# Improved piecewise vector quantized approximation based on normalized time subsequences

**Article** *in* Measurement · November 2013

DOI: 10.1016/j.measurement.2013.05.012

Hailin Li
Huaqiao University

Chonghui Guo
Dalian University of Technology

ELSEVIER

ISSN 0263–2241
Volume 46, Issue 9
November 2013

# Measurement

***SPECIAL SECTIONS:***

**IMEKO TC21 Special Section on Mathematical and Statistical Tools in Measurement**

**Special Section on Mathematical Models in Measurement**

Official Journal of the
**International
Measurement
Confederation**

IMEKO

Measurement 46 (2013) 3429–3439

Contents lists available at SciVerse ScienceDirect

Measurement

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

# Improved piecewise vector quantized approximation based on normalized time subsequences

Hailin Li [a,b,*], Libin Yang [a], Chonghui Guo [b]

[a] College of Business Adminstration, Huaqiao University, Quanzhou 362021, China
[b] Institute of Systems Engineering, Dalian University of Technology, Dalian 116024, China

ARTICLE INFO

*Article history:*
Received 21 April 2012
Received in revised form 25 March 2013
Accepted 13 May 2013
Available online 29 May 2013

*Keywords:*
Time series data mining
Piecewise vector quantized approximation
Distance measure
Normalized time sequence

ABSTRACT

Piecewise vector quantized approximation (PVQA) is a dimensionality reduction technique for time series data mining, which uses the closet codewords deriving from a codebook of key subsequences with equal length to represent the long time series. In this paper, we proposed an improved piecewise vector quantized approximation (IPVQA). In contrast to PVQA, IPVQA involves three stages, normalizing each time subsequence to remove the mean, executing the traditional piecewise vector quantized approximation and designing a novelly suitable distance function to measure the similarity of time series in the reduced space. The first stage deliberately neglects the vertical offsets in the target domain so that the ability of the codebook obtained from the training dataset is more powerful to represent the corresponding subsequences. The new function based on Euclidean distance in the last stage can effectively measure the similarity of time series. Experiments performing the clustering and classification on time series datasets demonstrate that the performance of the proposed method outperforms PVQA.

## 1. Introduction

A time series is usually composed of a series of data points, each of which represents a value at a certain time. Time series mining is an important and hot topic in the field of data mining, and recently has received much attention due to its wide application such as finance, medicine and atmosphere. Conventionally, the traditional techniques of data mining, including clustering methods, classification approaches, rule discovery and so on, are not suitable for mining time series because the high dimensionality of time series goes against the similarity (dissimilarity) computation. If the distance is directly measured on the original time series, not only much more time is consumed but also an inaccurate result is possibly retrieved. Thereby, dimensionality reduction is generally the preliminary work of similarity computation for time series data mining.

Two basic approaches are used to reduce the dimensionality, i.e. a piecewise discontinuous function and a low-order continuous function. The former mainly includes discrete wavelet transform (DWT)[1,2], piecewise linear approximation (PLA)[3,4], piecewise aggregate approximation (PAA)[5,6], symbolic aggregate approximation (SAX)[7,8] and their extended versions[9–11]. The later mainly includes non-linear regression, singular value decomposition (SVD) [12] and discrete fourier transform (DFT)[13]. After dimensionality reduction by the above techniques, the Euclidean distance can be used to measure the similarity of the long time series. In particular, if there is no prior knowledge about the data, the piecewise vector quantized approximation (PVQA) [14,15] based on piecewise constant approximation is more effective to represent time series than the above traditional methods. Moreover, it is a more flexible approximation of each sequence in the low reduced dimension. Especially, The multiresolution

[*] Corresponding author at: College of Business Adminstration, Huaqiao University, Quanzhou 362021, China. Tel.: +86 15909855612.
*E-mail address:* dr.lihailin@gmail.com (H. Li).

0263-2241/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.measurement.2013.05.012

vector quantized approximation [16,17] resolves the potential problems when only one codebook is provided to represent time subsequences.

As PVQA is one of the most excellent approximation methods for dimensionality reduction, we pay much more attention on this method to improve the performance of the approximation. Our work is to use the normalized subsequences (piecewise time series) in the training phase to obtain a codebook with more powerful ability of approximation than that generated by PVQA. After generating the codebook from the training normalized time subsequences, the test time series are also segmented into some subsequences with the length being equal to that of codewords in the codebook, and the subsequences also need to be normalized by subtracting their own mean. The second stages of the approximation are the same with PVQA. Due to the normalized stage, the reconstruction of time series by the codewords should reconsider the mean of the corresponding subsequence. Moreover, the distance function used by PVQA to measure time series should be remade to adapt to the new condition. Experiments on several time series datasets, performing the approximation, hierarchical clustering and classification, demonstrate that the codebook generated by the normalized time subsequences and the new distance function for time series can improve the performance of the original PVQA. We call this method Improved Piecewise Vector Quantized Approximation (IPVQA).

The remainder of the paper is organized as follows. In Section 2, we give a brief introduction on vector quantization. The framework of the improved version (IPVQA) is presented in Section 3. We provide some experiments on serval time series datasets in Section 4. In the last section we conclude our work and discuss the future work.

## 2. Vector quantization

Vector quantization (VQ) [18,19] is a lossy compression method based on the principle of block coding and it is often applied to signal compression and data hiding [20,21]. The VQ design problem can be illustrated as follows. Given a vector source $X$ with many sequences, a distortion measure $f$ and the number $S$ (size) of codewords (or code vectors), search a codebook $C$ and a partition which result in the smallest average distortion we have. As showed in Fig. 1, the dots are the training sequences of length 2 in the vector source $X$, the set of red stars represents the codebook $C$ obtained by VQ, the green lines which look like the cobweb constitute a partition.

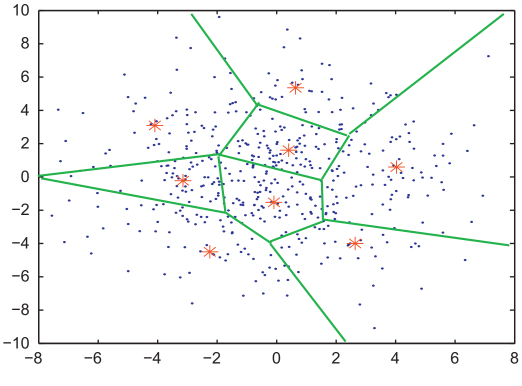


**Fig. 1.** A codebook with eight codewords is obtained by VQ.

Now we give the formalized expression for the above terms.

Each training sequence $V_i$ is a member of the vector source $V$, i.e., $V = \{V_1, V_2, \ldots, V_M\}$, where $V_i = \{v_{i1}, v_{i2}, \ldots, v_{iK}\}$, $i = 1, 2, \ldots, M$.

Let $S$ be the size of codebook $C = \{C_1, C_2, \ldots, C_S\}$, the length $K$ of each codeword is the same to that of the training sequences, i.e., $C_s = \{c_{s1}, c_{s2}, \ldots, c_{sK}\}$, $s = 1, 2, \ldots, S$.

Let $R_s$ be an encoding region and $R = \{R_1, R_2, \ldots, R_S\}$ denotes the set of the partitions of the training sequences. If the training sequence $V_i$ locates the region $R_s$, then its codeword is $C_s$, i.e., $g(V_i) = C_s$, if $V_i \in R_s$.

The average distortion measure $f$ is given by:

$$f(V,S) = \frac{1}{MK}\sum_{i=1}^{M}\|V_i - g(V_i)\|^2, \quad if \quad V_i \in R_s, s = 1, 2, \cdots, S. \tag{1}$$

Consequently, the VQ design problem is summarized that given $V$ and $S$, find $C$ and $R$ such that $f(V,S)$ is minimized.

