Hierarchical Piecewise Linear Approximation

A Novel Representation of Time Series Data

Vineetha Bettaiah, Heggere S Ranganath
Computer Science Department
The University of Alabama in Huntsville
Huntsville, USA
vineetha.bettaiah@gmail.com

Abstract— **This paper presents a Hierarchical Piecewise Linear Approximation (HPLA) for the representation of time series data in which the time series is treated as a curve in the time-amplitude image space. The curve is partitioned into segments by choosing perceptually important points as break points. Each segment between adjacent break points is recursively partitioned into two segments at the best point or midpoint until the error between the approximating line and the original curve becomes less than a pre-specified threshold. The HPLA achieves dimensionality reduction while preserving prominent local features and general shape of the time series. The HPLA permits coarse-fine processing, allows flexible definition of similarity between two time series based on mathematical measures or general time series shape, and supports query by content, clustering and classification based on whole or subsequence similarity.**

Keywords-Data Mining; Dimensionality Reduction; Piecewise Linear Representation; Time Series Representation.

I. INTRODUCTION

Many areas of science, engineering, and business are generating, archiving and processing vast amounts of data. Because of the sheer volume, the data science community is challenged to develop new methodologies for the modeling, representation, retrieval, processing, understanding, and visualization of "Big Data". Big data is a collection of larger and complex data sets, difficult to manage and process using traditional data management and processing techniques. One type of data that has received a lot of attention in recent years in diverse areas including medicine, astronomy, geology, atmospheric and space science, engineering, and financial markets is time series data. Mathematically, a time series $T = \{x_1, x_2, \ldots, x_n\}$ is a sequence of n real numbers in the increasing order of time, where each value has a time stamp. The time spacing between adjacent samples x_i and x_{i+1} may remain constant or vary over the duration of the time series.

The main processing tasks or operations associated with time series data are query by content, clustering, classification, prediction, anomaly detection, motif discovery, and rule discovery [1]. These are well known problems in pattern recognition and data mining areas for many years. However, the proven pattern recognition and data mining methods are not suitable for processing time series data, mainly because of three reasons. First, the dimensionality of time series is very high, could be as high as tens of thousands. Secondly, the corresponding elements of two time series may not align due to difference in length, scale, translation, shift or non-uniform spacing between adjacent elements. Finally, the notion of similarity in the context of time series is very different from the one used in pattern recognition. Unlike in pattern recognition, where all elements of pattern vectors are used to determine similarity between two patterns, only subsets of elements of the two time series may be used to determine their similarity. Two time series may be considered similar if they contain similar subsequences of sufficient length or several similar patterns in the same time order.

An obvious solution to the above problem is to use compact representations of time series that are capable of achieving a significant reduction in dimensionality without losing important features present in the original data. During the past two decades several piecewise linear, symbolic, transform and model based representations have been developed [2]-[5]. Each representation has its own advantages and disadvantages. For example, transform based representations being global representations do not provide local information about subsequences [6]. The symbolic representations, such as SAX loose most of the shape information due to two levels of approximation [4]. The Singular Value Decomposition (SVD) [3] and Hidden Markov Model (HMM) [5] representations are computationally very expensive. To the best of our knowledge, as of now, characteristics of an ideal (good) time series representation based on the needs of time series data mining applications are not explicitly identified.

This paper makes three contributions. First, in Section II, requirements an ideal time series representation should satisfy are identified. Secondly, in Section III, widely used piecewise linear representations are analyzed to determine their strengths and weaknesses by using the requirements identified in Section II as metrics. Thirdly, in Section IV, a new representation called Hierarchical Piecewise Linear Approximation (HPLA), which is closer to the ideal representation than existing representations, is described. The advantages of the HPLA representation are described in Section V. By using the compression ratio and representation accuracy as metrics, a comparison of the HPLA with Piecewise Aggregation Approximation (PAA) and Piecewise Linear Approximation (PLA) is given in Section VI.

 ISBN: 978-1-61208-334-6

Conclusions and recommendation for future research are given in Section VII.

II. CHARACTERISTICS OF AN IDEAL TIME SERIES DATA REPRESENTATION

In this section, characteristics for an ideal time series data representation are identified based on the needs of the important time series data mining applications.

