In Proc. 18th Int. Conf. on Scientific and Statistical Database Management (SSDBM'06), Vienna, Austria, 2006.

Time Series Analysis Using the Concept of Adaptable Threshold Similarity

Johannes Assfalg, Hans-Peter Kriegel, Peer Kröger, Peter Kunath, Alexey Pryakhin, Matthias Renz
Institute for Informatics, University of Munich, Germany
{assfalg,kriegel,kroegerp,kunath,pryakhin,renz}@dbs.ifi.lmu.de

Abstract

The issue of data mining in time series databases is of utmost importance for many practical applications and has attracted a lot of research in the past years. In this paper, we focus on the recently proposed concept of threshold similarity which compares the time series based on the time frames within which they exceed a user-defined amplitude threshold τ. We propose a novel approach for cluster analysis of time series based on adaptable threshold similarity. The most important issue in threshold similarity is the choice of the threshold τ. Thus, the threshold τ is automatically adapted to the characteristics of a small training dataset using the concept of support vector machines. Thus, the optimal τ is learned from a small training set in order to yield an accurate clustering of the entire time series database. In our experimental evaluation we demonstrate that our cluster analysis using adaptable threshold similarity can be successfully applied to many scientific real-world data mining applications.

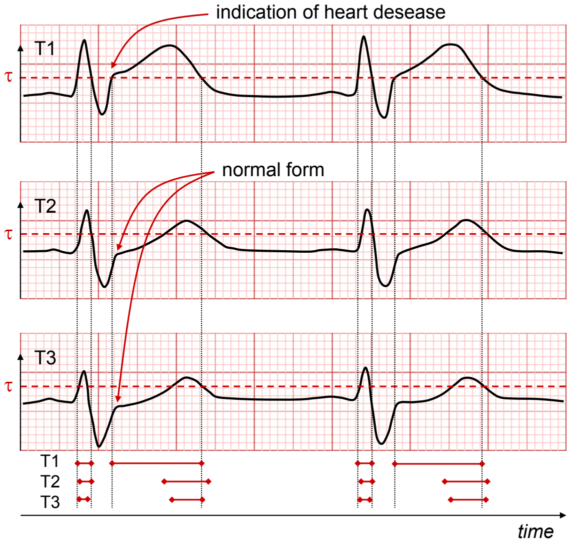


Figure 1. Sample application of threshold similarity.

1. Introduction

The analysis of time series data is of great practical importance in many application areas including stock marketing, astronomy, environmental analysis, molecular biology, and pharmacogenomics. As a consequence, a lot of research work has focused on similarity search in time series databases in the past years. Here, the similarity between time series, e.g. similar patterns of time series, plays a key role for the analysis. Recently, a novel but very important similarity measure called *threshold similarity* has been introduced [6, 5] which enables the analysis of time series tightly focused on a specific amplitude spectrum, in particular amplitudes that are important and significant for the analysis goal. Given two time series X and Y, and an amplitude threshold τ, X and Y are considered similar if their amplitudes exceed the threshold τ within similar time intervals. Using threshold similarity, the exact values of the time series are not considered. Rather it is only examined whether the time series objects have similar time intervals above or below the given threshold τ. Thus, time series can be considered as similar even if their absolute values are considerably different as long as they have similar time frames during which the time series exceeds the query threshold.

Threshold similarity has been shown to be useful in several applications [5]. A sample application from medical analysis is visualized in Figure 1 where three real electrocardiogram (ECG) plots $T1$, $T2$ and $T3$ are shown. Plot $T1$ indicates a high risk for cardiac infarct due to the abnormal deflection after the systole (ST-T-phase), whereas $T2$ and $T3$ both show a normal curve indicating a low risk. For the examination of time series w.r.t. this abnormal characteristic, there is no need to examine the entire curve. A better way to detect such kind of characteristics is to analyze only the relevant parts of the time series, for instance observing those parts of the time series which exceed a specified threshold as depicted in our example. Let us now consider the time interval sequences (below the ECG-curves) which correspond to the time frames within which the time series exceed the threshold τ. We can observe that the time interval sequences derived from $T2$ and $T3$ differs marginally. In contrast, the time series $T1$ shows a quite different characteristic, caused by the ECG-aberration which indicates the heart disease.

The most important issue of threshold similarity is obviously the choice of the threshold τ. In the medical analysis example (cf. Figure 1), the suitable threshold τ was selected by a domain expert (knowing about the characteristics of an abnormal time curve in case of a cardiac infarct patient), in order to discriminate between patients with a low/high risk for cardiac infarct. However, it can be easily seen that if the threshold would have been chosen lower than depicted in Figure 1, all three time series would have produced rather similar time intervals and, thus, the time series $T1$ could not have been discriminated from the other two time series $T2$ and $T3$. Since the optimal threshold for discriminating a predefined class system is not known in advance in many applications, a method for the automatical determination of the optimal threshold using a small number of labeled time series as training set is mandatory. Thus, for cluster analysis, a semi-supervised approach is envisioned where first, the best suitable threshold is determined automatically by means of a small training set and second, a cluster analysis using the concept of threshold similarity (based on the previously learned, i.e. adapted threshold) is performed in order to detect novel and important patterns. So far, there is only one approach to "optimally" adapt the threshold for a threshold similarity analysis of time series [4]. However, this approach uses a very simple and — in most cases — not very accurate method to judge the quality of a given threshold for the analysis task.