If we find a solution about $C$ and $R$ to minimize the $f(V,S)$, then it should satisfy two conditions [15,17]: (a) nearest neighbor condition and (b) centroid condition.

It is easy to find that VQ design problem is an iterative method to produce an optimal codebook from a set of training sequences. The relative algorithm description can be seen in the paper [18] or the website [22].v

## 3. Improved piecewise vector quantized approximation

To improve the performance of the time series representation, we propose an improved piecewise vector quantized approximation (IPVQA) based on the original PVQA. Simultaneously, we also provide a novel distance function. The method involves three stages: (a) normalizing time subsequences, (b) executing the original PVQA and (c) designing a new distance function for time series similarity measure.

### 3.1. Normalizing time subsequences

We regard the first stage (normalizing time subsequences) as the preprocessing step of IPVQA. There are three questions need us to answer, i.e., (a) why do the time subsequences need to be normalized? (b) How does the method normalize it? (c) Can the normalization improve the performance of the method?

Given a time series $Q = \{q_1, q_2, \ldots, q_K\}$, if we segment it into $W$ partitions equally, we say that each partition of length $l(l = \frac{K}{W})$ is a time subsequence.

The authors [15] recently has shown that the distance function of the original PVQA is based on the Euclidean distance $D(X,Y)$ between two time subsequences $X = \{x_1, x_2, \ldots, x_K\}$ and $Y = \{y_1, y_2, \ldots, y_K\}$, i.e.,

$$D(X,Y) = \sqrt{\sum_{i=1}^{K}(x_i - y_i)^2} \tag{2}$$

In fact, a shortcoming of Euclidean distance [1,23,24] is shown as in Fig. 2. Directly perceived through the senses, the two time subsequences $X$ and $Y$ are very similar because subsequence $X$ can be obtained by shifting up vertically subsequence $Y$. However, if we use Eq. (2) to measure the distance, they will be considered as the dissimilar ones. Thereby, we should either normalize time series or time subsequences in IPVQA to resolve the problem.

Now we change the distance function into another one, i.e.,

$$D'(X,Y) = \sqrt{\sum_{i=1}^{K}((x_i - \overline{X}) - (y_i - \overline{Y}))^2}, \tag{3}$$

where $\overline{X} = \frac{1}{K}\sum_{i=1}^{K} x_i$ and $\overline{Y} = \frac{1}{K}\sum_{i=1}^{K} y_i$.

If we use Eq. (3) to measure the two time subsequence as shown in Fig. 2, the result demonstrates that they are similar due to the neglect of their vertical offsets. In order to generalize a more representative codebook, we normalize time subsequences and transform the vector source of the training subsequences $X$ in IPVQA into another form $X'$ by

$$X' = X - \overline{X}. \tag{4}$$

By Eq. (4), we regard $X'$ as the vector source of the training sequences in IPVQA and make sure that a more accurate codebook can be obtained in the training step of IPVQA. Because there is a better codebook to describe those time subsequences, IPVQA can approximate time series better. So far, we have already answered the above three questions.

### 3.2. Codebook generation

Suppose there is a training time series dataset $R$ of size $M \times K$ and a test time series dataset $E$ of size $N \times K$ with a large $K$. According to the idea of paper [25,26], we can segment a long time series into some time subsequences of equal length $l$. In other word, we can preset the dimensionality $W$ in reduced space. The length of all time subsequences in this reduced space is $l$, where $l = \frac{K}{W}$. Generally, since $W \ll K$,$l$ must be greater than 1, i.e., $l > 1$.

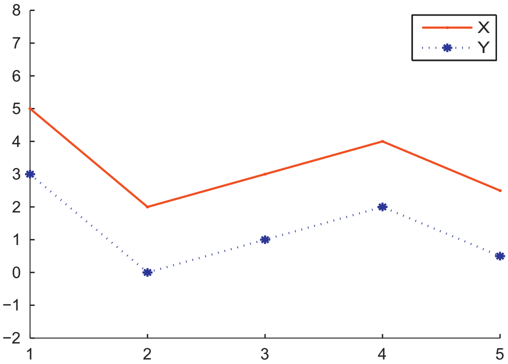


**Fig. 2.** The two time subsequences are similar except for the offsets.

After the segmentation of each long time series in dataset $R$ and $E$, the training time subsequences $R'$ of size $M \times W \times l$ and the test time subsequences $E'$ of size $N \times W \times l$ are obtained respectively. It means that there are $M \times W$ training time subsequences in dataset $R'$ and $N \times W$ test time subsequences in dataset $E'$. The length of each time subsequence in $R'$ and $E'$ is $l$.

PVQA directly uses the training subsequences dataset $R'$ to generate a codebook (as shown in Fig. 3), whose quality is better than the mean of time sequence in PCA [8,27]. However, since the two similar time subsequences shown in Fig. 2 maybe respectively locate in two different groups which are represented by two different codewords of the codebook, the quality of the codebook generated by PVQA is not the best.

In IPVQA, the normalizing stage addresses the above problem. Before the codebook generation, we should normalize the original time series dataset $R$ and the time subsequences dataset $R'$ respectively. For $R$, we use Eq. (5) to normalize and renew each time series in the dataset $R$.

$$R(i,:) = \frac{R(i,:) - mean(R(i,:))}{std(R(i,:))} \tag{5}$$

where $i = 1, 2, \ldots, M$. In the above formula, $R(i,:)$ denotes the $i$th time series in dataset $R$, $mean(\cdot)$ and $std(\cdot)$ respectively represent the mean function and standard deviation function.

After normalizing the training time series, we obtain normalized training time series $R$. Through segmenting time series, the training time subsequences dataset $R'$ is further obtained. Next, we execute the stage of normalizing time subsequences to renew the training time subsequences dataset $R'$. These normalized subsequences in the training process to generalize the codewords (key subsequences) can improve the quality of the codebook as shown in Fig. 4. It means that the quality of the codebook generalized by IPVQA is better than that generalized by PVQA. In particular, the mean of each codeword in codebook generalized by IPVQA is close to 0 but that generalized by PVQA is not (as shown in Figs. 3 and 4), which also demonstrates that the generation process of codebook by IPVQA considers the shape feature and neglects the vertical offsets.

### 3.3. Time series coding

After a group of codewords which constitute a codebook are constructed, we can use the codewords to represent each time series in the dataset including the training one and the test one. Without loss of generality, in this subsection we only acquire a new representation of each time series in the test time series dataset. From the previous subsection we could obtain the time subsequences dataset $E'$. The length of each time subsequence is equal to that of the codeword in the codebook.