In *query by content*, the objective is to retrieve time series from the database that are similar in information content to the given query time series Q [1]. The similarity between Q and time series T in the database may be determined by matching Q and T, or sub-sequences of Q and T. Though the content is almost always specified by a query sequence, it is desirable to have flexibility on how the content is specified. For example, general shape of time series, sequence of events, and similar subsequences are valid specifications of content in many applications. *Therefore, to support query by content, the representation should support broad specification of content, and should have a distance measure satisfying lower bound criterion.*

In *clustering*, given a set of N unlabeled time series T_1, T_2, T_3, …., T_N, the goal is to partition the set into K groups based on a meaningful similarity measure, such that members belonging to a group are similar to one another, and members belonging to different groups differ significantly from one another [1]. The feature-based methods compute a small number of features to represent each time series, and then use clustering algorithms, such as k-means algorithms to cluster feature vectors. The model-based methods extract a set of parameters for each time series, and then find clusters by clustering parameters. The raw-data-based methods are rarely used due to high dimensionality. Today, clustering algorithms are set in a mathematical framework, which use feature vectors or model parameters. A syntactic clustering approach using broad similarity as perceived by humans is needed for clustering time series data. *Therefore, the representation should preserve salient attributes of the time series to support the development of mathematical and syntactic clustering algorithms.*

Classification is the process of assigning an input time series to one of the several known classes or categories [1]. Bayesian classifiers are not practical for use with raw time series as the computation of probability density functions for such high dimensionality time series is not feasible. The linear classifiers (perceptron, least mean square methods, support vector machines, etc.), and non-linear artificial neural networks require large number of samples, at least two times the dimensionality of time series. Even if the required training samples are available, training classifiers with large number of high dimensional vectors is not practical. Thus, classification based on features appears to be the only practical solution. *Therefore, the representation should preserve salient attributes of the time series, and allow the computation of geometric and mathematical features needed for training classifiers.*

Given a time series $T = \{x_1, x_2, , , , , x_n\}$, *prediction* is the task of determining likely values of x_i for $i > n$. The future values are predicted based on the current evolution trend observed or mathematical models such as Hidden Markov Model developed from historic data similar to the current time series [1]. Usually, the model is based on prominent features of time series. *Therefore, the representation should preserve local features and evolution trends of time series as accurately as possible.*

Motif detection is the process of identifying an approximately repeating subsequence representing meaningful pattern in a time series or a group of time series [1]. Motifs have been widely used for rule-discovery, clustering and classification of time series data. *Therefore, the time series representation must facilitate the identification of motifs of varying lengths by preserving perceptually important points and local trends.*

Rule discovery learns temporal rules that are hidden or not obvious in time series data [1]. One approach is to transform time series to a sequence of symbols and use association rule mining algorithms to discover rules. Another approach is to transform time series to a sequence of events, and use classification trees to discover temporal rules. *Therefore, it should be possible to obtain from the representation a meaningful sequence of symbols and events as defined by the user.*

In addition to the application specific requirements identified above, a few general requirements are also listed below.

1) The representation should be as compact as possible to achieve maximum dimensionality reduction, and at the same time should allow the reconstruction of the original time series with little error.

2) The representation should allow matching two time series using full sequences or subsequences even if the two time series differ in length, scale, amplitude, and translation.

3) The representation should retain salient attributes and local evolution trends.

4) The representation should allow the computation of geometric and mathematical features, and model parameters to support feature and model based processing.

5) To support query by content, the representation should support broad and flexible specification of content.

6) The computation for building the representation itself should be reasonable, should not require prior knowledge of the type of motifs or general shape of the time series.

7) The representation should have a distance measure satisfying lower bound criterion.

III. RELATED WORK

The time series representations can be broadly classified into four categories, namely, piecewise linear representations, transform based representations, symbolic representations, and model based representations. As the HPLA representation proposed in this paper is a piecewise linear approximation, only the piecewise linear representations are briefly described and analyzed.

A. Piecewise Aggregation Approximation (PAA)

Let $X = \{x_1,x_2,…,x_n\}$ be a time series of length n. The PAA representation of X is obtained by partitioning X into N

segments of equal length n/N, where $N << n$, and then representing each segment by the mean of elements belonging to the segment [2].

