Pattern Recognition Letters 45 (2014) 129–135

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Support vector clustering of time series data with alignment kernels ☆

Benedikt Boecking [b,*], Stephan K. Chalup [a], Detlef Seese [b], Aaron S.W. Wong [a]

[a] School of Electrical Engineering & Computer Science, The University of Newcastle, Callaghan, NSW 2308, Australia
[b] Institute of Applied Informatics and Formal Description Methods (AIFB), Karlsruhe Institute of Technology, 76128 Karlsruhe, Germany

ARTICLE INFO

Article history:
Received 13 June 2013
Available online 4 April 2014

Keywords:
Support vector clustering
Time series
Alignment kernel
Clustering

ABSTRACT

Time series clustering is an important data mining topic and a challenging task due to the sequences' potentially very complex structures. In the present study we experimentally investigate the combination of support vector clustering with a triangular alignment kernel by evaluating it on an artificial time series benchmark dataset. The experiments lead to meaningful segmentations of the data, thereby providing an example that clustering time series with specific kernels is possible without pre-processing of the data. We compare our approach and the results and learn that the clustering quality is competitive when compared to other approaches.

1. Introduction

Support vector clustering (SVC), a kernel-based learning algorithm inspired by Support Vector Machines (SVMs) [27], was introduced by [2]. SVC offers some unique characteristics compared to other clustering algorithms such as its capabilities in dealing with outliers and arbitrarily-shaped cluster contours. Since its introduction, several researchers have contributed variations and improvements of the original SVC approach [30,13,15,11,18,19]. An important aspect of SVC is that it does not rely on knowledge of the data patterns' underlying probability distribution. Thus clustering can be approached without making assumptions on the number of clusters or their shape. SVC has the ability to work with structured data and has been shown to work well with high dimensional data [14]. SVC is a kernel method and can be used with any appropriate similarity measure that yields tight cluster contours and fulfils the requirements for being a Mercer kernel [2]. The Gaussian kernel is the most commonly used kernel in research concerning SVC and it has been shown to be successful on static data. Studies on the use of any other kernel function with SVC are very rare.

When attempting to cluster time series, one of the main challenges that presents itself is that clustering approaches used for static data cannot be used on time series without extensive pre-processing of the data. Standard similarity measures such as the Laplace kernel or the Gaussian kernel cannot compare sequences of different lengths. Furthermore, they are influenced by scale so that similarity in lower range can be weighed down by mild dissimilarity in higher range as [17] note. This can be mitigated by the normalisation of individual time series. But other problems such as the sensitivity of the standard similarity measures to offsets and phase shifts still remain. Additionally, simple similarity measures often lack the possibility to deal with noise within a sequence, which, for example, is a common feature of financial time series [25].

There are two ways through which SVC can be employed for time series clustering. One involves thorough pre-processing of the data before employing a standard kernel function with SVC. The second approach is using more advanced similarity measures such as dynamic time warping (DTW) [21] which address distinctive traits of time series. The experimental focus of our research lies in clustering time series with SVC and a time series kernel.

2. Support vector clustering

Ben-Hur et al. [2] introduced SVC, a non-parametric clustering method. It is closely related to one-class classification and density estimation using SVMs as proposed in [22–24] where a set of contours enclose data points with similar underlying distributions. Ben-Hur et al. [2] interpret these contours as cluster boundaries and points that lie within the same contour as belonging to the same cluster.

SVC is carried out in two stages. During the first stage, the *Cluster Description*, the minimal enclosing hypersphere of the data in feature space is computed by finding the support vectors which define the cluster boundaries. The approach is described in detail in [2] and uses a similar constraint optimisation approach as that

☆ This paper has been recommended for acceptance by G. Borgefors.

* Corresponding author. Tel.: +49 176 842 732 81.
E-mail address: c3141537@uon.edu.au (B. Boecking).

http://dx.doi.org/10.1016/j.patrec.2014.03.015
0167-8655/

of SVMs. The primal optimisation objective of finding the minimal enclosing sphere can be expressed as follows:

$$min_R R^2 + C\sum_j \xi_j \tag{1}$$

$$\text{s.t.} \quad \left\|\Phi(x_j) - a\right\|^2 \leqslant R^2 + \xi_j, \quad \xi_j \geqslant 0, \quad \forall j \in \{1,\ldots,N\} \tag{2}$$

where Φ is a nonlinear transformation from input space into some high dimensional feature space, a is the centre of the enclosing sphere of radius R, and $C > 0$ being a penalty term for introducing slack variables ξ_j that deal with outliers in the dataset.

During a second stage, the *Cluster Labelling*, the patterns are assigned to clusters. These two stages are generally carried out iteratively with the purpose of finding the best tuning for the hyper-parameters of SVC.

Ben-Hur et al. [2] use a geometric approach called Complete Graph (CG) labelling to assign the data points to clusters based on the idea that given two points that belong to different clusters, any path that connects them must exit from the minimal enclosing sphere in feature space. Such a path must therefore contain a point y for which $R(y) \geqslant R$. The cluster labelling is hence determined by the connected components of the graph induced by:

$$A_{ij} = \begin{cases} 1, & \text{if, for all } y \text{ on the line segment connecting } x_i \\ & \text{and} \quad x_j, \quad R(y) \leqslant R \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

In CG labelling the adjacency matrix A is computed first. In a second step the connected components of this undirected graph are calculated by sampling points on the paths to determine whether they exit the sphere in feature space.