In this paper, we present a novel semi-supervised framework for the cluster analysis of time series using adaptable threshold similarity. This frame work consists of two phases, a training phase and a clustering phase (cf. Figure 2). In the first phase, the most suitable parameter setting, i.e. the choice of the threshold value, is determined by applying training datasets for the complete clustering process. Our proposed method uses the observation that the classification of some labeled objects leads to different results depending on the chosen threshold τ, i.e. τ influences the separability of the classes which we quantify by a so-called separation score. Therefore, we compute the separation score for each threshold of a given training set in a training step first. This results in a quality curve depending on τ. The optima of this curve can give useful hints on how to adapt the threshold for the second phase where the entire dataset is clustered. One might argue that performing a number of clusterings for different thresholds followed by a clustering quality computation can lead to the same results and even eliminates the training phase. However, this ignores the fact that many clustering algorithms have a runtime of $O(n^2)$ or require several iterations until they terminate. Besides, we use only a small training set for calculating the separability score whereas the clusterings would have to be computed on the entire dataset. Furthermore, additional time would be required to analyze whether the clustering results are meaningful or not.

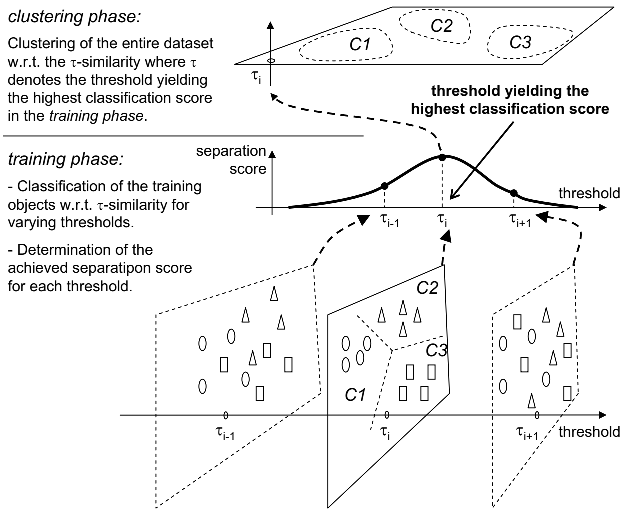

clustering phase:
Clustering of the entire dataset w.r.t. the τ-similarity where τ denotes the threshold yielding the highest classification score in the training phase.
C1
C2
C3
τ_i
threshold yielding the highest classification score
training phase:
- Classification of the training objects w.r.t. τ-similarity for varying thresholds.
- Determination of the achieved separatipon score for each threshold.
separation score
threshold
τ_{i-1}
τ_i
τ_{i+1}
C2
C3
C1
τ_{i-1}
τ_i
τ_{i+1}
threshold

Figure 2. General approach.

The remainder is organized as follows. Section 2 gives an overview of related work and points out our contributions. Section 3 presents our framework for semi-supervised cluster analysis using threshold similarity. Section 4 provides an experimental evaluation and Section 5 concludes the paper.

2. Related Work and Contributions

2.1. Time Series Analysis

In general, a time series of length d can be viewed as a feature vector in a d-dimensional space, where the similarity between two time series corresponds to their distance in the feature space, e.g. the Euclidean distance. Since d is usually large, the efficiency and the effectiveness of data analysis methods is rather limited due to the *curse of dimensionality*. Thus, several more suitable representations of time series data, e.g. by reducing the dimensionality, have been proposed. Most of them are based on the GEMINI indexing approach [15]: extract a few key *features* for each time series and map each time sequence X to a point $f(X)$ in a lower dimensional feature space, such that the distance between X and any other time series Y is always lower-bounded by the Euclidean distance between the two points $f(X)$ and $f(Y)$. For an efficient access any well known spatial access method can be used to index the feature space. The proposed methods mainly differ in the representation of the time series, including among others, DFT [1] and extensions [26], DWT [12], PAA [27], SVD [21, 2], APCA [19], Chebyshev Polynomials [11], and cubic splines [7]. However, all techniques which are based on dimensionality reduction cannot be directly applied to threshold similarity, because usually, in a reduced feature space, the original intervals indicating that a time series is above a given threshold is not available anymore. In addition, the approximation generated by dimensionality reduction techniques cannot be used for our purposes directly because they still represent the exact course of the time series rather than intervals of values above a threshold.

Beside the Euclidean distance, several other common distance functions have been successfully used for time series analysis including Dynamic Time Warping (DTW) which is conceptually similar to sequence alignment, Pearson's correlation coefficient which measures the global correlation between two time series, or angular separation, also known as cosine distance which defines the distance in terms of the angle between two feature vectors. In contrast to our approach, all these distance measures consider the absolute values of the time series rather than the intervals of values above a given threshold.

Having defined a distance measure on time series data, one can apply any analysis task. For clustering time series data, most of the various clustering methods proposed in the past decades have been successfully applied. A general overview over clustering methods is given in [18].

2.2. Semi-Supervised Cluster Analysis

In addition to the similarity information used by unsupervised clustering, in many cases a small amount of knowledge is available concerning either pairwise (must-link or cannot-link) constraints between data items or class labels for some items. In contrast to clustering which does not use any knowledge except for the similarity information of the data, semi-supervised analysis can profit from this knowledge to guide or adjust the clustering. Obviously, semi-supervised analysis methods achieve better results than their unsupervised counterparts. In recent years, several methods in the area of semi-supervised cluster analysis have been proposed. The main idea of semi-supervised clustering is to determine clusters that are 'immaculate' w.r.t. class labels of objects to be analyzed. It can also be considered as using labeled data as feedback in order to help to cluster unlabeled data. Most of the proposed methods for semi-supervised clustering assume that class labels for all objects to be processed are given.

[22] proposes a method based on a mixture of hidden Markov models that makes use of prior knowledge in order to improve the robustness and the quality of the local optima found. The author of [28] introduces a semi-supervised classification for time sequences based on hidden Markov models. Two different semi-supervised learning paradigms are discussed. The author observed that using unlabeled data can increase the classification accuracy.