Given a time series $X$ of length $K$ composed of $W$ time subsequences of length $l = \frac{K}{W} > 1$, i.e., $X = \{X_i, X_2, \ldots, X_W\}$ - $\in E$, and given a codebook $C = \{C_1, C_2, \ldots, C_S\}$ produced by the stage of codebook generation in the training dataset $R$, we can obtain the normalized time subsequences

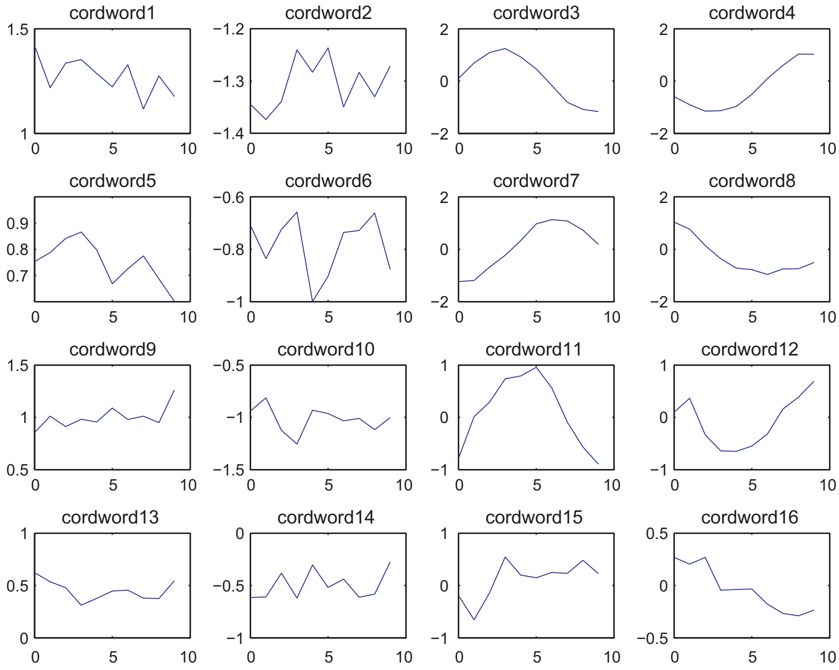


**Fig. 3.** The codebook generalized by PVQA in the Control Chart dataset when $S = 16$ and $W = 6$.

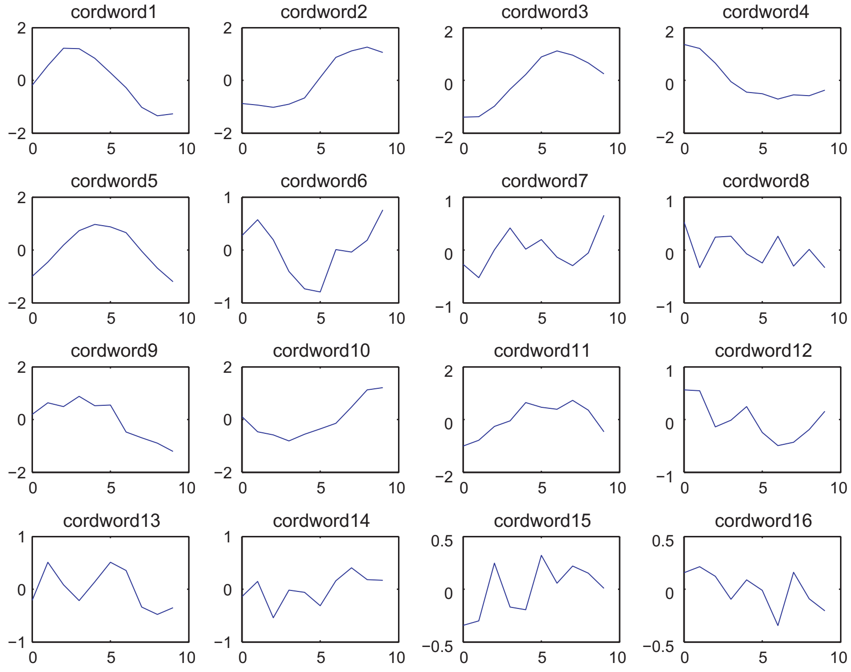


**Fig. 4.** The codebook generalized by IPVQA in the Control Chart dataset when $S = 16$ and $W = 6$.

$X' = \{X'_1, X'_2, \ldots, X'_W\}$ and the mean sequences $\overline{X} = \{\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_W\}$, where $X'_i \in E'$, then the codeword series of the time series $X$ is $X'' = \{X''_1, X''_2, \ldots, X''_W\}$, where $X''_i \in C$ and

$$X''_i = \arg_j min(D(X'_i, C_j)). \tag{6}$$

If we reconstruct the time series $\widehat{X}$ directly using the corresponding codewords in IPVQA as same as in PVQA, then the reconstructed one cannot approximate the original one well. The reason is that the codeword $X''_i \in X''$ are lack of the mean value $\overline{X}_i$ of the corresponding subsequence in $X$. Thereby, during the reconstruction of time series, we should consider the mean value which was pre-

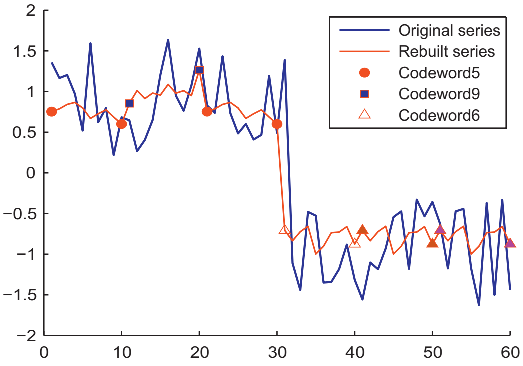


**Fig. 5.** The reconstruction of time series is rebuilt by three different codewords using PVQA.

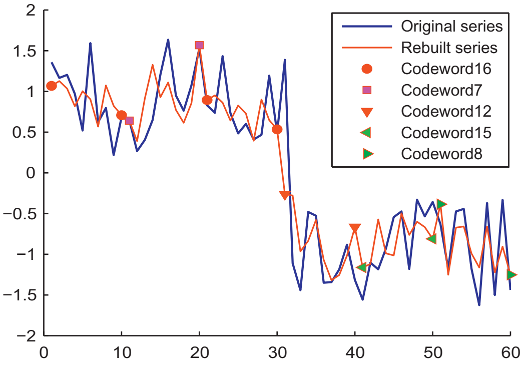


**Fig. 6.** The reconstruction of time series is rebuilt by 5 different codewords using IPVQA.

viously subtracted by the normalizing process, i.e., $\widehat{X}_i = X''_i + \overline{X}_i$, where $\widehat{X}_i \in \widehat{X}$. Comparing to the reconstruction with PVQA and IPVQA shown in Figs. 5 and 6, the indexes of codewords by PVQA are $\{5,9,5,6,6,6\}$ shown in Fig. 3 and the indexes of codewords by IPVQA are $\{16,7,16,12,15,8\}$ shown in Fig. 4. The time series is represented by five members $(7,8,12,15,16)$ of the codebook in IPVQA rather than by three members $(5,6,9)$ of the codebook in PVQA. It means that more information is provided to represent the time series by IPVQA. Intuitively, the reconstructed one generated by IPVQA can keep the features of the original time series and imitate the trend better than that generated by PVQA. It means that the approximation to the original time series using IPVQA is better than that using PVQA because of the smaller approximation error in IPVQA.

### 3.4. Distance measure

In Section 3.1, we know that the Euclidean distance is a popular distance measure and suitable to measure the distance between two new representations whose dimensionality is much smaller.

After generalizing a codebook and encoding the test time series using the codebook, we can obtain the two codeword series, $X'' = \{X''_1, X''_2, \ldots, X''_W\}$ and $Y'' = \{Y''_1, Y''_2, \ldots, Y''_W\}$, which respectively evolve from two original time series $X$ and $Y$ in test dataset.

In PVQA, $X''$ and $Y''$ directly evolve from the original time series without the process of normalizing subsequences. Its distance measure is Euclidean distance

$$PVQA\_RoughDist(X,Y) = \sqrt{\sum\nolimits_{i=1}^{W} (D(X''_i, Y''_i))^2}. \tag{7}$$

However, in IPVQA, $X''$ and $Y''$ evolve from the normalized subsequences $X'$ and $Y'$. When designing the distance measure, we should reconsider the mean of the corresponding original subsequence as the reconstruction of time series does.