B. Piecewise Linear Approximation (PLA)

The PLA is the most frequently used representation in which a time series X of length n is partitioned into $N << n$ sections and each section is represented by a straight line. Keogh and Pazzani refer to the process of generating PLA representation as segmentation of the time series [7]. Uniform segmentation produces segments of equal length $l = n/N$. Non-uniform segmentation partitions the time series into segments of unequal length to best fit the shape of the time series. Linear interpolation approximates the segment $X[a:b]$ by the line joining x_a and x_b. Linear regression fits the best possible line to $X[a:b]$ in the least square sense. As linear interpolation requires constant time, it is the most widely used method.

Several variations of PLA representations have been developed in recent years. Yan et al. and Pratt et al. segment the time series at important maxima and minima, and represent the time series by a polyline joining adjacent local maximum and minimum [8][9]. Park et al. partition the time series into monotonically increasing or decreasing segments, and characterize each segment by a six dimensional feature vector [10]. Zhou et al. suggest building a PLA representation (Slope Threshold Change) by using points at which slope changes significantly as break points [11]. In Piecewise Linear Aggregate Approximation (PLAA), Nguyen Quoc et al. divide the time series into N segments of equal length, and represent each segment by the mean and slope of the best fitting straight line [3].

C. Adaptive Piecewise Constant Approximation (APCA)

The APCA representation of a time series is obtained by segmenting the time series into N segments of unequal length based on data. Long segments are used to represent data regions of low activity, and short segments are used to represent regions of high activity. Each segment is represented by its mean value and the index of the right end point. Therefore, the time series $X = \{x_1,x_2,\ldots,x_n\}$ is represented as $\{\langle xv_1, xr_1\rangle, \ldots, \langle xv_N, xr_N\rangle\}$, where xv_i is the mean of all values in the i^{th} segment, and xr_i is the index of the right most element of the i^{th} segment.

An evaluation of PLR representations based on requirements identified in Section II is given below.

1) The compression ratio, reconstruction accuracy and shape complexity of time series are related. The only way of achieving high reconstruction accuracy is by partitioning the time series into a large number of segments, which limits the extent to which the dimensionality is reduced. The APCA and the PLA representations achieve higher compression than PAA as they limit the number of segments by placing long segments in regions, where values are fairly constant or linear, respectively. For a given compression ratio PLA achieves higher reconstruction accuracy than PAA and APCA [14].

2) The orders of computation for PAA, PLA and APCA representations are $O(n)$, $O(nL)$, and $O(nlog_2n)$, respectively [2][7][14].

3) The PAA and APCA representations do not provide any information regarding the shape of the time series within segments. Therefore, there may not be sufficient information to detect shapes and trends spanning one or more segments, and the computation of many features needed for feature based data mining applications may not be possible. The PLA is better than PAA and APCA in approximating the shape of the time series, especially if perceptually important points are used as break points during segmentation.

4) The PAA and APCA representations do not use perceptually important points like local maxima and minima as break points. As a result, matching two time series which differ in length, scale, or translation is not easy or even possible. These representations are not suitable for establishing similarity between two time series based on their subsequences. It may be possible to deal with differences in length, scale, amplitude, translation, and subsequence matching using PLA representation if time series are segmented at perceptually important points.

5) For query by content application, the PAA and APCA have distance measures that satisfy minimum bounding criterion. However, specification of content based on shape and subsequences is not possible. In general, PLA does not have distance measure that satisfies minimum bounding criterion. Broad specification for content is possible only if time series are segmented appropriately.

The findings are summarized in Table I. "Yes", "No", and "May be" are used to indicate that the requirement is well satisfied, not satisfied, or partially satisfied, respectively. The PLA representation, if break points include perceptually important maxima and minima, is expected to achieve higher reconstruction accuracy than other representations. This is supported by experimental results given in Section VI, and simulation study reported by other researchers [2]. Because of high reconstruction accuracy, the PLA retains local patterns and evolution trends better than other representations. In summary, a properly obtained PLA representation along with segment features has the potential to satisfy 6 out of 7 requirements identified in Section II.

