The Multi-Resolution Extended Edit Distance

Muhammad Marwan Muhammad Fuad
VALORIA, Université de Bretagne Sud
BP 573, 56017 Vannes
marwan.fuad@univ-ubs.fr

Pierre-François Marteau
VALORIA, Université de Bretagne Sud
BP 573, 56017 Vannes
pierre-francois.marteau@univ-ubs.fr

ABSTRACT

Similarity search is a fundamental problem in information technology. The main difficulty of this problem is the high dimensionality of the data objects. In large time series databases, it's important to reduce the dimensionality of these data objects, so that we can manage them. Symbolic representation is a promising technique of dimensionality reduction. In this paper we propose a new distance metric, which is applied to symbolic sequential data objects, and we test it on time series databases in classification task experiments. We also compare it to other distances that are well known in the literature for symbolic data objects, and we prove that it's metric.

Keywords
Information Retrieval, Time Series, Symbolic Representation.

1. INTRODUCTION

The problem of similarity search and information retrieval in large databases has attracted much attention lately, because it has a large number of applications in many fields of research. In most cases, the databases in question are very large, so using linear scanning in the similarity search can take a long time and can become ineffective. Research in this area has focused on its different aspects. One of these aspects is the distance metric used to measure the similarity between two data objects. Many distance metrics have been suggested. In time series databases, the most famous distance is the Euclidean distance, which is effective, but it has a few inconveniences; it is sensitive to noise and shifts on the time axis. It also requires many calculations, and is applied to series of identical lengths only [9]. Another feature of the similarity search problem is data representation. In multimedia IR the main problem we encounter is the so called "dimensionality curse". One of the best solutions to deal with this problem is to utilize a dimensionality reduction technique, then to utilize a suitable indexing structure on the reduced data objects. There have been different suggestions to represent time series. To mention a few; Discrete Fourier Transform (DFT) [1] and [2], Discrete Wavelet Transform (DWT) [3], Singular Value Decomposition (SVD)[8], Adaptive Piecewise Constant Approximation (APCA) [7], Piecewise Aggregate Approximation (PAA) [6] and [11],...etc. Symbolic representation is a dimensionality reduction technique that has many interesting advantages, because it allows using the ample text-retrieval algorithms and techniques. However, the first papers about symbolic representation of time series mainly addressed questions concerning the discretization scheme and the size of the alphabet [7].

There are several distance measures that apply to symbolic data. In the beginning these measures were restricted to data structures whose representation is naturally symbolic (DNA and proteins sequences, textual data…etc). But later these symbolic measures were also applied to other data structures that can be transformed into strings. There are quite a few distance metrics that apply to symbolically represented data. One of these measures is the edit distance (ED) [10], which is defined as the minimum number of delete, insert, and substitute (change) operations needed to transform string S into string T. This distance is the main distance measure used to compare two strings. Different variations of this distance were proposed later, to name a few; the edit distance on real sequence (EDR) [4], the edit distance with real penalty (EDRP) [4],and many others. The edit distance has a main drawback, in that it penalizes all change operations in the same way, without taking into account the character that is used in the change operation. In order to overcome this drawback we could predefine tables that give the costs of all possible change operations. But this approach is inflexible and highly dependent on the alphabet used. In this paper we propose a new distance metric that applies to strings. We call it "The Multi-Resolution Extended Edit Distance" (MREED). This distance adds new features to the well-known edit distance by adding additional terms to it. The new distance has a main advantage over the edit distance in that it deals with the above mentioned problem straightforwardly since there is no need to predefine a cost function for the change operation. This distance can, by itself, detect if the change operations use characters that are "familiar" or "unfamiliar" to the two strings concerned.

The rest of this paper is organized as follows: in section 2 we present a motivating example and the proposed distance with a related theorem. Section 3 provides the complexity analysis of the proposed distance. Section 4 contains the experiments that we conducted to test the proposed distance. In section 5 we discuss our distance in light of the results obtained. In section 6 we present conclusions and some future work.

INFOSCALE 2008, June 4-6, Vico Equense, Italy

DOI 10.4108/ICST.INFOSCALE2008.3508

2. THE PROPOSED DISTANCE

2.1 Introduction

Example1: Given the following strings: $S_1 = abca$, $S_2 = aabbcc$, $S_3 = adbecf$. Intuitively, we see that S_2 is closer to S_1 than S_3. Yet, if we calculate their edit distance we see that: $ED(S_1,S_2) = ED(S_1,S_3) = 3$. The reason for this is that the edit distance is based on local procedures, both in the way it is defined and in the algorithm used to compute it.