Given two time series $X = \{X_1, X_2, \ldots, X_W\}$ and $Y = \{Y_1,$ - $Y_2, \ldots, Y_W\}$, where $X_i$ (or $Y_i$) of length $l$ is the $i$th subsequence of $X$ (or $Y$) of length $K = l * W$ and $l > 1$. Record the mean value of each subsequence, $\overline{X} = \{\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_W\}$ and $\overline{Y} = \{\overline{Y}_1, \overline{Y}_2, \ldots, \overline{Y}_W\}$, obtain the codeword series $X'' = \{X''_1, X''_2, \ldots, X''_W\}$ and $Y'' = \{Y''_1, Y''_2, \ldots, Y''_W\}$, where $X''_i = \{x''_{i1}, x''_{i2}, \ldots, x''_{il}\}$ and $Y''_i = \{y''_{i1}, y''_{i2}, \ldots, y''_{il}\}$, finally rebuilt the two time sequences $\widehat{X}$ and $\widehat{Y}$, i.e., $\widehat{X}_i = X''_i + \overline{X}_i$ and $\widehat{Y}_i = Y''_i + \overline{Y}_i$, where $\widehat{X}_i = \{\hat{x}_{i1}, \hat{x}_{i2}, \ldots, \hat{x}_{il}\}$ and $\widehat{Y}_i = \{\hat{y}_{i1}, \hat{y}_{i2}, \ldots, \hat{y}_{il}\}$. The distance measure for the IPVQA is

$$IPVQA\_RoughDist(X,Y) = \sqrt{\sum\nolimits_{i=1}^{W} (D(\widehat{X}_i, \widehat{Y}_i))^2} \tag{8}$$

In order to conveniently compute the distance measure, In PVQA the distance between any two codewords in the codebook can be pre-computed and stored in a matrix [15] because the codeword used to represent the corresponding time subsequence is from the fixed codebook of size $S$. This matrix is

$$DM = \begin{pmatrix} 0 & d_{12} & d_{13} & \cdots & d_{1S} \\ d_{21} & 0 & d_{23} & \cdots & d_{2S} \\ d_{31} & d_{32} & 0 & \cdots & d_{3S} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ d_{S1} & d_{S2} & d_{S3} & \cdots & 0 \end{pmatrix}.$$

Then the distance measure in PVQA can be rewritten as

$$PVQA\_RoughDist'(X,Y) = \sqrt{\sum\nolimits_{i=1}^{W} (DM(ind(X''_i), ind(Y''_i)))^2}, \tag{9}$$

where $ind(Z)$ returns the index of codeword $Z$ in the codebook.

In our method IPVQA, we can also use this matrix to conveniently compute the distance. In addition, another matrix $EM$ is necessary, which records the mean values of the time subsequences. For instance, the $EM$ of two time series $X$ and $Y$ is

$$IPVQA\_RoughDist'(X,Y) = \sqrt{\sum\nolimits_{i=1}^{W} DM(ind(X''_i), ind(Y''_i))^2 + l * (EM(1,i) - EM(2,i))^2} \tag{10}$$

$$EM = \begin{pmatrix} \overline{X}_1 & \overline{X}_2 & \overline{X}_3 & \cdots & \overline{X}_W \\ \overline{Y}_1 & \overline{Y}_2 & \overline{Y}_3 & \cdots & \overline{Y}_W \end{pmatrix}.$$

Then the distance measure for IPVQA can be rewritten aswhere $l = \frac{K}{W} > 1$, and $K$ is the length (dimensionality) of the original time series.

We proved that Eq. (8) is approximately equal to Eq. (10) when they are used to measure the reduced time series, i.e.,

$$IPVQA\_RoughDist(X, Y) \approx IPVQA\_RoughDist'(X, Y) \qquad (11)$$

The interested reader is encouraged to check the Appendix for more detailed information about the proof.

The lower bounding function avoiding the false dismissals for time series similarity search can also be obtained according to the method in paper [15]. People interested in the detailed form and proof of the lower bounding function can refer to the paper.

From Eqs. (8) and (10), we know that the distance functions have already considered the mean values of the subsequences. In Eq. (10), it is convenient and available to measure the distance between each pair of the original time series.

## 4. Experiments

The authors [15] had already performed on serval real and simulated datasets to demonstrate the efficiency of PVQA by comparing to the Euclidean distance and the piecewise approximation based on PCA and SAX, which indicates that PVQA has a good performance to mine time series and also has an excellent interpretability. In view of the previously much work of PVQA [15,17], in this work we only compare the improved approach to PVQA for testing and verifying whether the performance of IPVQA is better than PVQA.

Our experiments mainly include three parts. First of all, we show the results of the approximating time series for IPVQA and PVQA, which demonstrates that the performance of approximating time series for IPVQA is better than that for PVQA. We also apply the two approaches to the field of time series data mining, including clustering and classification. The experimental results indicate that the utility and effectiveness of IPVQA are better than PVQA.

It's needed to point out that the efficiency of the two approaches is very close because their time and space complexity is equal with the exception of the time and memory consumption of the normalizing process whose time complexity and space complexity are $O(K)$ and $O(W)$ respectively. Thereby, we do not empirically compare the time and memory consumption between the two approaches.

### 4.1. Comparison of the approximation

Since the two approaches IPVQA and PVQA are based on VQ and are used to represent the original time series, the smaller the approximation error of an approach, the better quality will be achieved to approximate time series.

Suppose there is a time series $X$ of length $K$. $Y_1 = \Phi_{IPVQA}(X,P)$ and $Y_2 = \Phi_{PVQA}(X,P)$ are respectively the approximation function of the two approaches IPVQA and PVQA, which means we can respectively obtain the reconstruction time series $Y_1$ and $Y_2$ of $X$ using the two approaches. In the two approximation functions the $P$ denotes a set of parameters, such as the number of codebook $S$ and the number of time sequences $W$. Thereby, we use the approximation error ($AE$) to test which approach is better.

$$AE(X, Y) = \sqrt{\sum_{i=1}^{K} (x_i - y_i)^2} \qquad (12)$$

where $x_i$ and $y_i$ are respectively the element of the original time series $X$ and the reconstruction time series $Y$. If $X$ denotes a time series dataset of size $M \times K$, then we have

$$AE(X, Y) = \frac{\sqrt{\sum_{i=1}^{M} \sum_{j=1}^{K} (x_{ij} - y_{ij})^2}}{M}. \qquad (13)$$

We perform experiments on the Control Chart dataset from the UCI data archive [28]. The number of the training time series and the test time series is equal to 300, i.e., $M = 300$. We first use the training time series to generalize a codebook according to $P = \{W,S\}$, then we make the approaches IPVQA and PVQA generate the reconstructed ones of the test time series. Meanwhile, there is a different parameter $P$ for each reconstructing operation. The experimental results of approximation error for IPVQA and PVQA are shown in Table 1. We can find that each approximation error of IPVQA shown in parentheses is smaller than that of PVQA, which means that the performance of the approximation of IPVQA is better than PVQA.

In Figs. 7 and 8, it is obvious that for the two approaches IPVQA and PVQA the larger $S$ and $W$ are, the smaller the approximation error is. It means that the proposed approach is available in the representation of time series as same as PVQA. In particular, Comparing PVQA to IPVQA, Fig. 9 shows the differences between the two approximation errors using the two methods for the various values of $S$ and $W$. It indicates that the quality of approximation for IPVQA is better because all of the differences are positive. It also illustrates that the smaller $S$ is, the better the quality of approximation for IPVQA will be.