Several extensions of existing standard clustering algorithms have been proposed in the literature. A brief survey is given in [14] describing SPAM a supervised variant of PAM, SRIDHCR, a greedy algorithm with random restart, SCEC, an evolutionary algorithm, TDS, a medoid-based top down partitioning algorithm. In [25], a variant of a k-means based clustering algorithm is proposed. The authors derive constraints from the labeled objects which are used during the clustering. They distinguish between explicit and cannot-link constraints. In [8], a k-means based method is introduced which is based on both types of constraints and which exploits the data distribution. The authors of [13] describe an evolutionary method for semi-supervised clustering. This approach has to be initialized with k arbitrary centroids and optimizes a quality measure considering cluster dispersion and impurity. In order to detect a cluster structure that reflects the class distribution of the labeled training data, further methods have been developed which use a standard clustering algorithm by applying an adaptive similarity measure. The authors of [20] propose to apply a complete-link clustering algorithm after replacing the Euclidean distance with the shortest path algorithm. The approach described in [10] weights the edit distance using an expectation maximization algorithm to detect approximately duplicate objects in a database. [9] describes a probabilistic framework for semi-supervised clustering to additionally support several non Euclidian distance measures, e.g. the cosine distance.

All mentioned methods for semi-supervised clustering do not take the threshold-based similarity of time series data into account. In our approach, we use the density-based hierarchical clustering algorithm OPTICS [3]. However, any clustering algorithm is applicable in our framework.

2.3. Adaptable Threshold Similarity

The concept of threshold similarity in time series databases has been proposed in [6, 5]. We will review and extend the basic definitions in Section 3. The only method to adopt an "optimal" threshold for threshold similarity from a small training set so far has been proposed in [4]. If the training set contains instances of $m \geq 2$ classes C_i, $1 \leq i \leq m$, the best threshold of a query for a specific class C_i is evaluated by computing the pairwise silhouette width of C_i to all other classes C_j, $i \neq j$. The minimal silhouette width of all these pairs is used as so-called separation score which estimates the separability of the class C_i in the training set for a given threshold τ. The threshold with the best separation score is chosen as the optimal threshold for the given query. However, the quality of a given threshold for a query is only evaluated using the silhouette width in the training set. This is a rather simple heuristics and usually far from the optimal solution.

2.4. Our Contribution

In this paper, we propose a novel method for time series cluster analysis using adaptable threshold similarity. Our approach is different to that proposed in [4] in the following aspect. We propose a novel approach to compute the separation score for a given threshold τ using the concept of support vector machines that provides an optimal solution for the task of learning the best suitable threshold for the succeeding clustering step rather than using simple heuristics. For that purpose, we further extend the basic definition of threshold similarity in [6, 5] by defining a kernel function for threshold similarity. In addition, we show how our new concepts can be integrated into a complete framework for semi-supervised cluster analysis of time series databases using the concept of threshold similarity. Last but not least, we present an experimental evaluation where we show the superiority of our approach in comparison to [4] by evaluating the quality of the learned threshold using the entire database objects rather than some sample queries (as in [4]). We further demonstrate in our experiments that the concept of threshold similarity is better suitable in some applications than using the Euclidean distance measure and that our method is able to gain new insights from a database even in the case that the training set is incomplete, i.e. not all classes that should be discovered, are present in the training set.

3. Semi-supervised Threshold Similarity Analysis

3.1. Preliminaries

We define a time series X as a sequence of pairs $(x_i, t_i) \in \mathbb{R} \times T : (i = 1..N)$, where T denotes the domain of time and x_i denotes the measurement at time t_i. Furthermore, we assume that the time series entities are given in such a way that $\forall i \in 1, .., N-1 : t_i < t_{i+1}$. Let us note, that in most applications the time series are derived from discrete measurements of continuously varying attributes. However, time series are commonly depicted as continuous curves, where the missing curve values (i.e. values between two measurements) are estimated by means of interpolation. From the large range of appropriate solutions for time series interpolation, in this paper we assume that the time series curves are supplemented by linear interpolation which is the most prevalent interpolation method.

In the following, $\tau \in \mathbb{R}$ denotes a given threshold of interest. Intuitively, using threshold similarity (w.r.t. τ), two time series are considered similar, if they have similar time intervals during which they exceed the threshold τ. The set of intervals during which a given time series $X = \langle (x_i, t_i) \in \mathbb{R} \times T : i = 1..N \rangle$ is above τ is called *threshold-crossing time interval sequence* of X with respect to τ, denoted by $TC_\tau(X)$ [6, 5]. Formally,
$TC_\tau(X) = \langle (l_j, u_j) \in T \times T : j \in \{1, .., M\}, M \leq N \rangle$
is a sequence of time intervals, such that

$$\forall t \in T : (\exists j \in \{1, .., M\} : l_j < t < u_j) \Leftrightarrow x(t) > \tau.$$

An interval $t^X_{\tau,j} = (l^X_j, u^X_j) \in TC_\tau(X)$ is called *threshold-crossing time interval*. We omit the index τ if it is clear from context.

Given two arbitrary time intervals $t = (l_t, u_t) \in T \times T$ and $s = (l_s, u_s) \in T \times T$, the distance function $d_{int} : (T \times T) \times (T \times T) \rightarrow \mathbb{R}$ between two time intervals is defined as [6, 5] $d_{int}(t, t) = \sqrt{(l_t - l_s)^2 + (u_t - u_s)^2}$.

3.2. General Idea

As stated above, our main goal is to yield an accurate clustering of the database $\mathcal{D}$ of time series using threshold similarity. In order to choose the optimal threshold value τ, we want to apply a semi-supervised clustering procedure, where we learn the optimal threshold from a small training set $\mathcal{T}$ of already labeled time series before clustering (cf. Figure 2). This learning phase preceding the clustering phase is the key step in our framework.

Let $\mathcal{T}$ be the training set containing time series objects that are labeled according to a predefined class system $\mathcal{C} = \{C_1, \ldots C_k\}$ of $k \geq 2$ classes. We need to learn the threshold values from the objects in $\mathcal{T}$ that are able to separate time series data of one training class C_i from the other training classes C_j $(i \neq j)$, i.e. threshold values that yield low similarity values for time series belonging to different classes and high similarity values for time series belonging to the same class.