Most studies on SVC have focused on improving the labelling stage performance (see e.g. [30,15,18,12,20]). Most recently [19] proposed to use convex decomposition and cone cluster labelling during the labelling stage in an approach which also made adaptions to the cluster description stage.

The SVC algorithm with standard Gaussian kernel has two adjustable parameters which influence the number of outliers and SV. The first parameter q is the width parameter of the Gaussian Kernel. Increasing q leads to a greater number of SVs which leads to tighter cluster contours and contour splitting.

The second parameter $0 < C \leq 1$ is the soft-margin constant. For $C = 1$ no outliers are invoked. If C is decreased towards 0 the number of outliers invoked increases.

Ben-Hur et al. [2] recommend to systematically increase q and C along a direction that guarantees a minimal number of SVs.

3. Global Alignment Kernels

3.1. Global Alignment Kernels

Cuturi et al. [6] introduced a new family of kernels for time series based on Global Alignments (GA), related to the well-known Dynamic Time Warping (DTW) similarity measure by [21]. DTW similarities are not positive semi-definite and [6] list some approaches to address this deficiency such as considering the square of the Gram matrices. The GA kernel proposed by [6] however is positive definite and was therefore chosen for our experiments.

Let x and y be two finite sequences of length $|x| = n$ and $|y| = m$. A DTW alignment is defined by a pair of ordered lists, each containing p elements. Each of the two lists contains unitarily increasing positive integers which encode the alignment. A DTW alignment may repeat elements of sequences in order to match the sequences onto each other. However, it cannot repeat elements of both sequences at the same time and it cannot skip elements. Loosely speaking, in an alignment the two sequences can be stretched at certain points to match them onto each other. To encode the alignment of two sequences we exploit the fact that time series are ordered and that for an element x_i of sequence x we can use its index i as its position within the sequence. A DTW alignment π of length $|\pi| = p$ between two sequences x and y can then be described as a pair of increasing p-tuples $\pi = (\pi_1, \pi_2) \in \mathbb{N}^{2p}$ such that

$$1 = \pi_1(1) \leqslant \cdots \leqslant \pi_1(p) = n, \tag{4}$$

$$1 = \pi_2(1) \leqslant \cdots \leqslant \pi_2(p) = m, \tag{5}$$

with unitary increments, i.e., for all $1 \leqslant i \leqslant p-1$ and for all $1 \leqslant j \leqslant p-1$,

$$\pi_1(i+1) \leqslant \pi_1(i) + 1 \quad \text{and} \quad \pi_2(j+1) \leqslant \pi_2(j) + 1 \tag{6}$$

$$\text{and} \quad (\pi_1(i+1) - \pi_1(i)) + (\pi_2(i+1) - \pi_2(i)) \geqslant 1, \tag{7}$$

where (7) means that no simultaneous repetitions are permitted.

To define the cost of an alignment a score function $S(\pi)$ is used:

$$S(\pi) = \sum_{i=1}^{p} \varphi\left(x_{\pi 1(i)}, y_{\pi 2(i)}\right) \tag{8}$$

where φ is an arbitrary conditionally positive-definite (c.p.d.) kernel such as e.g. $\varphi(x,y) = -||x-y||^2$ [3]. Exact DTW distances compute the cost $S(\pi)$ of all possible alignments and choose the alignment with minimal cost as a measure of similarity between two sequences. In contrast [6] consider all score values $\{S(\pi),\ \pi \in A(x,y)\}$ spanned by all possible alignments to define a kernel:

$$k_{GA}(x,y) = \sum_{\pi \in A(x,y)} e^{S(\pi)} = \sum_{\pi \in A(x,y)} \prod_{i=1}^{|\pi|} k\left(x_{\pi 1(i)}, y_{\pi 2(i)}\right) \tag{9}$$

where $k = e^{\varphi}$ is a suitable kernel.

Cuturi et al. argue that this notion of similarity provides a richer and more robust measure between two sequences since it quantifies the quality of both the optimal alignment and all the alignments which are close to it. The time complexity of the kernel k_{GA} is similar to the exact DTW score complexity which lies in $O(dnm)$ for two d-dimensional sequences of lengths n and m [5].