TABLE I. EVALUATION OF REPRESENTATION BASED ON REQUIREMENTS OF AN IDEAL REPRESENTATION

Requirement	PAA	PLA	APCA
1	No	Yes	No
2	No	Yes	No
3	No	Yes	No
4	No	Yes	No
5	No	Yes	No
6	O(n)	O(n)	$O(nlog_2n)$
7	Yes	Yes	Yes

IV. THE HIERARCHICAL PIECEWISE LINEAR REPRESENTATION

The HPLA is a multi-level representation of time series data which facilitates the development of effective and efficient algorithms for coarse-fine processing and mining of time series data. The representation is developed to permit the determination of similarity between two time series when they share similar subsequences, patterns, or time ordered sequence of patterns. It is effective in handling differences in length, translation, time and amplitude scales, minor warp, and even some missing data. It is also possible to determine similarity between two time series using mathematical distance measures (quantitative) or general shape (subjective). The approach, by treating time series as a curve in the time-amplitude binary image, takes advantage of the well-established chaincode based curve smoothing and segmentation methods in the area of image processing [12], [13]. A step-by-step description of obtaining the HPLA representation of a time series is given below.

Step 1: *Normalize the time series.*

The time series $X = \{x_1,x_2,\ldots,x_n\}$ is normalized by replacing amplitude x_i by $(x_i - m)/\sigma$, , for $1 \leq i \leq n$, where m and σ are the mean and standard deviation of all amplitude values of the time series.

Step 2: *Digitize the normalized time series and obtain the chaincode representation of the resulting curve.*

In digital image processing, a curve is often represented compactly by its chaincode [12]. The chaincode of a curve is simply a sequence of directional codes, where the i^{th} element of the chaincode specifies the direction of the i^{th} pixel (point) relative to the $(i\text{-}1)^{th}$ pixel along the curve. A 3-bit binary code is used to encode the 8 possible directions.

The time series X may be considered as an open curve in the time-amplitude image space. As time increases monotonically, if X is represented as a digital curve by digitizing its values, from any pixel on the curve the next pixel can be reached by moving one unit in one of the five possible directions shown in Fig. 2. The algorithm for obtaining the chaincode of the time series X without actually transforming X to a digital image is given below.

```
Generate_Chaincode (X, n, b)
   // X: input time series of length n
   // b: amplitude resolution for quantizing elements of X
   Chaincode ← empty list
   p = int(x1/b + 0.5);  i = 2;
  while (i ≤ n)
      q = int(xi/b + 0.5);
      if (q = p)
         Append 2 to Chaincode;
      else if (q > p)
         Append 3 followed by (q-p-1) 4s to Chaincode
      else
         Append 1 followed by (q-p-1) 0s to Chaincode
      p = q;  i++;
return Chaincode
```

The above algorithm digitizes the time series, and the generates the chaincode in one pass in linear time ($O(n)$). The digital curve shown in Fig. 1 is obtained by digitizing the normalized amplitude values of a time series of length 77 with a bin size of 0.1. The chaincode of the curve is {2433223343343343342232220110111113212100101001101343434443444344434343343343343323113310001001000100100010112210112 12}. Note, the length of the chaincode is greater than the length of the time series due to filling. As the chaincode is computed using sliding window approach and discarded after the computation of feature vector, space is not a major issue.

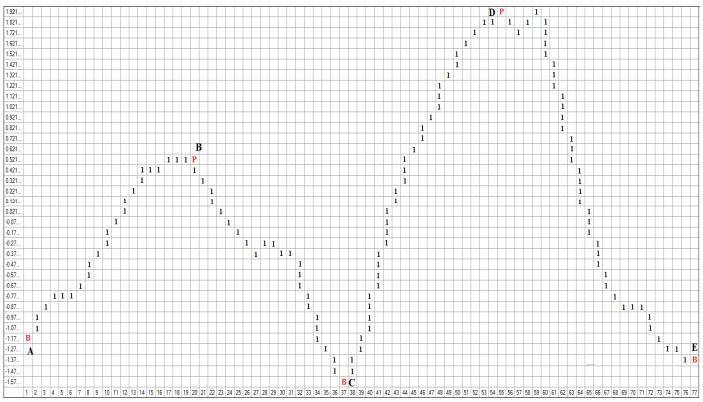

Figure 1. Digitized time series with break points.

