Time Series Data Mining Methods

Sonali Nayak, Bharathi Srinivasan
AIM3 – Scalable Data Science
DIMA/ TU Berlin
07.07.2017

Outline

- Introduction
- Properties and Challenges of Large Time Series
- Preprocessing Methods
  - Representation
  - Indexing
  - Segmentation
  - Visualization
  - Similarity Measures
- Mining in Time Series
  - Clustering and Classification
  - Knowledge Discovery : Pattern Mining
  - Rule Discovery
  - Prediction
- Novel algorithm for time series search – UCR
- Conclusion
- Demonstration – Recent Research

Introduction

- Sequential measurements over time
  - Sensor data: heights of ocean tides, counts of sunspots, etc.
  - Various fields: Medicine, science, entertainment, finance, etc.

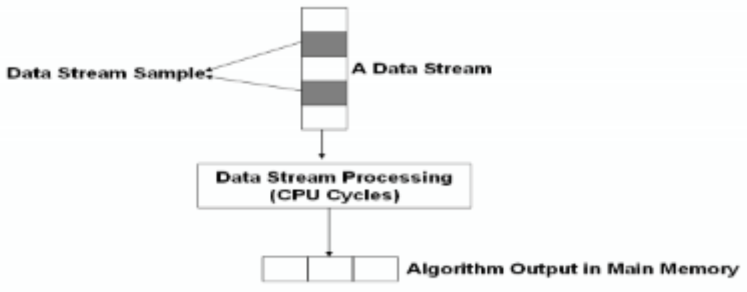
- Desire to look for hidden information
  - Frequently occurring patterns
  - Anomalies or natural groupings

Properties and Challenges

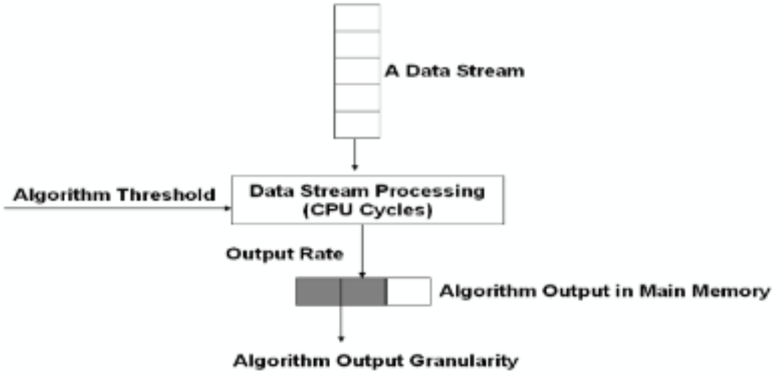
- A Trillion
  - One million million
  - Corresponds to every heartbeat of a 123 year old human
- High Dimensionality
  - One observation of a time series is viewed as one dimension
- Noise along characteristic patterns are additive white noise components
- Temporal and Amplitude shifting differences

Mining Data Streams

- Data Based Solutions
  - Sampling
  - Load Shedding
  - Sketching
  - Synopsis Data Structures
  - Aggregations



- Task Based Techniques
  - Approximation Algorithms
  - Sliding Window
  - Algorithm Output Granularity



- Introduction
- Properties and Challenges of Large Time Series
- **Preprocessing Methods**
  - Representation
  - Indexing
  - Segmentation
  - Visualization
  - Similarity Measures
- Mining in Time Series
  - Clustering and Classification
  - Knowledge Discovery : Pattern Mining
  - Rule Discovery
  - Prediction
- Novel algorithm for time series search – UCR
- Conclusion
- Demonstration – Recent Research

Pre-Processing Techniques
1.Representation

- **Non Data Adaptive Representation** : logic of spectral decomposition
  - Piecewise Aggregate Approximation (PAA) - generated by dividing the time series into ω equi-sized windows and calculating averages of the subsequences in the corresponding windows

$$\hat{x}_i = d^{-1} \sum_{j=d\,(i-1)+1}^{d\,i} x_j, \qquad i = 1, \ldots, \omega, \quad j = 1, \ldots, n, \quad d = n\,w^{-1}$$

- **Data Adaptive Representation :** transformation parameters are chosen depending on the available data and not fixed a priori
  - Symbolic Aggregate Approximation (SAX)
    1. the z-normalized time series is converted to a PAA representation
    2. the PAA coefficients are mapped to symbols (mostly letters)
    3. Symbols are stored as they require fewer bits than real valued numbers