2.2 Definition- The Multi-Resolution Extended Edit Distance

Let $\sum$ be a finite alphabet, and let $f_i^{(S)}$, $f_i^{(T)}$ be the frequency of the character i in S and T, respectively. $ff_{ij}^{(S)}$, $ff_{ij}^{(T)}$ be the frequency of the two-character subsequence ij in S and T, respectively (including the case where $i = j$), and where S, T are two non-empty strings on $\sum$. The Multi-Resolution Extended Edit Distance (MREED) is defined as;

$$MREED\ (S,T) = ED(S,T) + \lambda\left[|S| + |T| - 2\sum_i \min(f_i^{(S)}, f_i^{(T)})\right]$$
$$+\delta\left[|S| + |T| - 2(\sum_i \sum_j \min(ff_{ij}^{(S)}, ff_{ij}^{(T)}) + 1)\right]$$

Where $|S|$, $|T|$ are the lengths of the strings S,T respectively and where $\lambda \geq 0$, $\delta \geq 0$ ($\lambda, \delta \in R$). We call λ the frequency factor of the first degree, and δ the frequency factor of the second degree.

2.3 Theorem : MREED is a distance metric.

Before we prove the theorem, we can easily notice that;

$$\lambda\left[|S| + |T| - 2\sum_i \min(f_i^{(S)}, f_i^{(T)})\right] \geq 0 \qquad \forall S,T \qquad (1)$$

$$\delta\left[|S| + |T| - 2(\sum_i \sum_j \min(ff_{ij}^{(S)}, ff_{ij}^{(T)}) + 1)\right] \geq 0 \qquad \forall S,T \qquad (2)$$

In order to prove the theorem we have to prove that;

i) $MREED(S,T) = 0 \Leftrightarrow S = T$

i. a) $MREED(S,T) = 0 \Rightarrow S = T$

-Proof: If $MREED(S,T) = 0$, and taking into account (1) and (2), we get the following relations:

$$\lambda\left[|S| + |T| - 2\sum_i \min(f_i^{(S)}, f_i^{(T)})\right] = 0 \qquad (3)$$

$$\delta\left[|S| + |T| - 2(\sum_i \sum_j \min(ff_{ij}^{(S)}, ff_{ij}^{(T)}) + 1)\right] = 0 \qquad (4)$$

$$ED(S,T) = 0 \qquad (5)$$

From (5), and since ED is metric we get: $S = T$

i. b) $S = T \Rightarrow MREED(S,T) = 0$ (obvious).

From i. a) and i. b) we get $MREED(S,T) = 0 \Leftrightarrow S = T$

ii) $MREED(S,T) = MREED(T,S)$ (obvious).

iii) $MREED\ (S,T) \leq MREED\ (S,R) + MREED\ (R,T)$

-Proof: $\forall S,T,R$, we have:

$$ED(S,T) \leq ED(S,R) + ED(R,T) \qquad (6)$$

(Valid since ED is metric).

We also have:

$$\begin{aligned}
&\lambda[|S| + |T| - 2\sum_i \min(f_i^{(S)}, f_i^{(T)})] \leq \\
&\lambda[|S| + |R| - 2\sum_i \min(f_i^{(S)}, f_i^{(R)})] + \qquad (7)\\
&\lambda[|R| + |T| - 2\sum_i \min(f_i^{(R)}, f_i^{(T)})]
\end{aligned}$$

$$\begin{aligned}
&\delta[|S| + |T| - 2(\sum_i \sum_j \min(ff_{ij}^{(S)}, ff_{ij}^{(T)}) + 1)] \leq \\
&\delta[|S| + |R| - 2(\sum_i \sum_j \min(ff_{ij}^{(S)}, ff_{ij}^{(R)}) + 1)] + \qquad (8)\\
&\delta[|R| + |T| - 2(\sum_i \sum_j \min(ff_{ij}^{(R)}, ff_{ij}^{(T)}) + 1)]
\end{aligned}$$

(See the appendix for a brief proof of (7) and (8))

Adding (6), (7) and (8) side to side we get:
$MREED(S,T) \leq MREED(S,R) + MREED(R,T)$.

From i), ii), and iii) we conclude that the theorem is valid.