**Table 1**
The approximation error of IPVQA (in parentheses) and PVQA for various parameter values $S$ and $W$.

| | S | | | | |
|---|---|---|---|---|---|
| W | 4 | 8 | 16 | 32 | 64 |
| 3 | 0.3300<br>(0.3003) | 0.3102<br>(0.2853) | 0.2855<br>(0.2696) | 0.2747<br>(0.2597) | 0.2655<br>(0.2515) |
| 6 | 0.3173<br>(0.2693) | 0.2817<br>(0.2498) | 0.2653<br>(0.2333) | 0.2494<br>(0.2169) | 0.2351<br>(0.2038) |
| 10 | 0.2957<br>(0.2279) | 0.2635<br>(0.2062) | 0.2365<br>(0.1840) | 0.2142<br>(0.1631) | 0.1945<br>(0.1441) |
| 12 | 0.2785<br>(0.2119) | 0.2505<br>(0.1843) | 0.2225<br>(0.1611) | 0.1981<br>(0.1398) | 0.1744<br>(0.1203) |

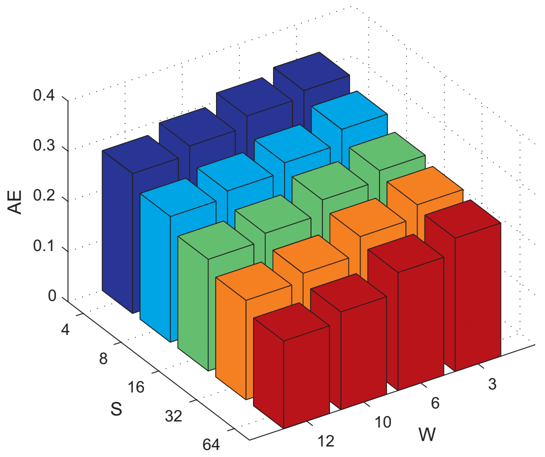


**Fig. 7.** The changeable trend of approximation error of PVQA for various values of *S* and *W*.

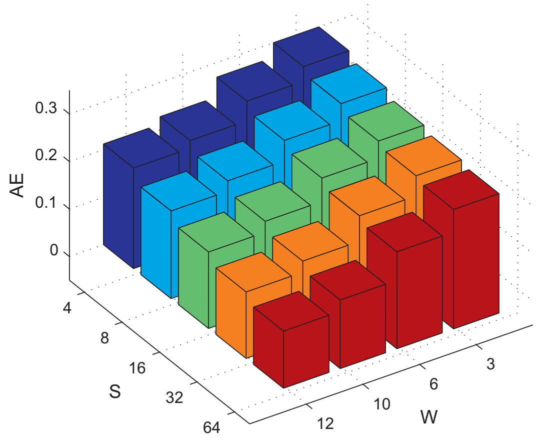


**Fig. 8.** The changeable trend of approximation error of IPVQA for various values of *S* and *W*.

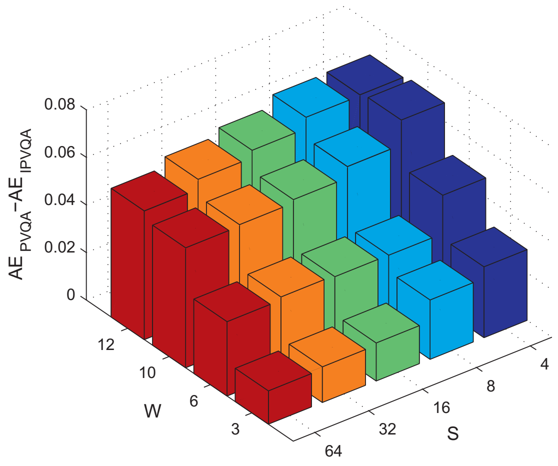


**Fig. 9.** The difference between the two approximation errors of PVQA and IPVQA for various values of *S* and *W*.

### *4.2. Clustering*

Clustering is one of the most important tasks in data mining. hierarchical clustering [7] is good at measuring the distance between different data objects and does well in providing the hierarchical results. Thereby, in this subsection we use the hierarchical clustering method to test whether the proposed approach IPVQA is better than PVQA.

We still choose the Control Chart dataset as the clustering objects. We arbitrarily pick out 16 time series from the test dataset used to hierarchical clustering. They are respectively in six groups {1,2,3,4}, {5}, {6,7}, {8,9,10,11}, {12,13}, {14,15,16}. The clustering results of the experiments for the various parameter *S* are shown in Figs. 10–12.

Comparing to the above groups of hierarchical clustering, we can find that the clustering performance of IPVQA is better than PVQA for each *S* under the same compress ratio. Moreover, the smaller *S* is, the more obvious the difference of the clustering results between the two method will be. As shown in Fig. 12, PVQA produces more false clusterings than IPVQA when *S* is small, i.e. $S = 4$. In PVQA, the false clusterings are (1,2,3,4), (8,9,10), (11,12,13) and (14,15,16). In IPVQA, there is only one false clustering, i.e. (1,2,3,4). It means that although the number of codewords in the codebook is small, these codewords generalized by IPVQA have the more powerful ability to represent the time series than that generalized by PVQA.

### *4.3. Classification*

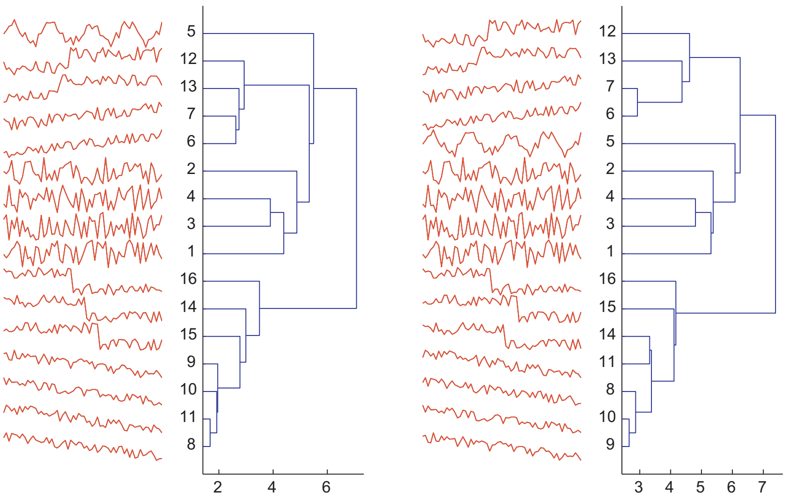
In order to further demonstrate the performance of data mining for our approach IPVQA. We first perform an experiment on the Control Chart dataset for various values of *S* (size of the codebook), which indicates that the larger the value of *S* is, the better the correctness of the classification in the dataset is. Other experiments on several time series datasets demonstrate the robustness of IPVQA for the various parameters *S* and *W* just mentioned in the above subsection.

In the first experiments of the classification, 300 time series in the training dataset are used to generalize the codebook according to different values of *S*. Let *S* be {4,8,16,32,64}. Meanwhile, we use the 1-nearest neighbor classification to classify the test dataset including another 300 time series. The classification results are denoted by the Classification Error Rate (*CER*), i.e.,

$$CER = \frac{\textit{\textbf{The number of wrong classified time series}}}{\textit{\textbf{The total number of time series in the test datasets}}}$$

Fig. 13 shows the result of the classification. It means that the correctness of classification for PVQA depends on the larger size of the codebook. However, for IPVQA even in the small size of the codebook its classification error rate is low. Moreover, The classification error rate of IPVQA for every value of *S* is lower than that of PVQA.