Pre-Processing Techniques
1.Representation

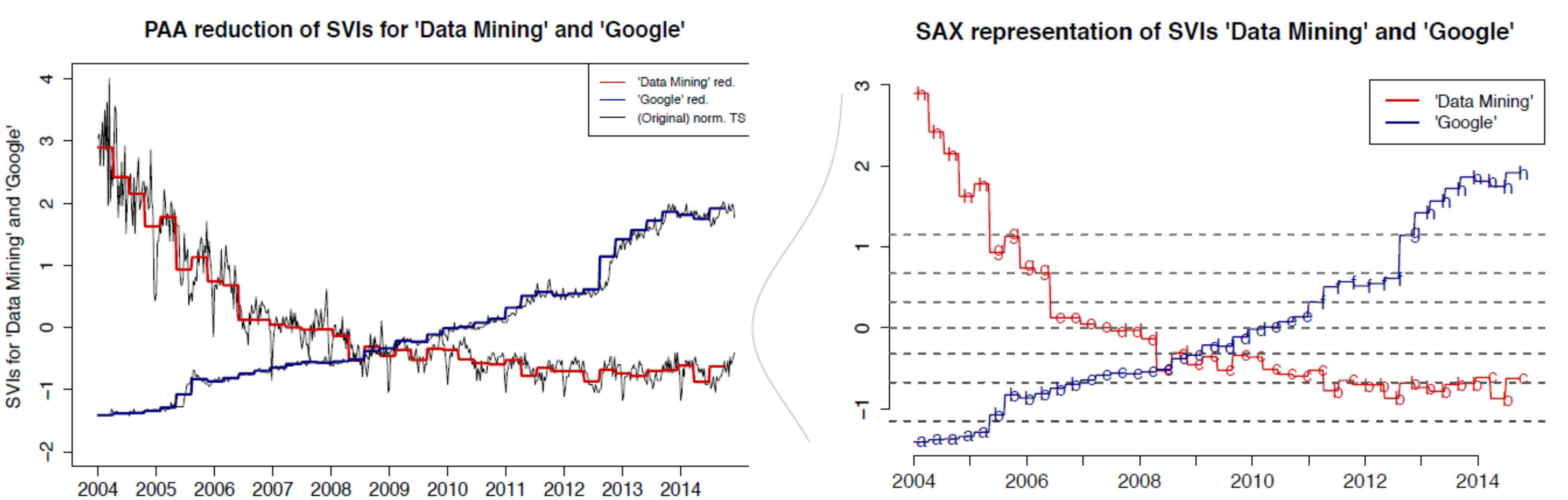


Fig: PAA dimension reduction (left) of Google search volume indices (SVIs) for the query terms “Data Mining” and “Google” with n = 569 and ω = 40 respectively. Corresponding SAX representation (right) with ω = 40 and α = 8. In case of the SVI for the query term “Google”, the raw data is replaced by the word “aaaaabbbbbbccccccccddddeeeghhhhhhh”

- **Model Based Representation** : Assumption that data has been produced by an underlying model
  - Fitting regression lines
  - Autoregressive Moving Average(ARMA)
  - Hidden Markov Models (HMM)

Pre-Processing Techniques
2. Indexing

- Enables quick processing request in large databases

1. Vector based index structures : clusters the vector based compressed sequences into similar groups
2. Metric based index structures : cluster the sequences with respect to relative distances to each other

Pre-Processing Techniques
3. Segmentation

- Discretization problem to accurately approximate the time series

1. Sliding Window: A segment is grown until it exceeds some error bound. The process repeats in the next data point ot included in the the newly approximated segment.

```
Algorithm Seg_TS = Sliding_Window(T , max_error)
anchor = 1;
while not finished segmenting time series
i = 2;
 while  calculate_error(T[anchor: anchor + i ]) < max_error
   i = i + 1;
 end;
 Seg_TS = conca Download  create_segment(T[anchor: anchor + (i-1)]);
 anchor = anchor + 1;
end;
```

Source: An Online Algorithm for segmenting Time Series, Keogh et al. SIGMOD June 2005