The class system $\mathcal{C}$ defined for the training data $\mathcal{T}$ need not be complete. There may be some further classes $\hat{C} \notin \mathcal{C}$ for which no training data is at hand, i.e. none of the objects in $\mathcal{T}$ is labeled with one of these classes $\hat{C}$. Furthermore, it is also possible that the user is not aware of the existence of all classes. The dataset may contain unknown classes which could also be interesting for the user.

Obviously, these classes are excluded from the learning phase, i.e. the learned threshold need not be optimal for these classes. However, as we show in the experimental section (cf. Section 4), threshold values that exhibit a high separability for only a few classes in the training data are quite often also a good choice for the detection of unknown or missing classes during the clustering phase. Thus, by providing only partial information during the training phase, our approach is able to retrieve novel information in the clustering phase. This is contrary to a fully supervised approach, where novel classes cannot be detected.

As mentioned above, the optimal threshold τ_{opt} separates the classes $C_i \in \mathcal{C}$, $1 \leq i \leq k$, in our training set $\mathcal{T}$ in a best possible way. We formalize the separability of a threshold τ by means of a *separation score*. In fact, we measure the separation score of a broad range of possible thresholds. One key difference of our approach to [4] is the computation of this separation score. In [4], the separability is measured using the minimum of all pairwise silhouette widths of each pair of classes C_i and C_j in $\mathcal{T}$. Obviously, this is a quite simple heuristics which leaves much space for improvement. In this paper, we compute an optimal score by applying the concept of support vector machines (SVMs). In general, SVMs provide an optimal separation of two classes [24] and can easily be extended to multi-class problems. However, in order to apply SVMs to threshold similarity of time series we have to extend the basic concepts of threshold similarity. In the following, we first explain these extensions of threshold similarity (cf. Section 3.3). We then introduce a new separation score in order to measure how good a given threshold separates the class system $\mathcal{C}$ in $\mathcal{T}$ (cf. Section 3.4). Last, we explain how the optimal threshold is determined (cf. Section 3.5) in order to yield a good clustering of the entire database $\mathcal{D}$.

3.3. Similarity Model

As discussed above, we use SVMs to separate the classes in $\mathcal{T}$ since they provide an optimal separation. However, basic SVMs can only be applied to feature vectors rather than interval sequences. Thus, in order to apply SVMs to time series using threshold similarity, we need the concept of kernel methods which have been successfully applied to learning from objects having a complex structure. Since, we represent the time series by sets of intervals we need a kernel method that can cope with set-based instances. Let χ denote the complete set of intervals of all objects in $\mathcal{T}$ generated by a given threshold τ. For our approach, in order to compare two time series $X, Y \in \mathcal{T}$, we use the set kernel $k(TC_\tau(X), TC_\tau(Y))$ which has been introduced in [16] and is defined by

$$k(TC_\tau(X), TC_\tau(Y)) := \sum_{t^X \in TC_\tau(X), t^Y \in TC_\tau(Y)} \kappa_\chi(t^X, t^Y),$$

where κ_χ denotes a kernel on χ, i.e. on single intervals. In order to keep the similarity function invariant to the size of the sets $TC_\tau(X)$ and $TC_\tau(Y)$, the kernel function has to be normalized by the cardinality of these sets:

$$\overline{k}(TC_\tau(X), TC_\tau(Y)) := \frac{k(TC_\tau(X), TC_\tau(Y))}{N(TC_\tau(X)) \cdot N(TC_\tau(Y))}.$$

Here, the normalization function

$$N(S) := \sum_{s \in S} S(s)$$

is used to compute the cardinality of the set S, i.e. $S(s)$ returns 1, if s is in the set S and 0 otherwise. Note, that this kind of normalization preserves the kernel property, i.e. the resulting function $\overline{k}_{set}$ is also a kernel [16].

The kernel function κ_χ is applied to a pair of threshold crossing time intervals $t^X \in TC_\tau(X)$ and corresponds to the similarity of two intervals. Since the distance function d_{int} proposed in [6, 5] is not a kernel, we cannot apply it directly to κ_χ. Rather, we need a similarity function κ_χ which fulfills the kernel property. In addition, this similarity function should fulfill the following condition: the smaller the distance between two intervals, the higher the similarity between them. With this condition, the similarity between two time intervals depends on their temporal offsets of their starting and ending times. Intuitively, the closer two time intervals start and the closer their interval length, the more similar they are. In our approach we apply the Gaussian kernel, which is defined as follows:

$$\kappa_\chi(t_i^X, t_j^Y) := e^{\frac{-d_{int}(t_i^X, t_j^Y)^2}{\sigma}},$$

where σ is a parameter which can be used to adjust the sensitivity of the similarity to the distance between t_i^X and t_i^Y. For low σ values, large interval distances have only little influence to the similarity.

Thus, the resulting kernel function to compare time series X and Y using the concept of threshold similarity w.r.t. threshold τ is defined as:

$$\mathcal{K}_\tau(X, Y) = \frac{\sum_{t^X \in TC_\tau(X), t^Y \in TC_\tau(Y)} e^{\frac{-d_{int}(t^X, t^Y)^2}{\sigma}}}{N(TC_\tau(X)) \cdot N(TC_\tau(Y))}.$$

Let us note, that the similarity between two time series according to threshold τ expressed by the kernel function $\mathcal{K}_\tau$ is also called *τ-similarity*.