3.2. Triangular Global Alignment Kernels

Cuturi [5] extended the family of Global Alignment (GA) kernels to Triangular Global Alignment (TGA) kernels. This new kernel family is both positive definite and also faster to compute than adapted DTW similarities and the basic GA kernels. It adds further constraints on computed alignments in order to reduce runtime. This is done by introducing the triangle parameter T to the GA kernels. We will now briefly summarise the construction of the TGA kernels and refer the reader to [5] for details. [5] define and implement the TGA kernel as a special version of the k_{GA} kernel by combining the triangular kernel ω with the Gaussian kernel κ_σ to replace $k = e^{\varphi}$ in Eq. 9:

$$k(i,x;j,y) = \frac{\omega(i,j)\kappa_\sigma(x_i,y_j)}{2 - \omega(i,j)\kappa_\sigma(x_i,y_j)}, \tag{10}$$

where ω is defined as $\omega(i,j) = \max\left[\left(1 - \frac{|i-j|}{T}\right), 0\right]$, $T > 0$. The triangular kernel acts as a position kernel modulating the similarity of two points (x_i, y_j) by using their locations i and j in the series while κ_σ quantifies the similarity of x_i and y_j. The parameter T can now be used to control a band of alignments that will be considered, i.e. alignments at the end of the spectrum can be discarded. The implementation returns the original GA kernel with a slight variation of the standard Gaussian kernel if the parameter T is approximating

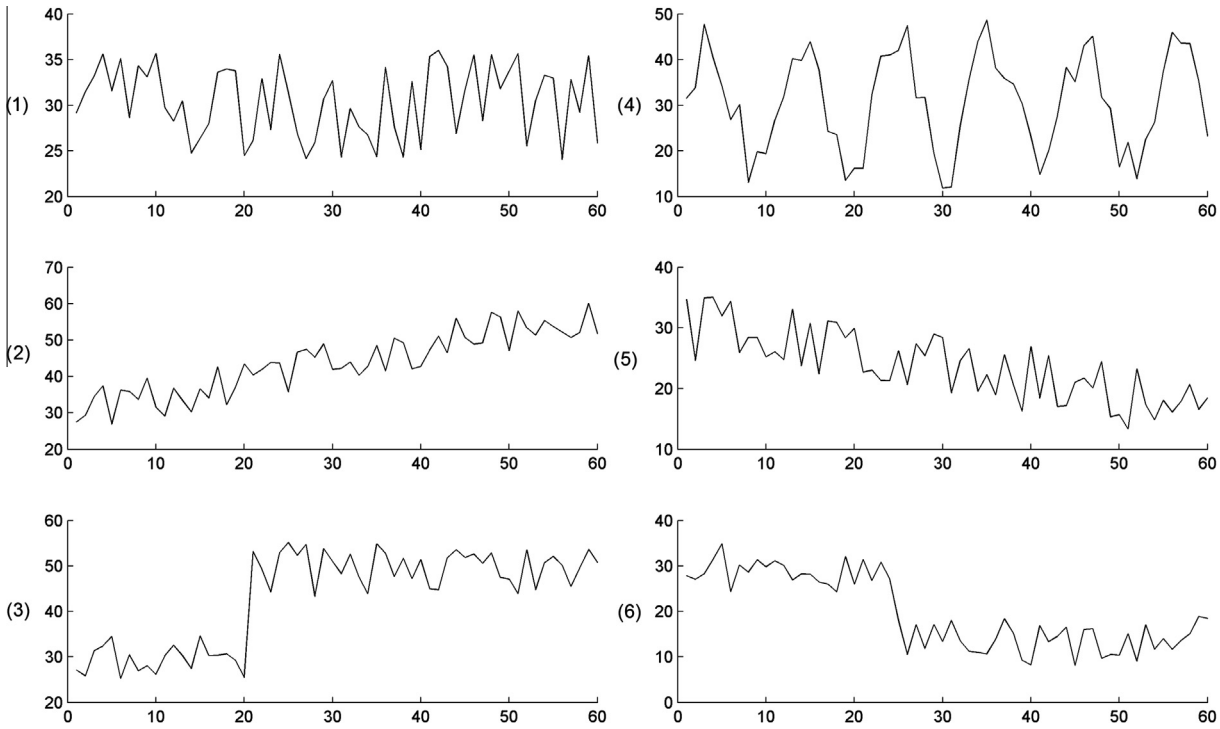

Fig. 1. Representative examples for each class of the CHART Dataset: (1) Normal, (2) Increasing trend, (3) Upward shift, (4) Cyclic, (5) Decreasing trend, (6) Downward shift. The sequences were generated artificially by six equations as described by [1].

0. $T = 1$ results in an element-wise comparison without alignments that can only compare sequences of the same length. For $T > 1$, only alignments for which $-T < \pi_1(i) - \pi_2(i) < T$ for all indices will be considered [5]. As T increases further the TGA kernel converges towards the GA kernel.[1]

4. Motivation and Setup

We have learned that a number of promising alternatives to the original SVC labelling stage have been proposed to improve the speed of the cluster labelling without sacrificing the quality of the partitioning. In comparison to faster clustering algorithms such as heuristic adaptations to kernel k-means clustering [7], the SVC algorithm stands out in its ability to find a partitioning without bias regarding the shape or number of clusters. The positive definite time series alignment kernel introduced by [5] enables us to exploit these advantages of the SVC algorithm to cluster time series. Literature on experiments using non-standard kernels with SVC as well as literature on clustering time series with SVC is very limited. We thus aim to cluster time series with SVC and the TGA alignment kernel to explore whether this combination produces meaningful results.