Pre-Processing Techniques

3. Segmentation

2. Top-Down: The time series is recursively partitioned until some stopping criteria is met.

```
Algorithm Seg_TS = Top_Down(T , max_error)
best_so_far = inf;
for i = 2 to length(T) - 2  // Find best place to split the time series.
improvement_in_approximation = improvement_splitting_here(T,i);
  if improvement_in_approximation < best_so_far
    breakpoint = i;
    best_so_far = improvement_in_approximation;
  end;
end;
                              // Recursively split the left segment if necessary.
if calculate_error(T[1:breakpoint]) > max_error
  Seg_TS = Top_Down(T[1: breakpoint]);
end;
                              // Recursively split the right segment if necessary.
if calculate_error( T[breakpoint + 1:length(T)] ) > max_error
  Seg_TS = Top_Down(T[breakpoint + 1: length(T)]);
end;
```

Source: An Online Algorithm for segmenting Time Series, Keogh et al. SIGMOD June 2005

Pre-Processing Techniques
3. Segmentation

3. Bottom-Up: Starting from the finest possible approximation, segments are merged until some stopping criteria is met.

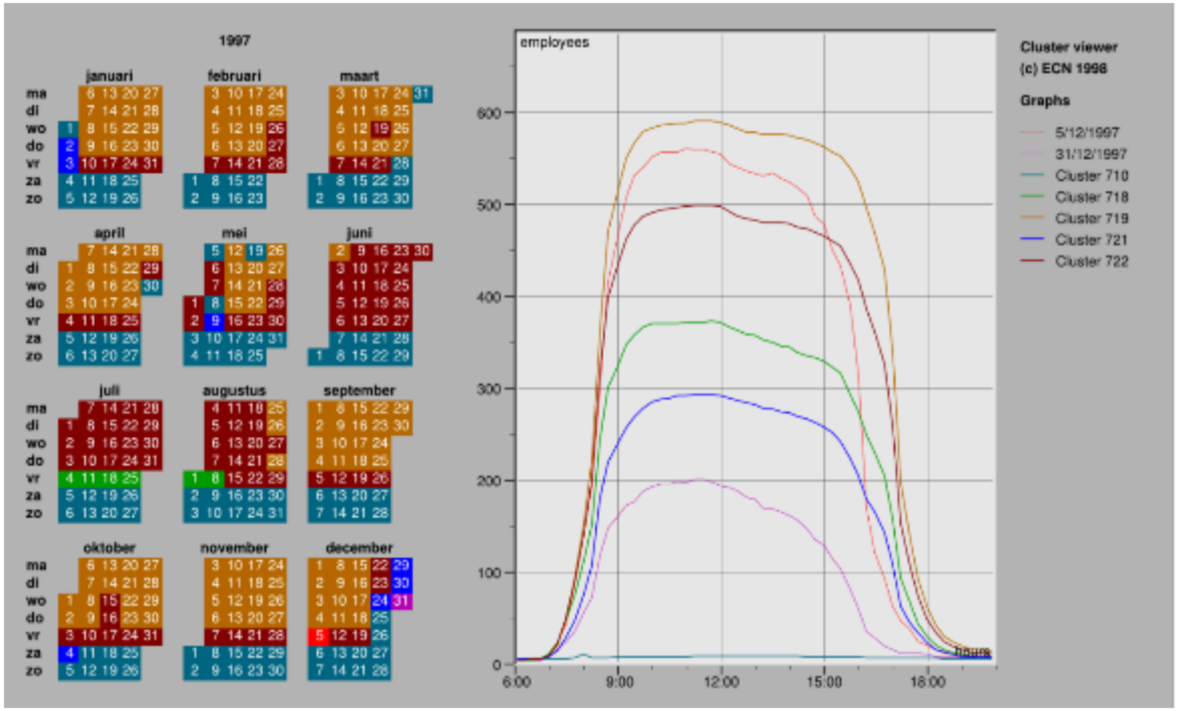
```
Algorithm Seg_TS = Bottom_Up(T , max_error)
for i = 1 : 2 : length(T)          // Create initial fine approximation.
 Seg_TS = concat(Seg_TS, create_segment(T[i: i + 1 ]));
end;
for i = 1 : length(Seg_TS) – 1      // Find cost of merging each pair o
 segments.
 merge_cost(i) = calculate_error([merge(Seg_TS(i), Seg_TS(i+1))]);
end;

while min(merge_cost) < max_error              // While not finished.
 index = min(merge_cost);                      // Find "cheapest" pair to merge
 Seg_TS(index) = merge(Seg_TS(index), Seg_TS(index+1))); // Merge them.
 delete(Seg_TS(index+1));                                // Update records.
 merge_cost(index) = calculate_error(merge(Seg_TS(index), Seg_TS(index+1)));
 merge_cost(index-1) = calculate_error(merge(Seg_TS(index-1), Seg_TS(index)));
end;
```

