angle}(t_i)$ of a time series value $(x_i, t_i) \in X : i \in 2 \ldots N$ as follows:

$$\mathrm{angle}(t_i) := \arctan(x_i - x_{i-1}).$$

Based on this definition, we can define the features:

$$ALF_{absTXA}(X,\tau) = \frac{1}{N\pi} \sum_{j=1}^{|ti_{\tau,X}|} (|\text{angle}(l_j)| + |\text{angle}(u_j)|) * (u_j - l_j)$$

$$ALF_{diffTXA}(X,\tau) = 0{,}5 + \frac{1}{N\pi} \sum_{j=1}^{|ti_{\tau,X}|} (\text{angle}(l_j) + \text{angle}(u_j)) * (u_j - l_j)$$

We weight the slope angles according to the corresponding interval length. The factor $\frac{1}{N\pi}$ as well as the constant value 0.5 are used to normalize the results to the range $(0, 1)$ and are necessary to compare time series of different lengths.

Obviously, this feature primarily aims at separating time series having different rate of changes. Unfortunately, the determination of such kind of patterns can be easily perturbed by noise. As a consequence, the quality of the similarity measures based on this feature mainly depends on the intensity of the noise in a dataset.

Threshold Balance (TB). The last feature incorporates the temporal behavior of the time series by considering the distribution of the amplitude values that are above the corresponding amplitude threshold. First, we need the auxiliary function

$$\text{above}_\tau(x_i) := \begin{cases} 1 & \text{if } x_i > \tau \\ 0 & \text{else} \end{cases}.$$

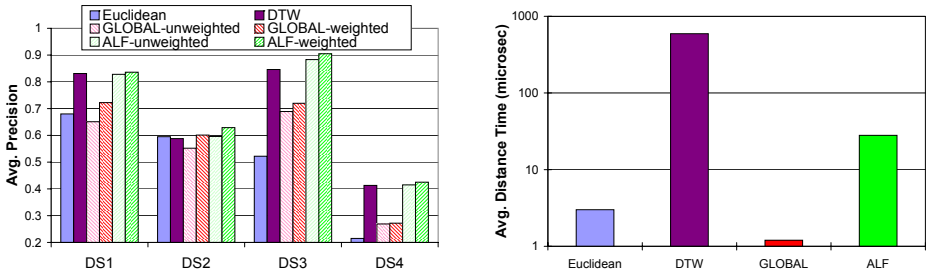
By means of this function, we can define the Threshold-Balance feature ALF_{TB} that aggregates those values of the time series which are above the amplitude threshold, whereas values of different time slots are weighted differently. Formally

$$ALF_{TB}(X,\tau) = \frac{1}{N^2} \sum_{i=1}^{N} \left(i - \frac{N}{2}\right) \text{above}_\tau(x_i).$$

Each of the presented features covers specific characteristics of a time series. Naturally, depending on the application, the full power of the features can be achieved if we use combinations of them.

5 Evaluation

We used four datasets from the UCR Time Series Data Mining Archive [17]. Dataset "DS1" contains 600 time series of length 60 divided into 6 classes. This dataset is the *SynthCtrl* dataset and contains artificially created time series. The *GunX* dataset will be denoted as "DS2". It contains tracking data of the movement of persons while either drawing or pointing a gun. This dataset contains 200 time series of length 150 grouped into 2 classes. The *Trace* dataset is a synthetic dataset which describes instrumentation failures in a power plant. It will be referred to as "DS3" and contains 200 time series in 4 different classes. The length of each time series is 275. The last dataset ("DS4") is



(a) Average classification accuracy

(b) Average runtime

Fig. 5. Comparison of different approaches.

the *Leaf* dataset. Images were used to create this dataset. It consists of 442 time series of length 150 grouped into 6 classes.

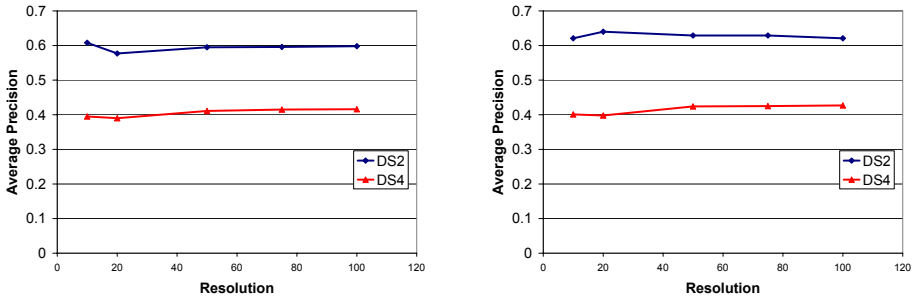
We compared our approach in terms of accuracy (average precision/recall of kNN classification experiments) with the following approaches: (1) Euclidean distance on the raw time series, (2) DTW on the raw time series, the global feature extraction approach from [1] with (3) unweighted and (4) weighted feature combination, and our approach using all proposed ALFs with (5) an unweighted feature combination as well as (6) the weighted feature combination. For both weighted competitors (approaches (4) and (6)), we applied the method proposed above using 20% of the particular datasets for learning the weights. Moreover, the k parameter of our weighting method was chosen to be 5 times the number of existing classes.