4.1. The Dataset

The experiments of the present study were conducted with the Synthetic Control Chart Time Series Data Set (CHART) obtained from the UCI Machine Learning repository [8]. It is a synthetically generated chart dataset containing 600 sequence vectors, each consisting of a sequence of 60 real valued elements. This dataset has been used in other clustering related research studies, making it possible to compare our results to some degree [29,4,28,16]. Each of the 600 time series belongs to one of six classes (see Fig. 1). Weng and Zhu [29] have already shown that the Euclidean distance cannot capture the similarity among the related time series. This is due to the fact that the Euclidean distance only captures pointwise differences and specific time series traits such as shifts or cyclic differences cannot be detected. [16] note that in particular the following pairs of classes of the CHART dataset have often been confused; normal/cyclic, decreasing trend/downward shift and increasing trend/ upward shift.

4.2. Validation

When clustering a benchmark set, there are several indices which measure how well the clustering partition $\mathbf{P}$ compares to the known optimum. With the intention of making our results comparable we use the same similarity indices as [4]. Chis et al. introduced a new evolutionary time series clustering algorithm (ETSC) and tested their algorithm on the CHART dataset. In order to assess the quality of the clustering result they computed three similarity coefficients; namely the Jaccard Score (Jaccard), the Rand statistic (Rand), and the Folkes and Mallow index (FM). We note that for all these indices a value close to 1 indicates great similarity and that all indices have a positive skew.

We calculate an additional similarity index used by [9,28] to further facilitate a comparison of our results with [28]. Given the partitioning result $\mathbf{P} = P_1, P_2, \ldots, P_n$ and the true clusters $\mathbf{C} = C_1, C_2, \ldots, C_N$ the cluster similarity can be computed by:

$$Sim(C, P) = \frac{\sum_{i=1}^{N} max_j Sim(C_i, P_j)}{n}, \tag{11}$$

where

$$Sim(C_i, P_j) = \frac{2|C_i \cap P_j|}{|C_i| + |P_j|}. \tag{12}$$

[1] Marco Cuturi provides more details on selecting parameters T and σ and an implementation of the Triangular Global Alignment (TGA) Kernel at URL: http://www.iip.ist.i.kyoto-u.ac.jp/member/cuturi/GA.html [Accessed 7 Feb 2014].

5. Experiments

We used SVC with the TGA kernel to cluster a subset of the CHART dataset aiming to study whether complex alignment kernels and SVC yield meaningful results on raw time series data. During our experiments the SEP-CG and CG labelling procedures produced almost identical results with CG sometimes producing false negatives. The bandwidth parameter σ of the TGA kernel was used in a similar fashion as the q parameter of the Gaussian kernel. As a reference the Gaussian kernel (RBF) was used to cluster the subset of the CHART data. The time series were normalised using two different methods: normalisation of the entire sequences and piecewise normalisation, following the approach of [9]. No promising partitions were obtained with either normalisation method when using the RBF kernel; however, for comparison purposes, the result with the greatest validity indices is shown in column 1 of Table 1.

5.1. GA Kernel

We begin our cluster analysis with the original GA kernel. Analogous to [2], we start with the lowest kernel width value and $C = 1$ (i.e. one cluster, no outliers), increasing the values until a stopping criterion is met. The validity indices of the following partitions can be observed in Table 1. A first stable and notable result was a partitioning of three clusters (see Fig. 2 and indices in column 3 of Table 1). It will be referred to as the "Super Cluster" result as it already contains an intuitive level of truth. It is noteworthy that this result was very stable over a wide range of the width parameter, i.e. a range of several numbers of SVs, a condition [2] state as an indicator of a good clustering result. This result was obtained without the invocation of outliers. The introduction of outliers did not lead to better results in any of the following experimental cases.

Continuing from the super cluster partitioning we increased the number of SVs and experienced further splitting of clusters. These partitions consisted of six clusters; the subset of the three super clusters, with three additional smaller clusters formed from noisy samples in cyclic, step-up and step-down sequences (see Table 1, Column 4). A further increase in the number of SVs eventually leads to a partition of nine clusters (see Table 1, Column 5). The additional clusters formed were a partition between normal and cyclic groups, while other clusters were partitions due to noisy samples. To the observer this behaviour appears understandable, as the sequences shown in Fig. 3 which separate and form their own cluster are very similar in the magnitude of the upward shift as well as the point at which the shift occurs.

Table 1
A comparison of Jaccard, Rand, and FM indices are presented for a standard Gaussian kernel (RBF), and a range of GA Kernels applied on the full CHART dataset. Column 1 shows the greatest index values obtained using a RBF kernel on normalised data which leads to 3 clusters comprising 447 support vectors. Columns 2–4 show partitions with generally twice the index values obtained using the GA kernel (where q values were incremented by steps of 10) and SEP-CG labelling applied on the complete un-normalised CHART dataset. The number of clusters increases with the amount of SVs used. The higher index values obtained in columns 2–4 suggest far better similarity to the benchmark partitioning compared to the RBF result.