Source: An Online Algorithm for segmenting Time Series, Keogh et al. SIGMOD June 2005

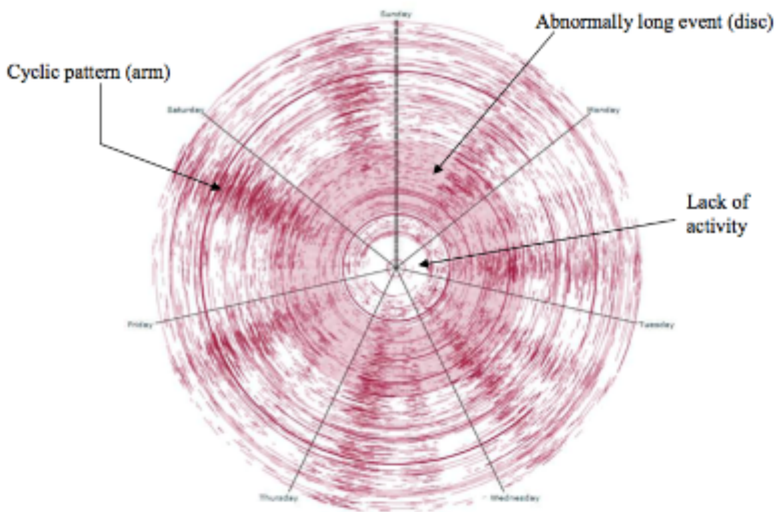
Pre-Processing Techniques
4. Visualization

- Most popular representations of visualization approaches/interfaces:
  - Calendar based Visualization : showing cluster averages of similar sequences as an aggregated representation of the data



Source: An Online Algorithm for segmenting Time Series, Keogh et al. SIGMOD June 2005

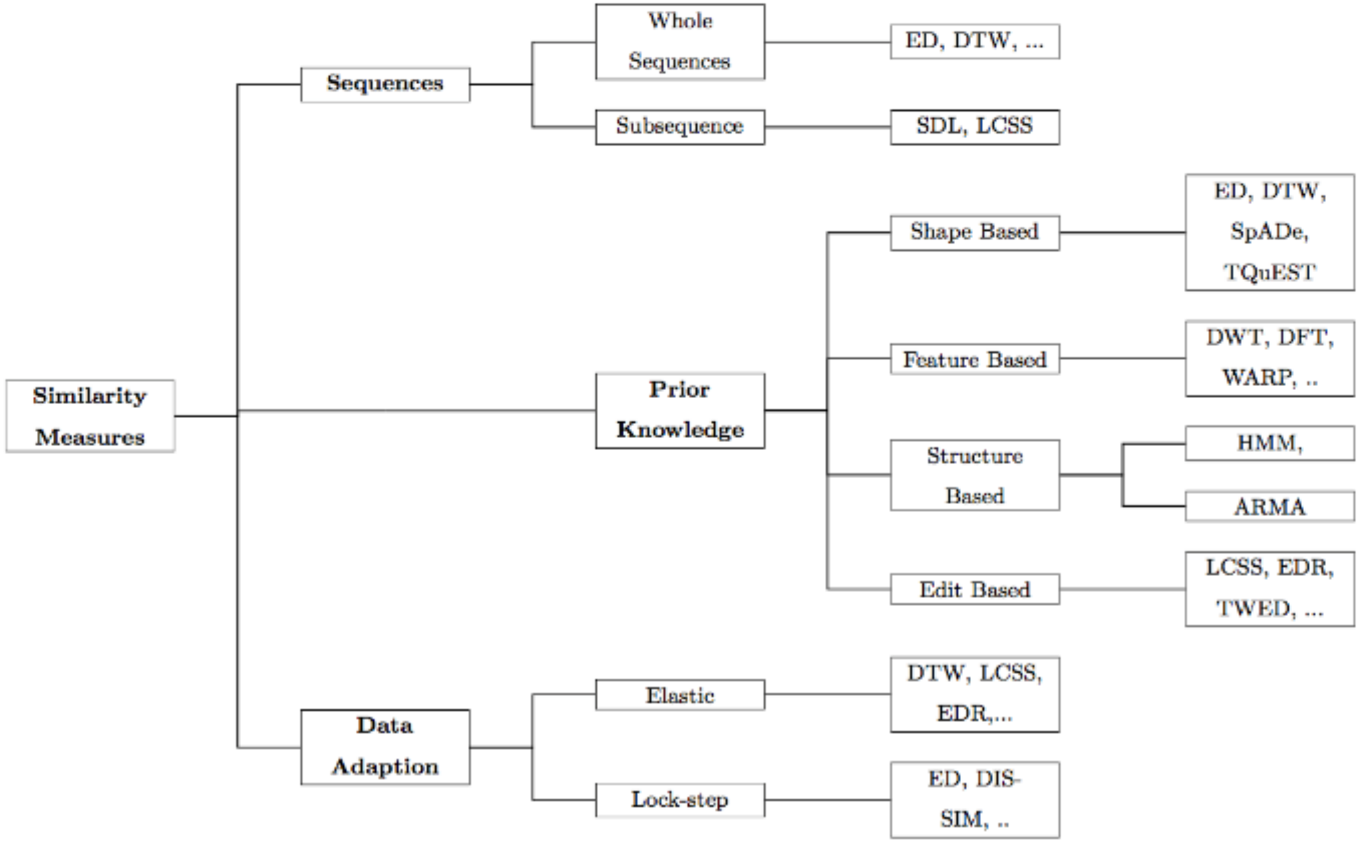
- Spiral : detecting periodic patterns and structures in the data