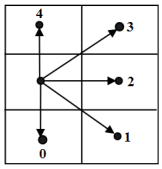

Figure 2. Directional codes.

Step 3: *Determine perceptually important maxima and minima of the curve.*

Nabors has defined four types of curves - type 1, type 2, type 3, and type 4 [12]. The slope along a type 1 curve is between negative infinity and -1, and is represented by a sequence of direction codes 0 and 1. The slope along a type 2 curve is between -1 and 0, and is represented by a sequence of direction codes 1 and 2. The slope along a type 3 curve is between 0 and 1, and is represented by a sequence of direction codes 2 and 3. Finally, the slope along a type 4 curve is between 1 and infinity, and is represented by a sequence of direction codes 3 and 4.

The chaincode of the curve is partitioned into non-overlapping subsequences, where each subsequence represents one of the four curve types. Points at which type 1 or type 2 curves meet type 3 or type 4 curves are local maxima or minima. For a local minimum (maximum), a type 1 or type 2 (type 3 or type 4) curve is followed by a type 3 or type 4 (type 1 or type 2) curve. Instead of selecting all, only the prominent local maxima and minima are selected as break points. A local maximum is taken as a prominent maximum if its raise from the immediately preceding

 ISBN: 978-1-61208-334-6

minimum is greater than the average of raises of all maxima. The algorithm is given below.

```
DeterminProminent_MaxMin(ChainCode)
(MaxMin, MaxMinIndex) = FindMax&Min(ChainCode);
(avgRaise, avgFall) =
FindAverageRaise&Fall(MaxMin);

for i=0 to length(MaxMin)-1
   if MaxMin(i) > avgRaise OR MaxMin(i) < avgFall
        Prominent_MaxMin_I.add(MaxMin(i));

Prominent_MaxMin_Index_I.add(MaxMinIndex(i));
i=0 ;
while i < length(Prominent_MaxMin _I - 1)
   if Prominent_MaxMin_I(i) > 0
       Add index of the global maximum between
       Prominent_MaxMin_Index_I(i) and
       Prominent_MaxMin_Index_I(i+1)
      (both inclusive) to Prominent_MaxMin_Index_F;

   if Prominent_MaxMin_I(i)  < 0
       Add index of the global minimum between
       Prominent_MaxMin_Index_I(i) and
       Prominent_MaxMin_Index_I(i+1)
       (both inclusive) to Prominent_MaxMin_Index_F;
```

The function FindMaxMin finds all local maxima and minima, and stores their values and indices in MaxMin and MaxMinIndex, respectively. The average raise and fall in value between adjacent maximum and minimum are computed by FindAverageRaise&Fall. Each local maximum with a raise from its immediately preceding minimum greater than average raise becomes an initial prominent maximum. The initial prominent minima are selected, similarly. The initial list of prominent maxima and minima is refined such that maxima and minima appear alternately in the final list. The algorithm identifies two local maxima and three minima (including two end points) as break points for the curve in Fig. 1. These break points are labeled A, B, C, D, and E.

Step 4: *Smooth the curve segments between adjacent break points.*

The curve segment connecting adjacent break points is smoothed by directly modifying the chaincode. A smoothing algorithm similar to the algorithm given by Kim is used for this purpose [13]. Unlike Kim's algorithm which first requires the identification of distorted sections of the curve, the new algorithm operates on the entire chaincode and selectively modifies elements likely responsible for distortion. It has been shown that the chaincode based algorithm keeps most of the points in their original positions as it smoothes the curve. The smoothing suppresses minor fluctuation due to noise, and is usually reduces the number of partitions into which the segment is partitioned in *Step 5*.

Step 5: *Recursively partition each curve segment.*

Each smoothed curve segment between adjacent break points is partitioned into two sub-segments, and each sub-segment is represented by the line joining its endpoints. If the mean square error or representation error (average of the square of the vertical distances between the approximating line and points on the curve) between a sub-segment and its approximating line is greater than a pre-specified tolerance ε then the sub-segment is partitioned again into two parts. Otherwise, it is not partitioned further. This recursive process continues until representation error becomes less than ε for all sub-segments. A curve segment may be partitioned at its midpoint or best point. The best point is defined as the point that minimizes the sum of the representation errors of the two sub-segments. The resulting HPLA of each segment is represented by a binary tree. The HPLA partitioning of the time series in Fig. 1 is shown in Fig. 3. It is obtained by recursively partitioning curve segments at midpoint until the mean square error between the curve and the approximating line becomes less than 0.5.