Example 2: (Revisiting the example presented in section 2.1)

$MREED(S_1,S_2) = 9$ while $MREED(S_1,S_3) = 15$. So we see that our distance could detect that S_2 is closer to S_1 than S_3

3. COMPLEXITY ANALYSIS

The time complexity of MREED is $O(m \times n)$, where m is the length of the first string and n is the length of the second string, or $O(n^2)$ if the two strings are of the same lengths. In order to make MREED scale well when applied to time series, we can find a symbolic representation method that would allow high compression of the time series (leading to drastic length reduction), with acceptable accuracy

4. EXPERIMENTS

As mentioned earlier, this new distance metric is applied to data structures which are represented symbolically. It's important to mention here that we believe that bioinformatics or textual data bases are the ideal data structures to apply the MREED to. However, and since our field of research is time series, we had to test MREED on time series data bases.

We tested our distance in a classification task on 12 datasets chosen from the 20 datasets available in UCR [12]. We used leaving-one-out cross validation. We meant to include quite a variety of cases in our tests; the number of classes varies between 2 (Gun-Point) and 50 (50words). The size of the training set varies between 30 (Beef and CBF) and 560 (Face all). The size of the testing set varies between 30 (Beef and Olive Oil) and 3000 (Yoga), and the length of the time series (before compression) varies between 60 (Synthetic Control) and 570 (Olive Oil)

Time series are not naturally represented symbolically. But a few methods have been proposed to present them as strings. One of the most famous methods in the literature is SAX [5]. SAX is based on the idea that normalized time series have highly Gaussian distribution, so one can determine breakpoints that will produce equal-sized areas under the Gaussian curve. SAX is applied as follows: the time series are normalized, then the PAA representation of the time series is obtained by dividing the data into w equal sized frames, then the mean of each frame is calculated. These means constitute the PAA approximation. The next step is the discretization of the PAA to get a discrete representation of the times series. This is achieved by benefiting from the Gaussian distribution of the normalized time series. The last step of SAX is using the following distance measure;

$$MINDIST(\hat{S},\hat{T}) \equiv \sqrt{\frac{n}{w}}\sqrt{\sum_{i=1}^{w}(dist(\hat{s}_i,\hat{t}_i))^2}$$

where n is the length of the original time series, w is the length of the strings (the number of the frames) , $\hat{S}$ and $\hat{T}$ are the symbolic representations of the two time series S and T respectively, and the function $dist()$ is implemented by using the appropriate statistical lookup table. After reaching this last step, SAX converts the resulting strings into numeric values so that the MINDIST can be calculated. The main of idea of our experiments is that instead of converting these strings into numeric values, we can use a distance that is directly applied to strings. To test MREED we proceed in the same way as SAX to get a symbolic representation of the time series, then we replace MINDIST with MREED, we also tested replacing MINDIST with ED for comparison reasons. It's important to mention that SAX (When we refer to SAX from now on in this paper, we mainly mean the distance measure used in SAX) is a method that is designed directly to be used on time series, so it's a very competitive method. In order to make a fair comparison, we used the same compression ratio that was used to test SAX (i.e. 1:4). We also used the same range of alphabet size [3, 10]. We chose to compare it to the 1-NN Euclidean distance, since SAX was compared to it. The Euclidean distance, of course, is calculated using the original time series. Both ED and SAX have one parameter, which is the alphabet size. MREED has two extra parameters; the frequency factor of the first degree λ, and the frequency factor of the second degree δ. For each of the 12 datasets we optimize the two parameters λ and δ on the training sets to get the optimal values of these parameters; i.e. the values that minimize the error rate. Then we utilize these optimal values on the testing set to get the error rate for each method and for each dataset. The final results of our experiments are shown in Table. 1. (There's no training phase for the Euclidian distance). The best method of the three is highlighted (the figures in red are the cases when the Euclidean distance gave the best results). For simplicity, we optimize the two parameters λ and δ in the interval $[0,1]$ only (step=0.25), except in the cases where there is strong evidence that the error is decreasing monotonously as λ or δ increases.

When comparing a method with another one there are two statistical parameters to be used, one of them is the mean error; the smaller the mean error is, the better the method is. Another statistical parameter is the standard deviation (STD). The importance of this latter is to show how robust the method is (i.e. can be applied to as many different types of datasets as possible). Here also, the smaller the STD is, the better the method is. The results obtained show that the average error is the smallest for MREED, it's even smaller than that of the Euclidean distance. They also show that of all the three tested methods (ED, MREED, and SAX) MREED has the minimum standard deviation, which means that MREED is the most robust one of the three tested methods. It's worth to mention since the Euclidian distance is applied to the raw data, where there's is no compression of information, so it may give better results in some cases than distances applied to symbolic, compressed data.