SVC Kernel Parameters [q,C]	RBF [0.3,0.1]	GA [60–80,1]	GA [90–200,1]	GA [250,1]	GA [300,1]
# Clusters	3	1	3	6	9
# SVs	447	154-204	217-319	413	434
Jaccard index	0.2183	0.1667	0.5000	0.5116	0.6683
Rand index	0.3031	0.1653	0.8331	0.8331	0.8877
FM index	0.3900	0.4065	0.7053	0.6973	0.7634

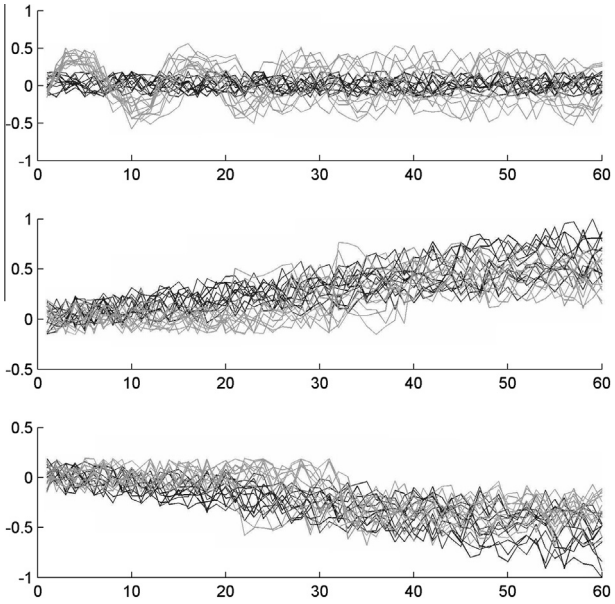

Fig. 2. Super cluster result using the GA and TGA kernels. One cluster contains all normal and cyclic sequences, a second cluster contains all downward shift and decreasing trend sequences while a third contains all upward shift and increasing trend sequences.

5.2. The TGA Kernel

A very similar behaviour is observed when clustering the data with the TGA kernel and varying window sizes, T. Analogous to the use of the GA kernel, the result of the three Super Clusters was also obtained with a small number of SVs. In contrast to the GA kernel, a small window leads to earlier separation of clusters with regard to the number of SVs. Furthermore with a smaller window, an increase of the width parameter q led to stronger separation within the benchmark clusters as more emphasis was put on local similarities. Increasing the number of SVs from the Super Cluster, a stable result of 4 clusters was obtained (see Table 2, column 4–5). In comparison to the partition results in columns 4–5 of Table 1 these obtained partitions were able to split cyclic sequences from the super cluster of all cyclic and normal sequences without any additional clusters formed from noisy samples. Results in columns 2–3 of Table 2 were able to separate the cyclic sequences completely from the normal sequences and form several clusters of their own while the other two super clusters remained unchanged. For an example, see Fig. 5. Judging from these results, it can be observed that a small window distinguishes the local similarities of the cyclic sequences but is unable to distinguish between shifts. A large window in contrast favours global similarities such as the overall trend or skewness due to the smoothing of all possible alignments.

From Table 2, it can be seen that with an increase of the window size, T, the number of SVs also increases. This trend of increasing SVs can be seen in Table 1, as the parameter q increases. This trend was repeated with partitions obtained from using $T = 4$ to 7, with $q = 90$ to 200. On the other-hand, with a decrease of the parameter C, the number of SVs increases and also the number of clusters dramatically increases, as seen in Table 3.

As general summary, the parameters q, C and T were varied across our series of experiments. When q was increased from 30 to 350 the numbers of support vectors increased proportionally. When C was increased from 0.01 to 1000, the number of support vectors decreased at an exponential rate. When T was increased from 0 to 30 the number of support vectors steadily decreased.

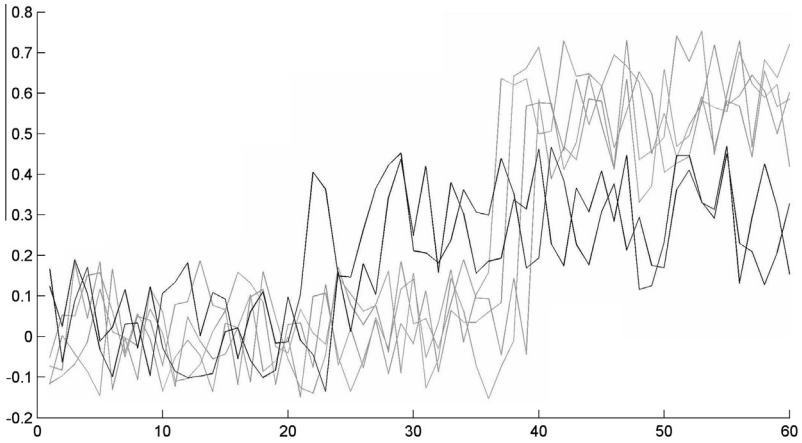

Fig. 3. Normalised data from class 5 (upward shift). With a large number of SVs the grey sequences separate from the cluster containing the upward trend and upward shift sequences and form a new cluster. The two black sequences are representatives of upward shift sequences that did not separate from the initial cluster.

Table 2
The table shows the partitions obtained using SVC with the TGA kernel and SEP-CG labelling (Column 2–6) compared to the ETSC clustering result by [4] (Column 1). Using SVC, increasing parameter T decreases the number of SVs and the number of partitions found.