Step 6: *Compute feature vectors.*

In the HPLA representation, curve segment between adjacent break points is represented by a binary tree. Each non-leaf node of the binary tree represents a part of the curve segment, and its child nodes represent its two partitions. Let, *length-l* and *slope-l* denote the length and slope of the line approximating the left partition, and *error-l* denote the root mean square error of the left partition. Similarly, length, slope and error of right partition are *length-r*, *slope-r*, and *error-r*. In this paper, the features used are *length-l/length-r*, *slope-l/slope-r* and *error-l/error-r*. Other features describing relative shape of the two curve segments and proximity of each curve segment to its approximating line may be used. The feature vector of a leaf-node specifies the segment's endpoints.

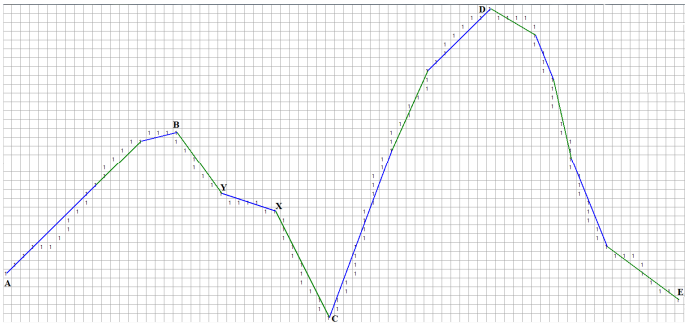

Figure 3. Segmentation of time series in figure 1.

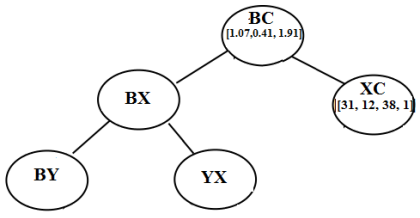

Figure 4. The HPLA representation of segment BC of the time series.

For the purpose of illustrating the computation of feature vectors, consider the binary tree of the segment BC in Fig. 4. The root node represents BC, and its child nodes represent

BX and XC, where X is the mid-point of the chaincode of BC. The length and slope of line BX are 14.21 and -0.82, respectively. The root mean square error between curve BX and line BX is 0.56. Similarly, the length and slope of line XC are 13.41 and -2.0, respectively. The root mean square error between curve XC and line XC is 0.47. Therefore, the 3X1 feature vector of the root node is [1.07 0.41 1.91]. It is also possible to compute features invariant to amplitude scale, amplitude shift, and time scale. The 4X1 feature vector of the leaf-node representing curve XC is [31 12 38 1].

V. ADVANTAGES OF HPLA REPRESENTATION

Many benefits of the HPLA representation are explained with the help of an example. Consider the task of determining the similarity between two time series T and Q. In the simplest case, T and Q are of same length, and there is one-to-one correspondence between their elements. Then almost all representations are able to compute a meaningful measure of similarity between T and Q in the representation space. This is not true if T and Q are unequal in length and their elements do not align. Now, assume that T and Q differ in length and their elements do not align due to translation or difference in temporal scale. The goal is to find $T[a: b]$ and $Q[c: d]$, the largest subsequences of T and Q that are similar to each other. Larger the length of the subsequences greater is the similarity between the two time series. When the non-alignment is only due to translation, T_{a+i} aligns with Q_{c+i}. If the temporal scales of T and Q are different then $T[a: b]$ and $Q[c: d]$ are similar in shape. However, establishing one-to-one correspondence between elements of T and Q is not possible.

The HPLA representation preserves prominent local maxima and minima as break-points, and represents the subsequence between adjacent break-points by a binary tree. The feature vectors of the non-leaf nodes of the binary tree can be invariant to time/amplitude translation and scale. Therefore, it is possible to determine possible correspondence between break-points in T and Q. Then a binary tree matching algorithm may be used for the identification of the longest sequence of binary trees in the HPLA representation of T that matches a sequence of binary trees in the HPLA representation of Q.