- Time Searcher : insert query orders which are called “TimeBoxes” for zooming in on certain patterns
- VizTree : provides both, a global visual summary of the whole time series and the possibility to zoom in for interesting subsequences
  - computes a symbolic representation of the data and then builds a suffix tree

Pre-Processing Techniques
5. Similarity Measures

- Backbone and bottleneck of time series data mining
- Shape based, Edit based, Feature based, Structure based



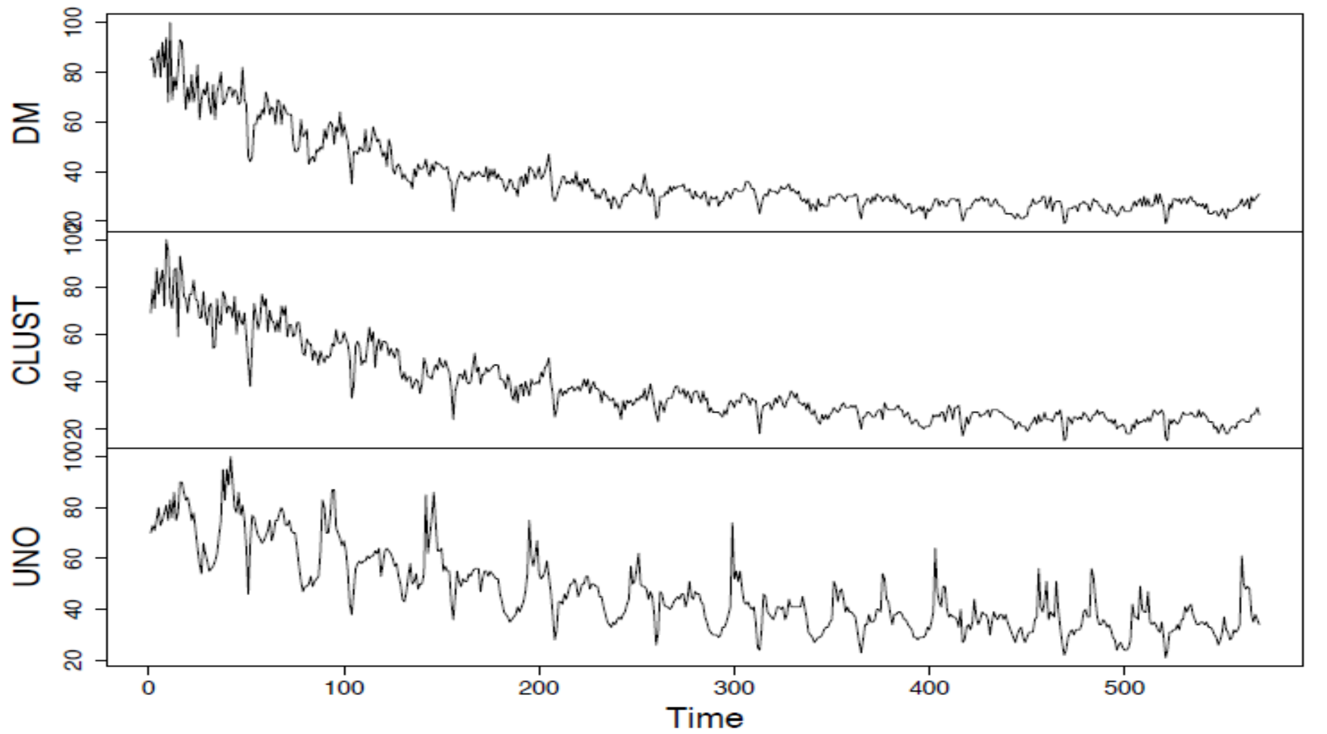
Pre-Processing Techniques
5. Similarity Measures