3.4. Computing the Separation Score

As we have defined the similarity function $\mathcal{K}_\tau$ as a kernel function, we can now apply the concept of SVMs in order to measure the separability of given training data w.r.t. a specified threshold τ. The use of SVMs provides an optimal solution for the separation of our classes $\mathcal{C}$ in the training set $\mathcal{T}$. In addition, SVMs already contain information

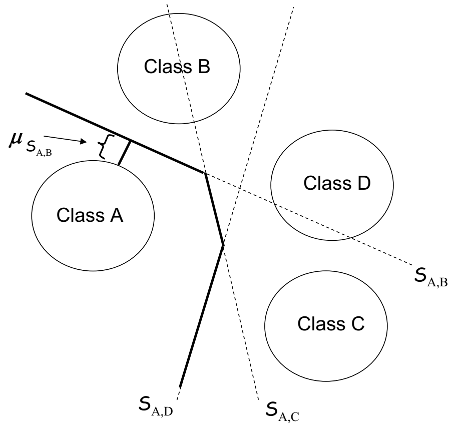


Figure 3. Computing the separation score.

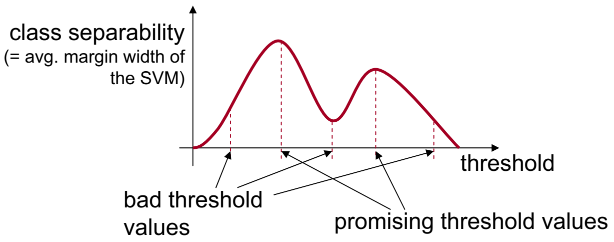


Figure 4. Determination of the Most Promising Threshold Values.

about the separability of the training data w.r.t. the desired classes and, thus, provide an elegant method to measure the separability of a given class system.

At first, we have to determine those threshold values which could be of interest and which we want to examine. Therefore, we select a range of amplitudes which could be meaningful for our analysis. In our experiments, we have chosen the complete amplitude spectrum covered by the time series objects contained in the training dataset. However, if any domain knowledge can be incorporated, this range of meaningful thresholds can be narrowed for performance reasons. In addition, we can apply the data structures proposed in [5] to access the threshold-crossing time intervals of a given time series X for any threshold τ very efficiently. Afterwards, we have to choose the resolution of our examination, i.e. how many thresholds we want to examine within the selected range.

After we have selected the increments of the threshold values, we evaluate each threshold value τ as follows: We determine the threshold-crossing time interval sequences of all training objects w.r.t. τ and train an SVM on this data. Standard SVMs are able to make only binary decisions. An SVM $\mathcal{S}^{\tau}_{A,B}$ computes a maximum-margin hyperplane which separates instances of two classes A and B using the kernel function $\mathcal{K}_{\tau}$. The width of the margin $\mu^{\tau}_{A,B}$ of this separating hyperplane is a valid indication on the separability of the two classes. Obviously, the larger this width is, the more confident is the SVM to separate objects from A and B correctly. Since usually, we may have more than two classes in $\mathcal{C}$, and standard SVMs can only handle the binary case, we apply the so-called *one-versus-one approach*, i.e. we for each pair of classes $C_i, C_j \in \mathcal{C}$ we train an SVM $\mathcal{S}^{\tau}_{i,j}$. Thus, we obtain for each pair of classes $C_i, C_j \in \mathcal{C}$ the margin-width $\mu^{\tau}_{i,j} = \mu^{\tau}_{j,i}$ which is a measurement for the separability of C_i and C_j. An example is depicted in Figure 3. Four classes A, B, C, and D are separated using an SVM for each pair. Only the SVMs for which A participates are shown. The width of the margin of $\mathcal{S}_{A,B}$, $\mu_{A,B}$ is visualized (parameter τ is omitted).

Given the margin-width of all SVMs trained on each pair $C_i, C_j \in \mathcal{C}$, all these values have to be combined suitably in order to get one value which represents the global class separability. It is possible to represent the global separability by the smallest of all margin-widths. This approach guarantees that all desired classes are well separated. However, this solution seems to be a too optimistic approach, since the training dataset may contain classes which cannot be separated at all, regardless of the selected threshold value. Another approach is to pick the largest margin-width. However, this pessimistic solution is also not suitable, since two classes which are separable well for each threshold value do not reflect the global separability. For our approach we argue to take the average margin-width, because the separabilities of all observed classes are considered. This decision is confirmed by the results in our experiments (cf. Section 4). The resulting *separation score* is thus defined as

$$\text{score}(\tau) = \frac{1}{2 \cdot N(\mathcal{C})} \sum_{C_i, C_j \in \mathcal{C}, i \neq j} \mu^{\tau}_{j,i}.$$

Obviously, the higher this value is, the better the classes in $\mathcal{C}$ can be separated w.r.t. threshold τ.

3.5. Determining the Optimal Threshold for Clustering

Having determined the separation score for a range of interesting values, we can consider a quality curve over all these threshold values (cf. Figure 4). In such a separability diagram, we plot the examined threshold values along the x-axis and the the corresponding separation scores along the y-axis. From this diagram we can easily determine those threshold values which yield high curve values as most promising for the clustering step.

4. Evaluation

In this section, we present the results of a large number of experiments performed on a selection of time series datasets. In particular, we will present the results of our experiments with respect to efficiency and effectiveness.

4.1. Datasets and System Environment

For our evaluation, we used several real-world and synthetic datasets which are described in the following. A summarized description is depicted in Table 1.

Audio Dataset. The audio dataset (cf. DS3 in Table 1) contains time sequences expressing the temporal behavior of the energy, dynamics and peaks in music sequences. It consists of 36 classes and contains 756 time series objects with a length of up to 30000 values per sequence.

Scientific Datasets. The scientific datasets are derived from two different applications: the analysis of environmental air pollution and gene expression data analysis. The data on environmental air pollution is derived from the Bavarian State Office for Environmental Protection, Augsburg, Germany [1] and contains the daily measurements of 8 sensor stations distributed in and around the city of Munich from the year 2000 to 2004. One time series represents the measurements of one station at a given day containing 48 values for one of 10 different parameters such as temperature, ozone concentration, etc. The gene expression data from [23] contains the expression level of approximately 6,000 genes measured at only 24 different time slots. The expression level of a gene indicates how active it is. This dataset was derived from the Gene Expression Omnibus (GO)[2].