Method	ETSC	SVC TGA $T=4$	SVC TGA $T=5$	SVC TGA $T=6$	SVC TGA $T=7$	SVC TGA $T=10$
[q,C]	–	[100,1]	[100,1]	[100,1]	[100,1]	[100,1]
Clusters	8,10,12	6	5	4	4	3
SVs	–	385	353	325	310	261
Jaccard	0.4425	0.6250	0.6650	0.6667	0.6667	0.5000
Rand	0.8512	0.8781	0.8882	0.8887	0.8887	0.8331
FM	0.6155	0.7378	0.7712	0.7730	0.7730	0.7053

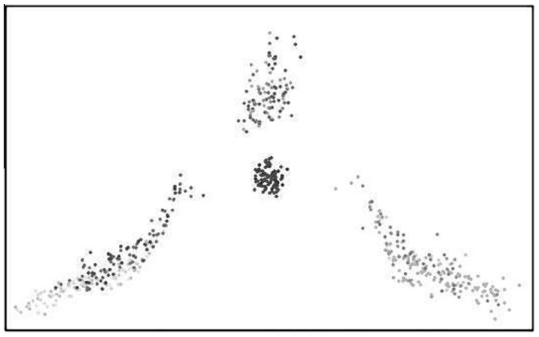

Fig. 4. Isomap visualisation with $k=300$ nearest neighbours results in 4 distinct clusters. Top cluster: class of cyclic series; centre cluster: class of normal series; left cluster: classes of decreasing series (dark) and step-down series (bright); right cluster: classes of increasing series (dark grey) and step-up series (bright grey).

Generally, as the number of support vectors increased, the number of clusters found also increased. At the extremes of these parameter ranges (i.e. large q values, small C and T values) large numbers of support vectors and consequently a large number of clusters occurred. However, none of the experiments with variations of T, q, or C could split the two large remaining superclusters and separate downward shift from decreasing trend or upward shift from increasing trend.

In order to find evidence that this result is sensible we employed isomap [26] to generate a visualisation of the CHART data and this resulted in the same superclusters; in the isomap visualisation with $k=300$ nearest neighbours four distinct clusters became visible (see Fig. 4): The top cluster showing the cyclic class is separated from the centre cluster showing the normal class. The left (super) cluster includes the decreasing series (dark dots) and the downward shift series (light grey dots), and the right (super) cluster shows the increasing series (dark grey) and upward shift series (light grey dots). The close proximity of the series included in a super cluster in this visualisation indicates that these series are very similar.

Our experiments employed workstations with Intel Xeon 2.13 GHz six core CPUs and 48 GB RAM. There is scope for parallelization however our experiments were conducted with single core implementations only. The computationally most expensive scenario partitioning a full CHART dataset took about 22 h while we tested extreme parameter cases. However most iterations were completed between 1 and 12 h. Comparable experiments using the standard Gaussian kernel required about 40 min, but were unable to capture the inherent structure. For more information regarding the time complexity using the TGA kernel, the reader is referred to [5].

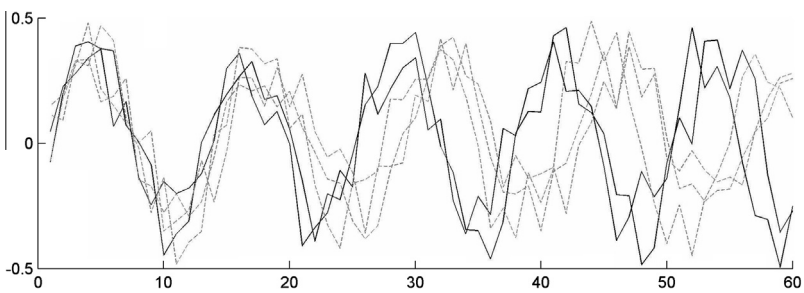

Fig. 5. A plot of normalised data of some cyclic sequences. With a large number of SVs, the dotted and solid sequences separate from the super cluster containing all cyclic and normal sequences. The dotted sequences are contained in a new, separate cluster. The solid sequences also form a new, separate cluster.

Table 3
Variation of C: The table shows the partitions obtained using SVC with the TGA kernel and SEP-CG labelling varying the parameter C on full CHART dataset. Decreasing parameter C, increases the number of SVs and varies exponentially the number of partitions found.

Method	SVC TGA $T = 6$	SVC TGA $T = 6$	SVC TGA $T = 6$	SVC TGA $T = 6$	SVC TGA $T = 6$
[q,C]	[100,10-0.5]	[100,0.4]	[100,0.3]	[100,0.2]	[100,0.1]
Clusters	4	5	11	71	247
SVs	325	325	326	336	373
Jaccard	0.6667	0.6650	0.6600	0.4483	0.3000
Rand	0.8887	0.8882	0.8866	0.8753	0.8675
FM	0.7730	0.7712	0.7628	0.6883	0.5374

6. Discussion

Using the approach of clustering the CHART dataset with SVC and a TGA kernel we achieve the super cluster partitioning containing an intuitive notion of similarity and dissimilarity. The splitting and forming of new clusters from the super cluster solution indicates that SVC with a GA and TGA kernel is in some way able to distinguish more elaborate differences such as cyclic sequences from normal sequences and shifts from trends. In these new segmentations, several cyclic sequences form their own clusters as they split from the super clusters. However these segmentations overfit the data as we ultimately aim to have all instances of one class in the same respective cluster separated from the other classes. We recall that [4,16] already mention that these classes are easily confused, which makes an ideal clustering a very difficult task.