- Rakthanmanon et al. (2012) were the first to develop a similarity search algorithm based on dynamic time warping that allows mining a trillion time series objects.
- most popular elastic shape based similarity measure
- proposed to handle warps in the temporal dimension
- Until 2012, the main disadvantage of DTW is said to be the computational complexity. DTW computation was too slow to be used for truly massive databases.
  - UCR Suite overcomes this with 4 novel ideas

- Introduction
- Properties and Challenges of Large Time Series
- Preprocessing Methods
  - Representation
  - Indexing
  - Segmentation
  - Visualization
  - Similarity Measures
- **Mining in Time Series**
  - Clustering and Classification
  - Knowledge Discovery : Pattern Mining
  - Rule Discovery
  - Prediction
- Novel algorithm for time series search – UCR
- Conclusion
- Demonstration – Recent Research

Mining in Time Series
1.Clustering

- Clustering is done for supervised and semi supervised learning algorithms.
- Clustering is done to find natural groupings in the data.



Mining in Time Series
2.Knowledge Discovery: Pattern Mining

- knowledge discovery is referred to as the detection of frequently appearing patterns, novelties and outliers or deviants in a time series database.
- Novelties – Anomalies or Surprising Patterns
- Pattern Discovery – Also known as Motifs, can be found by Clustering.
- Outliers – Individually surprising data points.
- Surprising patterns – Collections of time series data points which are collectively surprising or show anomalies.
- Partial periodic patterns can be classified into two types:
  - regular patterns – patterns exhibiting periodic behaviour throughout a series with some exceptions and
  - recurring patterns – patterns exhibiting periodic behaviour only for particular time intervals within a series. Past studies on partial periodic search have been primarily focused on finding regular patterns

Mining in Time Series
3.Rule Discovery

- Rule discovery is the approach to detect relations between variables, time series sequences or patterns that typically appear in a very large database.
- The very basic idea of association rules, is to search for correlations between variables.
- Some Association rule algorithms are - Apriori, Eclat or FP-growth. (Hipp et al. 2000)
- Rule mining in time series data, can broadly distinguish between the discovery of temporal rules and temporal patterns.

Mining in Time Series
4.Prediction

- Prediction or forecasting of the next few values of a time series is an extensively applied task.
- The most frequently used prediction techniques are:
  - ARMA models(Auto Regressive Integrated Moving Average)
  - SARIMA models(Seasonal Auto Regressive Integrated Moving Average)
  - Neural networks
  - Hidden Markov models
- The machine learning approaches applicable to time series forecasting are:
  - Multilayer perceptron
  - Bayesian neural networks
  - Radial basis functions
  - Kernel regression
  - k-nearest neighbour regression
  - Regression trees
  - Support vector regression
  - Gaussian process regression
- According to Ahmed et al. Gaussian process regression and Multilayer perceptron are the best approaches.

The UCR Suite

- This paper presents a collection of 4 optimization to DTW algorithm that enables exact searching which is very fast and efficient collectively called as the UCR suite.
- The time series data requires similarity comparison as a subroutine and one way to do that as illustrated in this paper is DTW.
- A Nearest Neighbour Classifier can achieve state-of-the-art performance when using Dynamic Time Warping as a distance measure.

The UCR Suite (Contd.)

- The 4 optimisations are:
  - Eliminating Square roots
  - Pruning to compute cheap lower bounds
  - Early abandoning of ED and LB_Keogh computation
  - Early abandoning of DTW
- Some explicit assumptions:
  - Time series Sub-sequences must be Normalized
  - DTW is the best measure for most Domains
  - Arbitrary query lengths cannot be Indexed
  - For some data Mining problems there can be some waiting hours to Answer

Application areas

- Robotics
- Medicine
- Climatology
- Biometrics
- Music/Speech Recognition
- Gesture Recognition
- Aviation
- And so on…