Other Datasets. The other datasets are derived from diverse fields and cover the complete spectrum of stationary/ non-stationary, noisy/ smooth, cyclical/ non-cyclical, symmetric/ asymmetric etc. data characteristics.

As mentioned above, we applied the density-based clustering method OPTICS [3] for the cluster analysis step. We used OPTICS due to its robustness w.r.t. data distribution and parameter setting. Again, let us note that any other clustering method is also applicable.

4.2. Effectiveness

In this section, we show that our proposed semi-supervised analysis yields promising results and can be successfully applied to several datasets with different characteristics.

Label	Dataset	Description
DS1	GDS 38	Gene expression dataset with 2562 instances, number of classes depends on GO-level
DS2	GDS 30	Gene expression dataset with 2628 instances, number of classes depends on GO-level
DS3	Audio	Audio dataset with 36 classes, 21 instances in each class
DS4	Trace	synthetic dataset with 16 classes, 50 instances in each class
DS5	Gunx	real world dataset with 2 classes, 100 instances in each class
DS6	CBF 100	synthetic dataset with 3 classes, 100 instances in each class

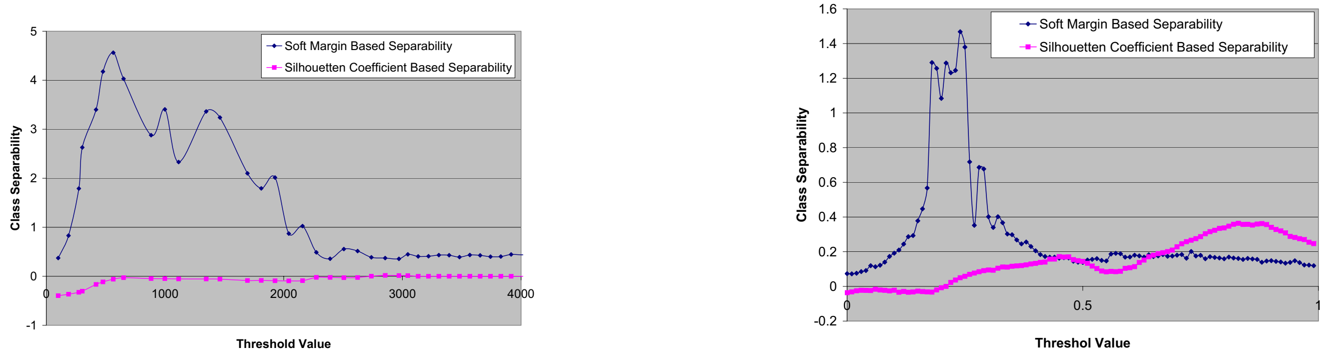
Table 1. Used datasets.

Validation of the Separation Score. We investigate the effectiveness of semi-supervised threshold queries which are used to find the optimal threshold value by means of a training dataset. We will show that choosing a supervised threshold leads to better clustering results than choosing an arbitrary threshold where no information about the dataset characteristics was used to select the threshold value. The curves depicted in Figure 5 show the separability (cf. Section 3) of the classes for varying threshold values. Figure 5(a) shows the results of the $DS3$ dataset and Figure 5(b) shows the results of the $DS5$ dataset. Obviously, different threshold values lead to different values for the separability of the classes. As mentioned in Section 3 high curve values should indicate threshold values which are promising for the cluster analysis.

The shown curves give raise to the question whether the threshold values with high separation score do indeed yield good results on the whole data set. To evaluate this, we clustered the time series for two different threshold values τ_+ and τ_- and determined the rand index and the average entropy [17]. For example, the threshold value $\tau_+ = 710$ which corresponds to a high separation score on the $DS3$ dataset resulted in a rand index equal to 0.97. Contrary, when using a threshold value of $\tau_- = 3064$ the rand index decreased to 0.61. Similar results were observed for other levels, for other threshold values, and on other datasets. Figure 6 shows the results for several datasets where τ_+ corresponds to a high separation value and τ_- corresponds to a low separation value. Figure 6(a) depicts the rand index and Figure 6(b) depicts the average entropy. The higher the rand index, the higher the clustering quality, whereas high aver-

[1] www.bayern.de/lfu

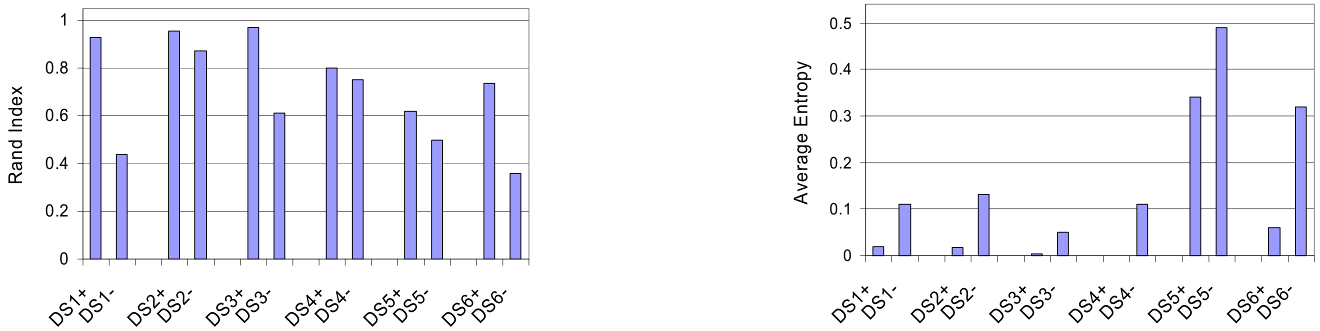
[2] http://www.ncbi.nlm.nih.gov/geo/



(a) Separability Plot on DS3.

(b) Separability Plot on DS6.