Table 2 shows a comparison of the accuracy of our results with regard to the different benchmark indices with the results obtained by [4]. We also calculated the accuracy statistic of Eqs. 11 and 12 used by [28] on their segmentations of the CHART data (see Fig. 6). The key differences are that Wang and Megalooikonomou used a Partitioning Around Medoids [10] algorithm and thus set the number of clusters k a priori. Their reported accuracy values are the average accuracy over 10 clustering iterations. The comparison to both of these literature results and the relevant validity indices shows that the quality of the partitioning using SVC on raw time series data is competitive. The segmentations we obtained were very stable within intervals of several numbers of SVs, which serves as an indicator in choosing relevant segmentations when clustering real world datasets. As expected, we saw that the GA kernel smoothens the alignments and favours global similarities compared to a TGA kernel with a small window size.

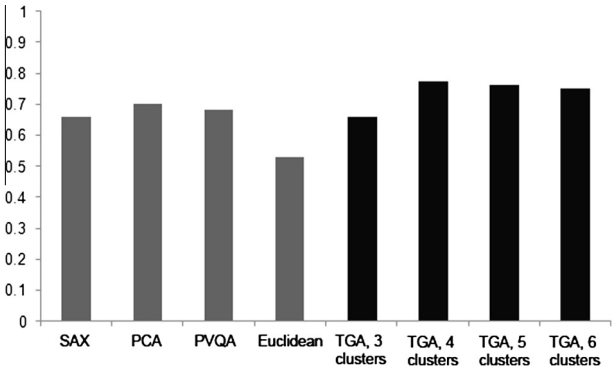

Fig. 6. A comparison of the accuracy index (Eqs. 11 and 12) of clustering results of the CHART dataset [28] (light grey) and some of our results (dark grey) obtained with SVC and the TGA kernel from Table 2. Wang and Megalooikonomou focused on 3 different piecewise dimensionality reduction techniques and a Partitioning Around Medoids [10] algorithm. The results suggest that our raw-data approach using SVC with a TGA kernel is competitive with other approaches. The TGA result with 4 clusters achieved an accuracy index of 0.7778, an increase of approximately 7% when compared with PCA.

In the context of SVC the kernel is not only used during the computation of the minimal enclosing sphere but also during the cluster labelling process as parts of the calculations are carried out in feature space. Since the labelling stage was already identified as a bottleneck of the SVC algorithm, it performs yet slower using such a complex kernel, which is an obvious disadvantage. However a big advantage is that the strengths of SVC in finding clusters of arbitrary number and shape can also be used on such complex datasets as the CHART data. In addition, raw data can be clustered without pre-processing. Moreover, incorporating knowledge of properties of the dataset may help in increasing clustering quality and performance, e.g. in choosing an appropriate window size T for the TGA Kernel.

7. Conclusion

We have found that raw data clustering of time series with SVC provides a competitive alternative to existing clustering approaches with regard to the quality of the clustering results. The results indicate that using SVC with specialised kernels for structured data can provide a competitive alternative to existing clustering approaches on structured data such as time series. The most important challenges concern the development of faster approaches to the cluster description stage, faster labelling strategies in SVC as well as improved normalised kernels for time series to reduce runtime. Further important tasks involve establishing validity measures and stopping criteria that can be used on real datasets where no benchmark exists. A general disadvantage of non-parametric techniques is their lack of transparency. With improvements in hardware and the potential of parallel implementations, the currently still extensive running times can be improved in a straightforward way. This will allow for more systematic experimentation with variations of the kernel and other time series data sets.

References

[1] R.J. Alcock, Y. Manolopoulos, Time-series similarity queries employing a feature-based approach, in: 7th Hellenic Conference on Informatics, Ioannina, 1999, pp. 27–29.
[2] A. Ben-Hur, D. Horn, H.T. Siegelmann, V. Vapnik, Support vector clustering, J. Mach. Learn. Res. 2 (2001) 125–137.
[3] F. Camastra, A. Vinciarelli, Machine Learning for Audio, Image and Video Analysis. Theory and Applications, Advanced Information and Knowledge Processing, Springer-Verlag, London, 2008.
[4] M. Chiș, S. Banerjee, A. Hassanien, Clustering time series data: an evolutionary approach, foundations of computational intelligence, in: A. Abraham, A.E. Hassanien, A. Leon F. de Carvalho, V. Snášel (Eds.), Studies in Computational Intelligence, vol. 206, 2009, pp. 193–207.
[5] M. Cuturi, Fast global alignment kernels, in: Proceedings of the 28th International Conference on Machine Learning, ACM, New York, NY, USA, 2011, pp. 929–936.
[6] M. Cuturi, J.P. Vert, O. Birkenes, T. Matsui, A kernel for time series based on global alignments, in: IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 2, 2007, pp. 413–416.
[7] M. Filippone, F. Camastra, F. Masulli, S. Rovetta, A survey of kernel and spectral methods for clustering, Pattern Recognit. 41 (1) (2008) 176–190.
[8] A. Frank, A. Asuncion, UCI machine learning repository, University of California, Irvine, School of Information and Computer Sciences, 2010.
[9] M. Gavrilov, D. Anguelov, P. Indyk, R. Motwani, Mining the stock market (extended abstract): which measure is best?, in: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2000, pp 487–496.
[10] L. Kaufman, P.J. Rousseeuw, Partitioning around medoids (Program PAM), in: Finding Groups in Data: An Introduction to Cluster Analysis, John Wiley & Sons, 2008, pp. 68–125.
[11] B.Y.S. Khanloo, D. Dargahi, N. Aghaeepour, A. Masoudi-Nejad, Support vector clustering: from local constraint to global stability, Foundations of Computational Intelligence, vol. 6, Springer, 2009, pp. 209–227.
[12] J. Lee, D. Lee, An improved cluster labeling method for support vector clustering, IEEE Trans. Pattern Anal. Mach. Intell. 27 (3) (2005) 461–464.