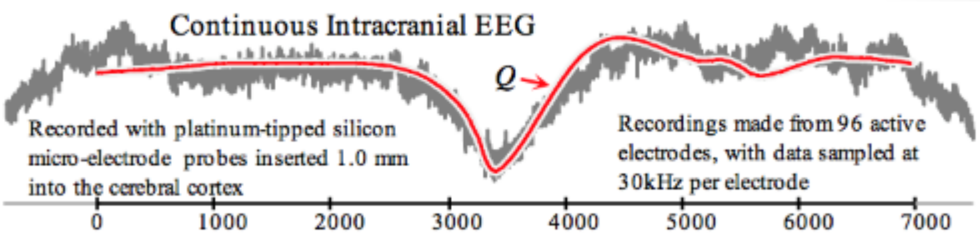


Fig: Searching massive archives of EEG data for epileptic spikes

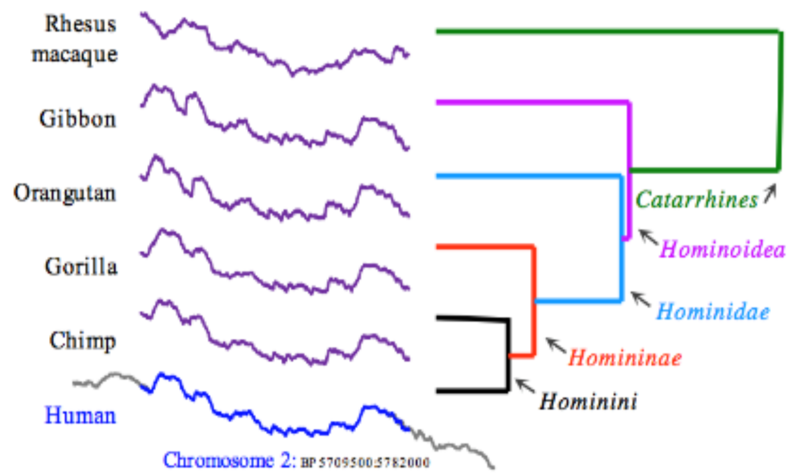


Fig: A subsequence of DNA from Human chromosome 2, of length 72,500 beginning at 5,709,500 is clustered using single linkage with its Euclidean distance nearest neighbors from five other primates

UCR Algorithm

Algorithm	Similarity Search
Procedure	[*nn*] = *SimilaritySearch* (T,*Q*)

```
best-so-far ← ∞, count ← 0
Q ← z-normalize(Q)
while !next(T)
    i ← mod(count,m)
    X[i] ← next(T)
    ex ← ex+X[i],   ex2 ← ex2+X[i]^2
    if count ≥ m-1
        μ ← ex/m,   σ ← sqrt(ex2/m - μ^2)
        j ← 0, dist ← 0
        while j < m and dist < best-so-far
            dist ← dist + (Q[j]-(X[mod(i+1+j,m)]-μ)/σ)^2
            j ← j+1
        if dist < best-so-far
            best-so-far ← dist,   nn ← count
        ex ← ex-X[mod(i+1,m)]
        ex2 ← ex2-X[mod(i+1,m)]^2
    count ← count+1
```

Experimental Results

- Time taken to search increasingly large dataset with queries of length 128.

	Million (*Seconds*)	Billion (*Minutes*)	Trillion (*Hours*)
UCR-ED	0.034	0.22	3.16
SOTA-ED	0.243	2.40	39.80
UCR-DTW	0.159	1.83	34.09
SOTA-DTW	2.447	38.14	472.80

- Time to search 303,523,721,928 EEG data-points, with queries of length 7000.

Note that only ED is considered here because DTW may produce false positives caused by eye blinks		**UCR-ED**	**SOTA-ED**
	EEG	3.4 hours	494.3 hours

Experimental Results(Contd.)

- Time taken to search one year of EEG data with query length of 421

	UCR-ED	SOTA-ED	UCR-DTW	SOTA-DTW
ECG	4.1 minutes	66.6 minutes	18.0 minutes	49.2 hours

Experimental Demo

- **Input**
  - UCR_DTW (or UCR_ED): the executable file of our algorithm
  - Data_File: the data file containing a long time series
  - Query_File: the query file containing the query, Q
  - M: length of the query time series. It usually is |Q|, but can be smaller.
  - R: size of warping windows. The value is in range 0-1, e.g., R=0.05 means windows of size +/-5%.