The HPLA representation permits the user to choose coarse or fine approximation depending on the level of accuracy needed, and is natural for coarse-fine processing of time series data. The ability to determine similarity by matching individual sections allows flexibility in defining similarity, and supports the development of section based clustering, classification and indexing methods.

VI. EXPERIMANTAL RESULTS

Eleven different data sets (7 data sets from UCR archive [15], one from UC Irvine KDD archive, and 3 stock market data sets) are used in the comparative study. From each data set, 10 time series are selected randomly, and the reconstruction error for the HPLA representation is computed for each of them as described below.

1) The time series is normalized to have zero mean and unit standard deviation.

2) The normalized time series is transformed into a digital curve by digitizing the amplitude values with a bin-size of 0.01.

3) The curve is partitioned into segments by choosing perceptually prominent maxima and minima as break points.

4) The HPLA representation is obtained by recursively partitioning each segment at the best point ($\varepsilon = 5$ in pixels or 0.05 in original values).

5) Using the information in root nodes of segments, an approximation of the time series is reconstructed. The compression ratio (percent) and the mean square error between the original time series and the approximation are calculated.

6) An approximation better than the one in 5 is constructed by using the information in root nodes and their non-leaf child nodes. The compression ratio and reconstruction error for this case are also calculated.

TABLE II. EXPERIMENTAL RESULTS

Data Set	Compression Ratio	Reconstruction Error		
		HPLA	PAA	PLA
Mallat	95.4 92.6	0.01557 0.01331	0.13176 0.06130	0.06825 0.02255
Pseudo Periodic Synthetic	97.2 94.6	0.02205 0.00051	0.08922 0.03560	0.04603 0.00962
OliveOil	93.8 90.2	0.01401 0.00251	0.11397 0.05550	0.08470 0.03365
Adiac	88.5 84.7	0.00359 0.00339	0.03193 0.00894	0.01131 0.00465
Yoga	91.3 87.7	0.00451 0.00132	0.01836 0.00748	0.00646 0.00184
Fish	93.9 90.1	0.00413 0.00235	0.09535 0.08044	0.05453 0.00402
Swedish Leaf	86.8 80.3	0.02613 0.02235	0.13432 0.03474	0.06100 0.02465
OSU Leaf	92.5 88.3	0.02119 0.00893	0.07163 0.04130	0.03406 0.01648
Amazon	90.7 86.4	0.01898 0.00321	0.08219 0.03154	0.02832 0.00552
IBM	87.4 80.8	0.03515 0.02199	0.12166 0.09529	0.05139 0.03357
Microsoft	83.5 79.6	0.03185 0.01811	0.10156 0.06188	0.03763 0.02157

The average compression ratio and reconstruction accuracy for each set given in Table II. For each set there are two entries. The first entry is coarse (step 5), and the second entry is relatively finer than the first entry (step 6). The PAA and PLA representations of each time series are obtained by partitioning time series into equal length segments. Each PLA segment is fitted with the best line using linear regression. The number of segments is adjusted for each representation to achieve the same level of compression as the corresponding HPLA representation. As

 ISBN: 978-1-61208-334-6

expected, the HPLA representation achieved significantly higher reconstruction accuracy for all data sets.

VII. Conclusion and Future Work

This paper has made two primary contributions. First, seven requirements, a good time series representation should satisfy are explicitly identified by analyzing the needs of time series data mining applications. Secondly, a new time series representation (HPLA), which satisfies six of the seven requirements better than PAA, APCA and PLA representations, is proposed. The distance measure that satisfies the lower bound criterion is not known for the HPLA representation.

The experimental results illustrate, for a given compression ratio, the HPLA represents the time series more accurately than the PAA and uniformly segmented PLA representations. A time series can have many PLA representations based on how it is segmented. The representation accuracy, usefulness, and effectiveness for mining time series is determined by the number of break points, and how well the break points are selected during segmentation. The strength of the HPLA representation comes from the novel two-stage segmentation approach, which identifies the perceptually important local maxima and minima as primary break points. These break points provide a broad perception of the shape. They also identify trend changes. Additional break points are placed between primary breakpoints to achieve the desired degree of accuracy.