[13] J. Lee, D. Lee, Dynamic characterization of cluster structures for robust and inductive support vector clustering, IEEE Trans. Pattern Anal. Mach. Intell. 28 (11) (2006) 1869–1874.
[14] S.H. Lee, K. Daniels, Gaussian kernel width exploration and cone cluster labeling for support vector clustering, Pattern Anal. Appl. 15 (3) (2012) 327–344.
[15] S.H. Lee, K.M. Daniels, Cone cluster labeling for support vector clustering, in: Proceedings of 6th SIAM Conference on Data Mining, 2006, pp. 484–488.
[16] Y. Liu, D. Tian, B. Li, A wireless intrusion detection method based on dynamic growing neural network, in: Proceedings of the First International Multi-Symposiums on Computer and Computational Sciences, IEEE Computer Society, Washington, DC, USA, 2006, pp. 611–615.
[17] C.-S. Perng, H. Wang, S.R. Zhang, D.S. Parker, Landmarks: a new model for similarity-based pattern querying in time series databases, Proceedings of the 16th International Conference on Data Engineering (ICDE'00), IEEE, San Diego, CA, USA, 2000, pp. 33–42.
[18] L. Ping, Z. Chun-Guang, Z. Xu, Improved support vector clustering, Eng. Appl. Artif. Intell. 23 (4) (2010) 552–559.
[19] Y. Ping, H. Li, Y. Zhang, Z. Zhang, A new perspective of support vector clustering with boundary patterns, in: Z.H. Zhou, F. Roli, J. Kittler (Eds.), Multiple Classifier Systems, Springer, 2013, pp. 224–235.
[20] Y. Ping, Y.J. Tian, Y.J. Zhou, Y.X. Yang, Convex decomposition based cluster labeling method for support vector clustering, J. Comput. Sci. Tech. 27 (2) (2012) 428–442.
[21] H. Sakoe, S. Chiba, Dynamic programming algorithm optimization for spoken word recognition, IEEE Trans. Acoust. Speech Signal Process. 26 (1) (1978) 43–49.
[22] B. Schölkopf, J.C. Platt, J.C. Shawe-Taylor, A.J. Smola, R.C. Williamson, Estimating the support of a high-dimensional distribution, Neural Comput. 13 (7) (2001) 1443–1471.
[23] B. Schölkopf, R.C. Williamson, A.J. Smola, J. Shawe-Taylor, J.C. Platt, Support vector method for novelty detection, in: S.A. Solla, T.K. Leen, K.R. Müller (Eds.), Advances in Neural Information Processing Systems, vol. 12, MIT Press, Cambridge, MA, 2000, pp. 582–588.
[24] D.M.J. Tax, R.P.W. Duin, Support vector domain description, Pattern Recognit. Lett. 20 (11–13) (1999) 1191–1199.
[25] F.E.H. Tay, L. Cao, Application of support vector machines in financial time series forecasting, Omega 29 (4) (2001) 309–317.
[26] Joshua B. Tenenbaum, Vin De Silva, John C. Langford, A global geometric framework for nonlinear dimensionality reduction, Science 290 (5500) (2000) 2319–2323.
[27] V.N. Vapnik, The Nature of Statistical Learning Theory, Springer-Verlag New York Inc., New York, NY, USA, 1995.
[28] Q. Wang, V. Megalooikonomou, A dimensionality reduction technique for efficient time series similarity analysis, Inf. Syst. 33 (1) (2008) 115–132.
[29] Y.J. Weng, Z.Y. Zhu, Time series clustering based on shape dynamic time warping using cloud models, 2003 International Conference on Machine Learning and Cybernetics, IEEE, 2003, pp. 236–241.
[30] J. Yang, V. Estivill-Castro, S.K. Chalup, Support vector clustering through proximity graph modelling, in: L. Wang, J. Rajapakse, K. Fukushima, S. Young Lee, X. Yao (Eds.), Proceedings of the 9th International Conference on Neural Information Processing, IEEE, 2002, pp. 898–903.