- **Output**
  - Location: starting location of the nearest neighbour of the given query, of size M, in the data file. Note that location starts from 0.
  - Distance: the distance between the nearest neighbour and the query.
  - Data Scanned: number of data in the input data file.
  - Pruned by LB: the number of subsequence can be pruned by using that LB (in percentage).
  - DTW Calculation: the number DTW calculation has been made in total.
  - Note that, when approximate one million data points have been read, one dot "." will be displayed on screen (only in UCR_DTW).

Conclusion

- Real world time series data sets can take a size up to a trillion observations and even more
- The overall goal is to detect new information that is hidden in these massive data sets.
- The lack of well established (best) practices in time series data mining becomes evident as a plethora of sparsely systematized different approaches exist and they are accepted alike in spite of their very distinct strategies.

List of Abbreviations

- AR(I)MA : Autoregressive (Integrated) Moving Average
- DFT: Discrete Fourier Transform
- DTW: Dynamic Time Warping
- DWT: Discrete Wavelet Transform
- ED: Euclidean Distance
- EDR: Edit Distance on Real Sequence
- HMM: Hidden Markov Model
- LCSS: Longest Common Subsequence
- SpADe: Spatial Assembling Distance
- TQuEST: Threshold Query Execution
- TWED: Time Warp Edit Distance

References

- Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P., and Uthurusamy, R. (1996). Advances in knowledge discovery and data mining.
- Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., Zakaria, J., and Keogh, E. (2012). Searching and mining trillions of time series subsequences under dynamic time warping. In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 262–270. ACM.
- Esling, P. and Agon, C. (2012). Time-series data mining. ACM Computing Surveys (CSUR), 45(1):12.
- Lin, J., Keogh, E., and Lonardi, S. (2005). Visualizing and discovering non-trivial patterns in large time series databases. Information visualization, 4(2):61–82
- Berndt, D. J. and Cli↵ord, J. (1994). Using dynamic time warping to find patterns in time series. In KDD workshop, volume 10, pages 359–370. Seattle, WA
- Mining Data Streams: A Review , Gaber et al. SIGMOD June 2005
- Keogh, E., Chakrabarti, K., Pazzani, M., and Mehrotra, S. (2001a). Dimensionality reduction for fast similarity search in large time series databases. Knowledge and information Systems, 3(3):263–286

References

- Keogh, E., Chakrabarti, K., Pazzani, M., and Mehrotra, S. (2001b). Locally adaptive dimensionality reduction for indexing large time series databases. ACM SIGMOD Record, 30(2):151–162
- Keogh, E., Chu, S., Hart, D., and Pazzani, M. (2004a). Segmenting time series: A survey and novel approach. Data mining in time series databases, 57:1–22
- Korn, F., Jagadish, H. V., and Faloutsos, C. (1997). E ciently supporting ad hoc queries in large datasets of time sequences. ACM SIGMOD Record, 26(2):289–300
- Keogh, E., Chu, S., Hart, D., and Pazzani, M. (2001c). An online algorithm for segmenting time series. In ICDM, Proceedings IEEE International Conference on Data Mining, pages 289–296. IEEE
- Van Wijk, J. J. and Van Selow, E. R. (1999). Cluster and calendar based visualization of time series data. In Information Visualization, Proceedings IEEE Symposium on, pages 4–9. IEEE
- Weber, M., Alexa, M., and Mu¨ller, W. (2001). Visualizing time-series on spirals. In Information Visualization, IEEE Symposium on, pages 7–7. IEEE Computer Society
- Hochheiser, H. and Shneiderman, B. (2004). Dynamic query tools for time series data sets: timebox widgets for interactive exploration. Information Visualization, 3(1):1–18

References

- Fu, T.-c. (2011). A review on time series data mining. Engineering Applications of Artificial Intelligence, 24(1):164–181
- Ahmed, N. K., Atiya, A. F., Gayar, N. E., and El-Shishiny, H. (2010). An empirical comparison of machine learning models for time series forecasting. Econometric Reviews, 29(5-6):594–621
- R. Uday Kiran, Haichuan Shang, Masashi Toyoda and Masaru Kitsuregawa (2015). Discovering Recurring Patterns in Time Series.
- Diego F. Silva Gustavo E. A. P. A. Batista (2015). Speeding Up All-Pairwise Dynamic Time Warping Matrix Calculation.
- Hipp, J., G¨untzer, U., and Nakhaeizadeh, G. (2000). Algorithms for association rule mining— a general survey and comparison. ACM sigkdd explorations newsletter
- A. Fu, E. Keogh, L. Lau, C. Ratanamahatana, and R. Wong. 2008. Scaling and time warping in time series querying. *VLDB J*. 17, 4, 899-921.