The HPLA being a multi-level representation, permits coarse-fine processing of time series. Most time series representations do not (effectively) support the finding the longest subsequence of one time series that has a matching (similar) subsequence in the other time series. The problem becomes even more challenging if the two time series do not have the same time and amplitude scale. The HPLA facilitates aligning corresponding segments of the two time series by using perceptually important primary break points as anchor points. The feature vector, which specifies the relative values of slope, length and error of the two partitions of each segment, is invariant to time and amplitude scale. These two features make the HPLA more suitable than other piecewise linear or constant representations for time series matching.

The clustering, classification, and query by content require a representation that facilitates the development of efficient and effective algorithms to determine the similarity between time series. The preliminary research and limited simulation results suggest that the HPLA representation is highly suitable for almost all time series data mining applications including clustering, classification and query by content. Therefore, future research should focus on the development of the HPLA based algorithms for aligning two time series, matching time series, and clustering and classification of time series based on piecewise matching.

References

[1] P. Esling and C. Agon, “Time-series data mining,” ACM Computing Surveys, vol. 45, Dec. 2012, 34 pages.

[2] E. Keogh, K. Chakrabarti, M. Pazzani and S. Mehrotra, “Dimensionality reduction for fast similarity search in large time series databases,” Knowledge and information Systems, vol. 3, pp. 263--286, 2001.

[3] N. Q. Hung and D. T. Anh, “An improvement of PAA for dimensionality reduction in large time series databases,” Proceedings of the 10th Pacific Rim International Conference on Artificial Intelligence: Trends in Artificial Intelligence (PRICAI '08), Jan. 2008, pp. 698-707.

[4] J. Lin, E. Keogh, L. Wei, and S. Lonardi, “Experiencing SAX: a novel symbolic representation of time series,” Data Mining and Knowledge Discovery, Oct. 2007, pp. 107-144.

[5] M. Azzouzi and I. T. Nabney, "Analysing time series structure with hidden Markov models," Neural Networks for Signal Processing VIII, Proceedings of the 1998 IEEE Signal Processing Society Workshop , 1998, pp. 402-408.

[6] F. Korn, H. V. Jagadish, and C. Faloutsos, “Efficiently supporting ad hoc queries in large datasets of time sequences,” Proc. ACM SIGMOD international conference on Management of data, 1997, pp. 289-300.

[7] E. Keogh and M. Pazzani, “An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback,” Fourth International Conference on Knowledge Discovery and Data Mining (KDD'98), 1998, pp. 239-241.

[8] C. Yan, J. Fang, L. Wu, and S. Ma, “An approach of time series piecewise linear representation based on local maximum, minimum and extremum”, Journal of Information & Computational Science, June 2013, pp. 2747-2756.

[9] K. B. Prat and E. Fink, “Search for patterns in compressed time series [J],” International Journal of Image and Graphics, vol. 2, Issue. 1, 2002, pp. 89-106.

[10] S. Park, S. W. Kim, and W. W. Chu, “Segment-based approach for subsequence searches in sequence databases,” In Proceedings ACM symposium on Applied computing (SAC '01), 2001, pp. 248-252.

[11] J. Zhou, G. Ye, D. Yu, “A new method for piecewise linear representation of time series data,” Physics Procedia, vol. 25, 2012, pp. 1097-1103.

[12] D. H. Nabors, A boundary based image segmentation and representation method for binary images, Doctoral Dissertation, The University of Alabama in Huntsville, 2000.

[13] S. K. Kim, Hierarchical representation of edge images for geometric feature based image interpretation, Doctoral Dissertation, The University of Alabama in Huntsville, 2007.

[14] K. Chakrabarti, E. Keogh, S. Mehrotra, and M. Pazzani, "Locally adaptive dimensionality reduction for indexing large time series databases", ACM Transaction on Database Systems. vol. 27, Jun. 2002, pp. 188-228.

[15] E. Keogh, Q. Zhu, B. Hu, Y. Hao, X. Xi, L. Wei and C. A. Ratanamahatana (2011). The UCR Time Series Classification/Clustering. [retrieved: August, 2013] Homepage: www.cs.ucr.edu/~eamonn/time_series_data/

 ISBN: 978-1-61208-334-6