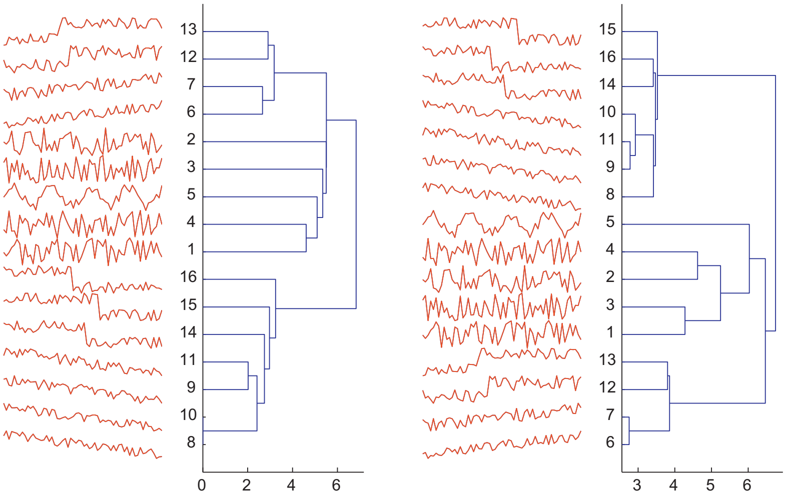
The other classification experiments are on different time series datasets. The means of the experiments are the same as the first experiment in this subsection. The de-



(a) PVQA

(b) IPVQA

**Fig. 10.** The result of hierarchical clustering for $W = 6$ and $S = 16$.



(a) PVQA

(b) IPVQA

**Fig. 11.** The result of hierarchical clustering for $W = 6$ and $S = 8$.

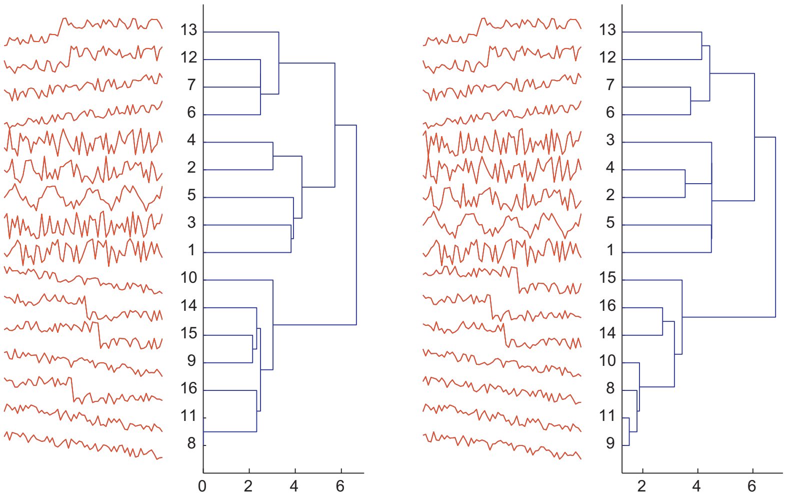
tailed information of the datasets used in those experiments is shown in Table 2, where $Nc$ denotes the number of class, $Str$ and $Ste$ respectively denote the size of the training set and test set, $K$ is the length of time series. Especially, Let the size of codebook be $S = \{4,8,16,32,64\}$.

For each size of the codebook, we use the two approaches to perform the experiments on different datasets according to the various values of $W$, which means each original time series is segmented into $W$ subsequences. Then we average the classification results of the experiments on the same dataset according to the corresponding $W$ for each $S$. Finally, we can obtain the average results as shown in Table 3.

In Table 3, the values in parentheses are the classification error rates of IPVQA, the others are the ones of PVQA. It is easy to find that most of the results of the classification of IPVQA are better than PVQA. Only three average results (in bold) of classification error rate of IPVQA are a little higher than PVQA, which means that our approach is also data dependent as same as PVQA. Meanwhile, In Fig. 14 we also can obtain those information for the six datasets of them because most of the error differences ($CER_{PVQA}$ - $- CER_{IPVQA}$) between the two approaches are positive. Moreover, for some dataset the performance of classification of IPVQA is better than PVQA such as the OliveOil dataset. Thereby, we can conclude that the performance of classification of IPVQA outperforms that of PVQA.

## 5. Conclusions

In this paper, we proposed an improved piecewise vector quantized approximation (IPVQA) based on the nor-



(a) PVQA

(b) IPVQA

**Fig. 12.** The result of hierarchical clustering for $W = 6$ and $S = 4$.

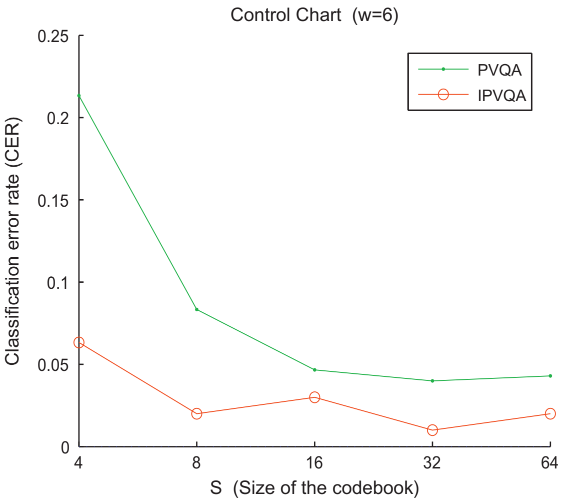


**Fig. 13.** The results of classification of IPVQA and PVQA for the various $S$.

**Table 2**
The datasets is used to experiments for various paraments $W$.

| ID | Name | Nc | Str | Ste | $k$ | $W$ |
|---|---|---|---|---|---|---|
| 1 | Adiac | 2 | 300 | 3000 | 426 | [4,8,11] |
| 2 | Beef | 5 | 30 | 30 | 470 | [10,47] |
| 3 | CBF | 3 | 30 | 900 | 128 | [4,8,16] |
| 4 | Coffee | 2 | 28 | 28 | 286 | [13,26] |
| 5 | ECG200 | 2 | 100 | 100 | 96 | [4,8,16] |
| 6 | Facefour | 4 | 24 | 88 | 350 | [10,35] |
| 7 | Gun_Point | 2 | 50 | 150 | 150 | [6,15,25] |
| 8 | Lighting2 | 2 | 60 | 61 | 637 | [7,13,49] |
| 9 | Lighting7 | 7 | 70 | 73 | 319 | [11,29] |
| 10 | OSULeaf | 6 | 200 | 242 | 427 | [7,61] |
| 11 | OliveOil | 4 | 30 | 30 | 570 | [10,19,57] |
| 12 | Swedishleaf | 15 | 500 | 625 | 128 | [4,16,32] |
| 13 | Trace | 4 | 100 | 100 | 275 | [5,11,25] |
| 14 | 2Pattern | 4 | 1000 | 4000 | 128 | [4,16,32] |
| 15 | Control | 6 | 300 | 300 | 60 | [6,10,15] |
| 16 | Wafer | 2 | 1000 | 6174 | 152 | [8,19] |
| 17 | 50word | 50 | 450 | 455 | 270 | [6,18,27] |
| 18 | Yoga | 2 | 300 | 3000 | 426 | [6,71] |

malized time subsequences. This proposed approach mainly uses the vector quantization to generate a codebook. To improve the ability of the codebook generated

**Table 3**
The average results of the classification error rate of PVQA and IPVQA (in parentheses) on different datasets for various values of $S$.