Figure 5. Separability Plots for Different Datasets.



(a) Rand Index.

(b) Average Entropy.

Figure 6. Analysis Quality for Suitable and Non-Suitable Threshold Values.

age entropy values indicate low clustering qualities. The results show that the clustering analysis based on threshold τ_+ always outperforms the analysis based on τ_-. This underlines the validity of our semi-supervised analysis approach.

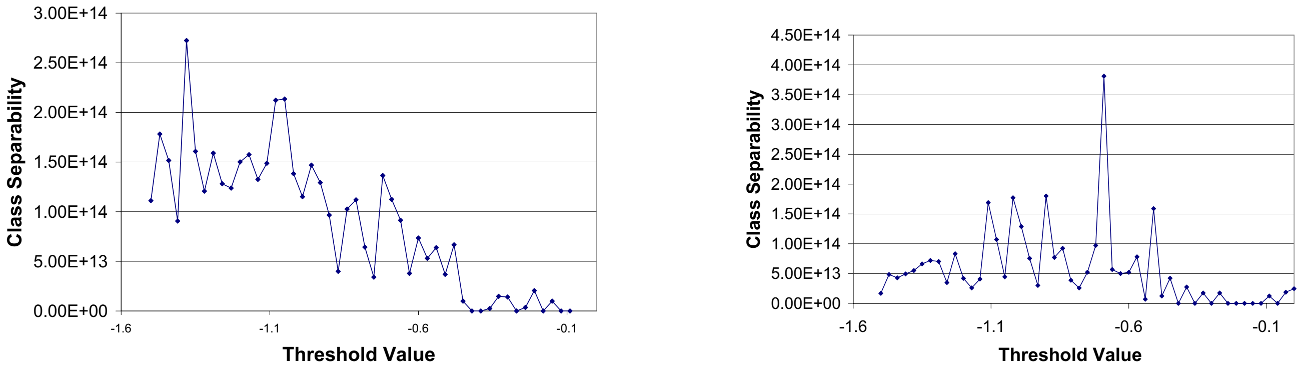
Adjustability to Different Training Classes. In this experiment, we are interested in how the optimal threshold values change when the expected results change, i.e. when the focus of the query changes. The following experiments were performed on the dataset $DS1$. For the first experiment, depicted in Figure 7(a) we used the GO functional classes on level 3. Afterwards we changed the focus of our analysis to the GO level 6. The results are depicted in Figure 7(b). As expected, we obtained different optimal threshold values for different purposes of the analysis.

Sensitivity to Incomplete Training Data. In our next experiment, we examined the sensitivity of our approach to missing classes in the training data. A low sensitivity corresponds to the ability to detect unknown knowledge. It is interesting, whether our analysis finds those classes which are existent in the training dataset, but which are not used in the training phase. The results for different datasets are depicted in Table 2.

Quality Measure	Dataset	classes present in $\mathcal{T}$ 25%	50%	75%	100%
Rand Index	DS1	0.929	0.93	0.93	0.93
	DS2	0.947	0.956	0.956	0.957
	DS3	0.8	0.969	0.969	0.97
Entropy	DS1	0.027	0.026	0.02	0.02
	DS2	0.021	0.017	0.017	0.017
	DS3	0.042	0.042	0.05	0.042

Table 2. Sensitivity to incomplete training data.

We performed several semi-supervised analysis tasks using different portions of the complete class information for the training phase. For each analysis we measured the rand index and the average entropy w.r.t. the complete class information. Obviously, the completeness of the detected classes increases with an increasing number of classes used for the training phase. However, a small amount of training classes suffices to find nearly the complete set of classes within our dataset.



(a) Separability Curve for GO level 3.

(b) Separability Curve for GO level 6.

Figure 7. Separability Curves for Different Training Classes.

Analysis Results on Scientific Datasets. The results on the air pollution dataset were very useful. We examined time sequences representing particulate matter parameters (M_{10}) derived from rural and urban sensor stations. The threshold-based analysis shows that the pollution by particle components in the city differs considerably from the pollution in rural regions.

The results on the gene expression dataset were also very interesting. Indeed, we retrieved functionally related genes in most of the reported clusters. For example, gene CDC25 and gene CIK3 were located in the same cluster. Both genes play an important role during the mitotic cell cycle. Furthermore, genes DOM34 and MRPL17 were in the same cluster as two genes that are not yet labeled (ORF-names: YOR182C and YGR220C, respectively). However all four genes are participating in the protein biosynthesis. In particular, our proposed analysis tool can be used to predict the function of genes whose biological role is not resolved yet.

Comparison with Related Work. Last but not least, we compared our method to existing approaches. Here, we compared our approach denoted by "Kernel" to the work in [4], denoted by "SilCoef", and to another similarity measure, in particular the Euclidean distance, denoted as "Euclid". The results of the comparison are depicted in Table 3. As it can be seen our method "Kernel" achieves the better Rand Index and entropy values on all data sets and, thus, outperforms both competitive methods, "SilCoef" and "Euclid" in terms of effectiveness.

Quality Measure	Dataset	Euclid	SilCoef	Kernel
Rand Index	DS1	0.46	0.927	0.9367
	DS2	0.252	0.94	0.957
	DS3	0.27	0.95	0.97
	DS4	0.5	0.75	0.8
	DS5	0.33	0.5228	0.619
	DS6	0.183	0.67	0.737
Entropy	DS1	0.0067	0.047	0.026
	DS2	0.018	0.0277	0.017
	DS3	0.9	0.01	0.0042
	DS4	0.89	0.018	0.001
	DS5	0.8	0.02	0.3
	DS6	0.96	0.025	0.06

Table 3. Comparison of clustering results w.r.t. different similarity measures.