We ran a kNN classifier with several k values. The resulting average precisions for all competitors applied to our datasets are presented in Figure 5(a). As it can be observed, our ALF approaches outperform the competing techniques for all examined datasets (cf. Figure 5(a)). Let us note that we even achieved a higher classification accuracy than DTW which, in contrast to our approach, has a quadratic runtime w.r.t. the length of the time series. In addition, the experiments show that our novel weighted feature combination in average further boosts the classification quality. Even the global feature based classification can be improved with our proposed feature combination technique. Let us note, that the single precision results obtained for each tested k value look very similar. The average runtimes for one distance computation of the competitors are depicted in Figure 5(b). It can be observed that our approach clearly outperforms DTW by more than one order of magnitude. This is important because DTW is the only competitor that achieves roughly similar accuracy like our approach but for the cost of far higher runtimes.

We also evaluated the classification accuracy of each proposed ALF separately. Table 1 compares these accuracy values with the accuracy of the (weighted) combination of all ALFs. It can be observed, that on three out of four datasets choosing only one (the best) ALF is less accurate than combining all ALFs for similarity search.

ALF	ATQ	TIC	$\varnothing TIL$	$maxTIL$	$\varnothing TID$	$maxTID$	$absTXA$	$diffTXA$	TB	**combined**
DS1	77.2	**78.6**	64.8	73.5	56.4	64.0	78.5	44.9	62.8	**83.6**
DS2	59.2	54.8	56.2	59.9	52.4	52.5	**69.5**	55.9	53.8	**62.9**
DS3	63.8	43.8	65.4	74.5	74.8	**84.5**	65.3	81.3	54.7	**90.5**
DS4	**41.0**	37.6	32.4	30.9	30.7	33.3	26.0	27.1	18.8	**42.5**

Table 1. Classification accuracy of single ALFs (in percent).



(a) Unweighted combination

(b) Weighted combination

Fig. 6. Average classification accuracy of ALF for different resolutions.

In a further experiment, we evaluated the resolution r that specifies the number of amplitude-levels used for deriving the corresponding ALFs. Since r determines the length of the derived ALF sequences, this parameter seems to be important for the accuracy of the search. In fact, for a broad range of resolution values (r between 10 and 100), the average classification accuracy was very stable for all datasets. In Figure 6 we depicted the results for the two real-world datasets DS2 and DS4 for the unweighted and the weighted combination. The experiments showed that even for a very low value of r, i.e. rather short ALF sequences, we gain very accurate results.

6 Conclusions

In this paper we proposed a new general framework for generating high quality Amplitude-Level Features (ALF) from time series. An advantage when using ALFs for similarity search is that the runtime is independent of length of the time series. Thus, ALFs are adequate even for long sequences as occurring frequently in multimedia applications. We furthermore introduced several ALFs that are able to describe the characteristic properties of a time series. We also proposed a method to combine several feature representations. We showed in our experimental evaluation that our proposed technique outperforms traditional similarity search methods in terms of accuracy. In addition, our

approach significantly outperforms the only competitor that achieves roughly similar accuracy in terms of runtime.

References

1. Wang, X., Smith, K., Hyndman, R.: Characteristic-based clustering for time series data. Data Min. Knowl. Discov. **13**(3) (2006) 335–364
2. Naphade, M., Wang, R., Huang, T.: Multimodal pattern matching for audio-visual query and retrieval. In: Proc. SPIE. (2001)
3. Naphade, M., Yeung, M., Yeo, B.: A novel scheme for fast and efficient video sequence matching using compact signatures. In: Proc. SPIE. (2000)
4. Yeung, M.M., Liu, B.: Efficient matching and clustering of video shots. In: Proc. ICIP. (1995)
5. Berndt, D., Clifford, J.: Using dynamic time warping to find patterns in time series. In: Proc. AAAI-94 W. on Knowledge Discovery and Databases. (1994) 229–248
6. Nanopoulos, A., Alcock, R., Manolopoulos, Y.: Feature-based classification of time-series data. Information processing and technology, isbn 1-59033-116-8 (2001) 49–61
7. Deng, K., Moore, A., Nechyba, M.: Learning to recognize time series: Combining arma models with memory-based learning. In: IEEE CIRA. (1997) 246–250
8. Ge, X., Smyth, P.: Deformable markov model templates for time-series pattern matching. In: Proc. ACM SIGKDD. (2000) 81–90
9. Keogh, E., Lonardi, S., Ratanamahatana, C.A.: Towards parameter-free data mining. In: Proc. ACM SIGKDD. (2004) 206–215
10. Sun, X.: Pitch determination and voice quality analysis using subharmonic-to-harmonic ratio. In: Proc. ICASSP. (2002)
11. Liu, M., Wan, C.: Feature selection for automatic classification of musical instrument sounds. In: Proc. JCDL. (2001)
12. Tremain, T.: The government standard linear predictive coding algorithm: Lpc-10. (1982)
13. Pampalk, E.: A matlab toolbox to compute music similarity from audio. In: Proc. ISMIR. (2004)
14. Mitrovic, D., Zeppelzauer, M., Breiteneder, C.: Discrimination and retrieval of animal sounds. In: Proc. MMM. (2006)
15. Bustos, B., Keim, D., Saupe, D., Schreck, T., Vranic, D.: "Automatic selection and combination of descriptors for effective 3D similarity search". In: Proc. ICME. (2004)
16. Kriegel, H.P., Pryakhin, A., Schubert, M.: Multi-represented knn-classification for large class sets. In: Proc. DASFAA. (2005)
17. Keogh, E.: The ucr time series data mining archive (2006) Riverside CA. University of California, `http://www.cs.ucr.edu/~eamonn/TSDMA/index.html`.