| | $S$ | | | | |
|---|---|---|---|---|---|
| ID | 4 | 8 | 16 | 32 | 64 |
| 1 | 0.9233<br>(0.6633) | 0.8772<br>(0.6232) | 0.7962<br>(0.5601) | 0.6939<br>(0.5132) | 0.6317<br>(0.4783) |
| 2 | 0.5833<br>(0.5167) | 0.5500<br>(0.4667) | 0.5167<br>(0.5000) | 0.5500<br>(0.4667) | 0.5000<br>(0.4667) |
| 3 | 0.1970<br>(0.0630) | 0.1085<br>(0.0785) | 0.1007<br>(0.0667) | 0.1107<br>(0.0781) | 0.1026<br>(**0.1170**) |
| 4 | 0.4286<br>(0.3036) | 0.3036<br>(0.2500) | 0.3750<br>(0.2679) | 0.3393<br>(0.2500) | 0.3214<br>(0.2321) |
| 5 | 0.2300<br>(0.1433) | 0.1633<br>(0.1533) | 0.1300<br>(**0.1333**) | 0.1233<br>(**0.1300**) | 0.1333<br>(0.1267) |
| 6 | 0.2898<br>(0.2102) | 0.2045<br>(0.1875) | 0.1648<br>(**0.1989**) | 0.2557<br>(0.2273) | 0.2500<br>(0.2216) |
| 7 | 0.2978<br>(0.1111) | 0.2467<br>(0.1133) | 0.2133<br>(0.1111) | 0.1356<br>(0.0844) | 0.1267<br>(0.0844) |
| 8 | 0.2787<br>(0.1639) | 0.2459<br>(**0.2623**) | 0.2404<br>(**0.2568**) | 0.2842<br>(0.2732) | 0.2678<br>(0.2623) |
| 9 | 0.5068<br>(0.3836) | 0.4247<br>(0.3425) | 0.4110<br>(0.3767) | 0.4452<br>(0.3493) | 0.4452<br>(0.4315) |
| 10 | 0.5475<br>(0.4938) | 0.5145<br>(0.4979) | 0.4979<br>(0.4835) | 0.4979<br>(0.4938) | 0.4855<br>(0.4711) |
| 11 | 0.8222<br>(0.1556) | 0.8111<br>(0.1667) | 0.5667<br>(0.1667) | 0.2556<br>(0.1556) | 0.2222<br>(0.1222) |
| 12 | 0.5552<br>(0.2971) | 0.4416<br>(0.2763) | 0.3456<br>(2475) | 0.2928<br>(0.2389) | 0.2747<br>(0.2219) |
| 13 | 0.4933<br>(0.3100) | 0.3000<br>(**0.3033**) | 0.3367<br>(0.2733) | 0.3233<br>(0.2633) | 0.2567<br>(0.2433) |
| 14 | 0.2598<br>(0.1158) | 0.2238<br>(0.1188) | 0.1113<br>(1039) | 0.0903<br>(0.0846) | 0.0952<br>(0.0872) |
| 15 | 0.1433<br>(0.0311) | 0.0778<br>(0.0233) | 0.0600<br>(0.0289) | 0.0789<br>(0.0256) | 0.0744<br>(0.0478) |
| 16 | 0.0066<br>(0.0058) | 0.0070<br>(0.0041) | 0.0049<br>(0.0042) | 0.0043<br>(0.0041) | 0.0040<br>(**0.0054**) |
| 17 | 0.4176<br>(0.3729) | 0.4198<br>(0.3773) | 0.3971<br>(0.3648) | 0.3751<br>(0.3619) | 0.3685<br>(0.3553) |
| 18 | 0.3325<br>(0.1867) | 0.3015<br>(0.1930) | 0.2937<br>(0.1858) | 0.2750<br>(0.1838) | 0.2648<br>(1833) |

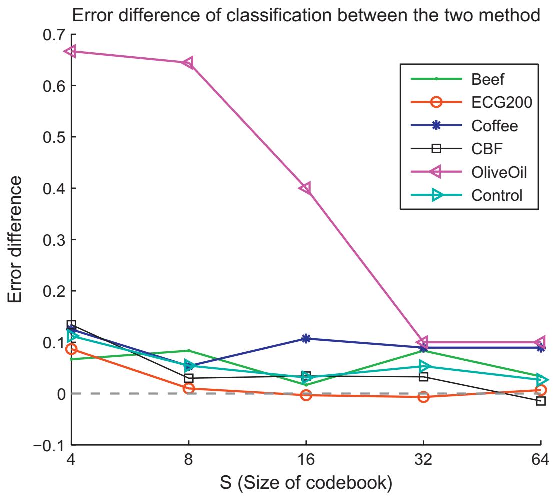


**Fig. 14.** The difference between the classification error rate of PVQA and IPVQA for various values of $S$ and $W$.

by the traditional PVQA without processing the vertical offsets of time subsequences, IPVQA considers the vertical offset and remove the mean of each subsequence used to generate the best codebook for the representation of time series. Meanwhile, we also provide a new function reconsidering the mean of time subsequence to better measure the distance between two time series. This distance function retains the good properties of the one applied in the original PVQA and can fast compute the distance of time series. The experiments on different time series datasets also demonstrate that the proposed IPVQA is better than PVQA in the field of time series data mining, including representation, clustering and classification.

Since a codebook can be obtained by IPVQA, we can use the methods of symbolic representation to mine time series in the future. Moreover, the proposed approach may also be extended to represent subsequences at multiple resolution as the paper [17] have done. Additionally, because the codewords generated by IPVQA keep the most information about the original time series, we may use it to visualize the time series and discover more accurate patterns in the field of time series data mining.

## Acknowledgments

This work has been partly supported by the Fundamental Research Funds for the Central Universities (12SKGC-QG03), the Research Funds of Huaqiao University (13SKBS104) and the National Natural Science Foundation of China (70871015 and 71031002).

## Appendix A

**Proposition 1.** *Given two time series $X = \{X_1, X_2, \ldots, X_W\}$ and $Y = \{Y_1, Y_2, \ldots, Y_W\}$, where $X_i$ (or $Y_i$) of length $l$ is the ith subsequence of $X$ (or $Y$) of length $K = l{*}W$ and $l > 1$. Record the mean value of each subsequence, $\overline{X} = \{\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_W\}$, $\overline{Y} = \{\overline{Y}_1, \overline{Y}_2, \ldots, \overline{Y}_W\}$ and $EM = [X, Y]^T$, where T denotes matrix transposition. Obtain the codeword series $X'' = \{X''_1, X''_2, \ldots, X''_W\}$ and $Y'' = \{Y''_1, Y''_2, \ldots, Y''_W\}$, where $X''_i = \{x''_{i1}, x''_{i2}, \ldots, x''_{il}\}$ and $Y''_i = \{y''_{i1}, y''_{i2}, \ldots, y''_{il}\}$. The reconstructed two time sequences are $\widehat{X_i} = \{\hat{x}_{i1}, \hat{x}_{i2}, \cdots, \hat{x}_{il}\}$ and $\widehat{Y_i} = \{\hat{y}_{i1}, \hat{y}_{i2}, \cdots, \hat{y}_{il}\}$, i.e., $\widehat{X}_i = X''_i + \overline{X}_i$ and $\widehat{Y}_i = Y''_i + \overline{Y}_i$. There are IPVQA_RoughDist(X,Y) and IPVQA_RoughDist′(X,Y). The following approximate equation between the two distance function holds:*

$$IPVQA\_RoughDist(X, Y) \approx IPVQA\_RoughDist'(X, Y).$$

## Proof

$$
\begin{aligned}
&(D(\widehat{X}_i, \widehat{Y}_i))^2\\
&= \sum_{j=1}^{l}\left(x''_{ij} + \overline{X}_i - \left(y''_{ij} + \overline{Y}_i\right)\right)^2\\
&= \sum_{j=1}^{l}\left(\left(x''_{ij} - y''_{ij}\right) + (\overline{X}_i - \overline{Y}_i)\right)^2\\
&= \sum_{j=1}^{l}\left(\left(x''_{ij} - y''_{ij}\right)^2 + 2 * \left(x''_{ij} - y''_{ij}\right)(\overline{X}_i - \overline{Y}_i) + (\overline{X}_i - \overline{Y}_i)^2\right)
\end{aligned}
$$