5. Conclusions

In this paper, we proposed a framework for semi-supervised cluster analysis using adaptable threshold similarity. In particular, we proposed a method to adapt the threshold by learning the optimal threshold from a small training set in order to yield an accurate clustering of the entire time series. In our experimental evaluation, we showed that our proposed approach yields valuable clustering results, even if only partial information is available for adapting the threshold to an optimal value. Beside the analysis of a dataset according to specific class labels, our approach can help to find unknown but potentially useful knowledge. For future work, we plan to investigate threshold similarity analysis with multiple thresholds at the same time.

6. Acknowledgments

We would like to thank U. Böllmann and M. Meindl providing us with the real-world datasets from the Bavarian State Office for Environmental Protection, Augsburg, Germany.

Part of this work is supported by the German Ministry for Education, Science, Research and Technology (BMBF) under grant no. 031U112F within the BFAM (Bioinformatics for the Functional Analysis of Mammalian Genomes) project which is part of the German Genome Analysis Network (NGFN).

References

[1] R. Agrawal, C. Faloutsos, and A. Swami. "Efficient Similarity Search in Sequence Databases". In *Proc. 4th Conf. on Foundations of Data Organization and Algorithms*, 1993.

[2] O. Alter, P. Brown, and D. Botstein. "Generalized Singular Value Decomposition for Comparative Analysis of Genome-Scale Expression Data Sets of two Different Organisms". *Proc. Natl. Aca. Sci. USA*, 100:3351–3356, 2003.

[3] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. "OPTICS: Ordering Points to Identify the Clustering Structure". In *Proc. ACM SIGMOD*, 1999.

[4] J. Aßfalg, H.-P. Kriegel, P. Kröger, P. Kunath, A. Pryakhin, and M. Renz. "Semi-supervised Threshold Queries on Pharmacogenomics Time Sequences". In *to appear in Proc. 4th Asia Pacific Bioinformatics Conf. (APBC'06), Taipei, Taiwan*, 2006.

[5] J. Aßfalg, H.-P. Kriegel, P. Kröger, P. Kunath, A. Pryakhin, and M. Renz. "Similarity Search on Time Series Based on Threshold Queries". In *to appear in Proc. (EDBT'06), Munich, Germany*, 2006.

[6] J. Aßfalg, H.-P. Kriegel, P. Kröger, P. Kunath, A. Pryakhin, and M. Renz. "Threshold Similarity Queries in Large Time Series Databases". In *to appear in Proc. Int. Conf. on Data Engineering (ICDE'06), Atlanta, GA*, 2006.

[7] Z. Bar-Joseph, G. Gerber, T. Jaakkola, D. Gifford, and I. Simon. "Continuous Representations of Time Series Gene Expression Data". *J. Comput. Biol.*, 3-4:341–356, 2003.

[8] S. Basu, M. Bilenko, and R. J. Mooney. A probabilistic framework for semi-supervised clustering. In *KDD*, pages 59–68, 2004.

[9] M. Bilenko, S. Basu, and R. J. Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *ICML*, 2004.

[10] M. Bilenko and R. J. Mooney. Adaptive duplicate detection using learnable string similarity measures. In *KDD*, pages 39–48, 2003.

[11] Y. Cai and R. Ng. "Index Spatio-Temporal Trajectories with Chebyshev Polynomials". In *Proc. ACM SIGMOD*, 2004.

[12] K. Chan and W. Fu. "Efficient Time Series Matching by Wavelets". In *Proc. IEEE ICDE*, 1999.

[13] A. Demiriz, K. P. Bennett, and M. J. Embrechts. Semi-supervised clustering using genetic algorithms. In *Intelligent Engineering Systems Through Artificial Neural Networks 9*, pages 809–814, 1999.

[14] C. F. Eick, N. M. Zeidat, and Z. Zhao. Supervised clustering - algorithms and benefits. In *16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2004), 15-17 November 2004, Boca Raton, FL, USA*, pages 774–776, 2004.

[15] C. Faloutsos, M. Ranganathan, and Y. Maolopoulos. "Fast Subsequence Matching in Time-series Databases". In *Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'94), Minneapolis, MN*, 1994.

[16] T. Gärtner, P. A. Flach, A. Kowalczyk, and A. J. Smola. "Multi-Instance Kernels". In *ICML*, pages 179–186, 2002.

[17] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. "On Clustering Validation Techniques". In *Intelligent Information Systems Journal*, 2001.

[18] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Academic Press, 2001.

[19] E. Keogh, K. Chakrabati, S. Mehrotra, and M. Pazzani. "Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases". In *Proc. ACM SIGMOD*, 2001.

[20] D. Klein, S. D. Kamvar, and C. D. Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *ICML*, pages 307–314, 2002.

[21] F. Korn, H. Jagadish, and C. Faloutsos. "Efficiently Supporting Ad Hoc Queries in Large Datasets of Time Sequences". In *Proc. ACM SIGMOD*, 1997.

[22] A. Schliep, I. G. Costa, C. Steinhoff, and A. Schonhuth. Analyzing gene expression time-courses. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 2(3):179–193, 2005.

[23] P. Spellman, G. Sherlock, M. Zhang, V. Iyer, K. Anders, M. Eisen, P. Brown, D. Botstein, and B. Futcher. "Comprehensive Identification of Cell Cycle-Regulated Genes of the Yeast Saccharomyces Cerevisiae by Microarray Hybridization.". *Molecular Biolology of the Cell*, 9:3273–3297, 1998.

[24] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

[25] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl. Constrained k-means clustering with background knowledge. In *ICML*, pages 577–584, 2001.

[26] S. Wichert, K. Fokianos, and K. Strimmer. "Identifying Periodically Expressed Transcripts in Microarray Time Series Data". *Bioinformatics*, 20(1):5–20, 2004.

[27] B. K. Yi and C. Faloutsos. "Fast Time Sequence Indexing for Arbitrary Lp Norms". In *Proc. VLDB*, 2000.

[28] S. Zhong. Semi-supervised sequence classification with hmms. *IJPRAI*, 19(2):165–182, 2005.