Notice that,

$$
\begin{aligned}
&\sum_{j=1}^{l}\left(x''_{ij} - y''_{ij}\right)(\overline{X}_i - \overline{Y}_i)\\
&= (EM(1,i) - EM(2,i))\sum_{j=1}^{l}\left(x''_{ij} - y''_{ij}\right)\\
&= (EM(1,i) - EM(2,i))\sum_{j=1}^{l}((\widehat{X_i} - EM(1,i)) - (\widehat{Y_i} - EM(2,i)))\\
&= (EM(1,i) - EM(2,i))\sum_{j=1}^{l}((\widehat{x}_{ij} - \hat{y}_{ij}) - (EM(1,i) - EM(2,i)))\\
&= (EM(1,i) - EM(2,i))\sum_{j=1}^{l}(\hat{x}_{ij} - \hat{y}_{ij}) - l * (EM(1,i) - EM(2,i))^2\\
&= l * (EM(1,i) - EM(2,i))\frac{1}{l}\sum_{j=1}^{l}(\hat{x}_{ij} - \hat{y}_{ij}) - l * (EM(1,i) - EM(2,i))^2\\
&\approx l * (EM(1,i) - EM(2,i))^2 - l * (EM(1,i) - EM(2,i))^2 \quad (\bigstar)\\
&= 0
\end{aligned}
$$

Since $\widehat{X}_i$ and $\widehat{Y}_i$ can approximate the corresponding original time subsequences $X_i$ and $Y_i$ well whose the mean values are $EM(1,i)$ and $EM(2,i)$ respectively, there is $\frac{1}{l}\sum_{j=1}^{l}\hat{x}_{ij} \approx EM(1,i)$ and $\frac{1}{l}\sum_{j=1}^{l}\hat{y}_{ij} \approx EM(2,i)$ existing in the row marked (★).

Thereby, we have

$$
\begin{aligned}
&\sum_{i=1}^{W}(D(\widehat{X}_i, \widehat{Y}_i))^2\\
&\approx \sum_{i=1}^{W}\sum_{j=1}^{l}\left(\left(x''_{ij} - y''_{ij}\right)^2 + (\overline{X}_i - \overline{Y}_i)^2\right)\\
&= \sum_{i=1}^{W}\sum_{j=1}^{l}\left(\left(x''_{ij} - y''_{ij}\right)^2\right) + l * (\overline{X}_i - \overline{Y}_i)^2\\
&= \sum_{i=1}^{W} DM(ind(X''_i), ind(Y''_i))^2 + l * (EM(1,i) - EM(2,i))^2\\
&= IPVQA\_RoughDist'(X, Y). \quad \square
\end{aligned}
$$

So far, we have already proven that Eq. (8) is approximately equal to Eq. (10).

## References

[1] K.P. Chan, A. Fu, Efficient time series matching by wavelets, in: Proceedings of the 15th IEEE International Conference on Data Engineering. Sydney, Australia, 1999, pp. 126–133.

[2] K.P. Chan, A. Fu, C. Yu, Haar wavelets for efficient similarity search of time series: with and without time warping, IEEE Transactions on Knowledge and Data Engineering (2003) 686–705.

[3] M.A. Iyer, M.M. Harris, L.T. Watson, M.W. Berry, A performance comparison of piecewise linear estimation methods, in: Proceedings of the 2008 Spring Simulation Multi-Conference, 2008, pp. 273–278.

[4] H. Li, C. Guo, W. Qiu, Similarity measure based on piecewise linear approximation and derivative dynamic time warping for time series mining, Expert Systems with Applications 38 (12) (2011) 14732–14743.

[5] J. Lin, E. Keogh, L. Wei, S. Lonardi, Experiencing SAX: a novel symbolic representation of time series, Data Mining and Knowledge Discovery 15 (2007) 07–144.

[6] H. Li, C. Guo, Piecewise cloud approximation for time series mining, Knowledge-Based Systems 24 (4) (2011) 492–500.

[7] J. Lin, E. Keogh, S. Lonardi, B. Chiu, A symbolic representation of time series with implications for streaming algorithms, in: Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 2003, pp. 2–11.

[8] E. Keogh, J. Lin, A. Fu, Hot SAX: efficiently finding the most unusual time series subsequence, in: Proceedings of the 5th IEEE International Conference on Data Mining, 2005, pp. 226–233.

[9] C. Guo, H. Li, D. Pan, An improved piecewise aggregate approximation based on statistical features for time series mining, in: Proceedings of the 4th International Conference on Knowledge Science, Engineering and Management, 2010, pp. 234–244.

[10] N.Q.V. Hung, D.T. Anh, An improvement of PAA for dimensionality reduction in large time series Databases, in: Proceedings of the 10th Pacific Rim International Conference on Artificial Intelligence, 2008, pp. 698–707.

[11] A. Camerra, T. Palpanas, J. Shieh, E. Keogh, iSAX 2.0: indexing and mining one billion time series, in: Proceedings of 2010 IEEE International Conference on Data Mining, 2010, pp. 58–67.

[12] S.Theodoridis, K. Koutroumbas, Feature generation I: data transformation and dimensionality reduction, Pattern Recognition, fourth ed., 2009, pp. 323–409.

[13] C. Faloutsos, M. Ranganathan, Y. Manolopoulos, Fast subsequence matching in time series databases, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, 1994, pp. 419–429.

[14] V. Megalooikonomou, G. Li, Q. Wang, A dimensionality reduction technique for efficient similarity analysis of time series databases, in: Proceedings of the 13th ACM Conference on Information and Knowledge Management, Washington, DC, USA, 2004, pp. 160–161.

[15] Q. Wang, V. Megalooikonomou, A dimensionality reduction technique for efficient time series similarity analysis, Information Systems 33 (1) (2008) 115–132.

[16] V. Megalooikonomou, Q. Wang, G. Li, C. Faloutsos, A multiresolution symbolic representation of time series, in: Proceedings of the 21st IEEE International Conference on Data Engineering, Tokyo, Japan, 2005, pp. 688–678.

[17] Q. Wang, V. Megalooikonomou, C. Faloutsos, Time series analysis with multiple resolutions, Information Systems 35 (2010) 56–74.

[18] Y. Linde, A. Buzo, R.M. Gray, An algorithm for vector quantizer design, IEEE Transactions on Communications (1980) 702–710.

[19] A. Gersho, R.M. Gray, Vector Quantization and Signal Compression, Kluwer Academic, Boston, 1992.

[20] Z.M. Lu, J.X. Wang, B.B. Liu, An improved lossless data hiding scheme based on image VQ-index residual value coding, Journal of Systems and Software 82 (6) (2009) 1016–1024.

[21] C.H. Yang, S.C. Wu, S.C. Huang, Y.K. Lin, Huffman-code strategies to improve MFCVQ-based reversible data hiding for VQ indexes, Journal of Systems and Software. 84 (3) (2011) 388–396.

[22] VQ(vector quantization) <http://www.data-compression.com/vq.html>, 2002.

[23] S. Lee, D. Kwon, S. Lee, Minimum distance queries for time series data, Journal of Systems and Software 68 (1) (2004) 105–113.

[24] S.W. Kim, J. Yoon, S. Park, J.I. Won, Shape-based retrieval in time-series databases, Journal of Systems and Software 79 (2) (2006) 191–203.

[25] C. Faloutsos, W. Equitz, M. Flickner, W. Niblack, D. Petkovic, R. Barber, Efficient and effective querying by image content, Journal of Intelligent Information Systems 3 (1994) 231–262.

[26] R. Agrawal, K.I. Lin, H.S. Sawhney, K. Shim, Fast similarity search in the presence of noise, scaling, and translation in time-series databases, in: Proceedings of the 21st International Conference on Very Large Databases, Zurich, Switzerland, 1995.

[27] E. Keogh, K. Chakrabarti, M. Pazzani, S. Mehrotra, Dimensionality reduction for fast similarity search in large time series databases, Journal of Knowledge and Information Systems 3 (3) (2000) 263–286.

[28] E. Keogh, Welcome to the UCR time series classification/clustering page <http://www.cs.ucr.edu/eamonn/time_series_data/>, 2003.