5. DISCUSSION

1-In the experiments we conducted we had to use time series of equal lengths for comparison reasons only, since SAX can be applied only to strings of equal lengths. But MREED (and ED, too) can be applied to strings of different lengths

2- We didn't conduct experiments for alphabet size=2 because SAX is not applicable in this case

3- In order to represent the time series symbolically, we had to use a technique prepared for SAX. Nevertheless, a representation technique prepared specifically for MREED may even give better results.

4-The improvement of MREED over the edit distance may seem relatively small. We think that the main reason for this is because our distance is mainly useful in the cases where the frequency of sub-sequences plays the main role in the similarity search (like, for example, when the similarity is based on repetitions of a sequence of certain amino-acids in a protein).

6. CONCLUSIONS AND FUTURE WORK

In this paper we presented a new distance metric applied to strings. The main feature of this distance is that it considers the frequency of characters and sub-sequences, which is something other distance measures do not consider. Another important feature of this distance is that it is a metric. We tested our distance on a time series classification task, and we compared it to two other distances. We showed that our distance gave better results, even when compared to a method (SAX) that is designed mainly for symbolically represented time series.

A possible future work is to use MREED in motif discovery in time series data mining, by representing the motif symbolically and applying MREED by using the frequency of the motif rather than the frequency of symbols or subsequences.

Table 1. The error rate of ED, MREED, SAX , together with the Euclidean distance on the testing sets of the 12 datasets. The parameters used in the calculations are those that give optimal results on the training tests.

	1-NN Euclidean distance	The edit distance (ED)	The multi-resolution extended edit distance (MREED)	SAX
Synthetic Control	0.12	0.037 α* =7	0.053 α =8, λ=0, δ=0.25	0.033 α =10
Gun-Point	0.087	0.073 α =4	0.06 α =4, λ =0.25, δ=0	0.233 α =10
CBF	0.148	0.029 α =10	0.023 α =3, λ =0.25,0.5, δ=0.25	0.104 α =10
Face (all)	0.286	0.324 α =7	0.324 α =7=, λ =0, δ=0	0.319 α =10
OSULeaf	0.483	0.318 α =5	0.302 α =5, λ =0, δ=0.25	0.475 α =9
SwedishLeaf	0.213	0.344 α =7	0.365 α =7, λ =0.25, δ=0	0.490 α =10
50words	0.369	0.266 α =7	0.266 α =7, λ =0, δ=0	0.327 α =9
Trace	0.24	0.11 α =10	0.02 α =6, (λ =0, δ≥0.75), (λ =0,0.25, δ=1)	0.42 α =10
Adiac	0.389	0.701 α =7	0.642 α =9, λ =0.5, δ=0	0.903 α =10
Yoga	0.170	0.155 α =7	0.155 α =7, λ =0, δ=0	0.199 α =10
Beef	0.467	0.467 α =4	0.367 α =4, λ =0.5, δ=0.25	0.533 α =10
OliveOil	0.133	0.467 α =9	0.367 α =9,(λ=0.75,δ≥0.5), (λ =1, δ≥0.75)	0.833 ∀ α
MEAN	0.259	0.274	0.245	0.406
STD	0.138	0.205	0.189	0.265

(*: α is the alphabet size)

7. REFERENCES

[1] Agrawal, R., Faloutsos, C., & Swami, A. 1993. Efficient similarity search in sequence databases". Proceedings of the 4th Conf. on Foundations of Data Organization and Algorithms.

[2] Agrawal, R., Lin, K. I., Sawhney, H. S. and Shim, K,. 1995

Fast similarity search in the presence of noise, scaling, and translation in time-series databases, in Proceedings of the 21st Int'l Conference on Very Large Databases. Zurich, Switzerland, pp. 490-501.

[3] Chan, K. & Fu, A. W.1999. Efficient Time Series Matching by Wavelets. In proc. of the 15th IEEE Int'l Conf. on Data Engineering. Sydney, Australia, Mar 23-26.

[4] Chen, L., Özsu, M.T. and Oria,V.2003. Robust and efficient similarity search for moving object trajectories. In CS Tech. Report. CS-2003-30, School of Computer Science, University of Waterloo.

[5] Jessica Lin, Eamonn J. Keogh, Stefano Lonardi, Bill Yuan-chi Chiu, 2003. A symbolic representation of time series, with implications for streaming algorithms. DMKD 2003 2-11

[6] Keogh, E,. Chakrabarti, K,. Pazzani, M. & Mehrotra. 2000. Dimensionality reduction for fast similarity search in large time series databases. J. of Know. and Inform. Sys.

[7] Keogh, E,. Chakrabarti, K,. Pazzani, M. & Mehrotra.2001. Locally adaptive dimensionality reduction for similarity search in large time series databases. SIGMOD,

pp 151-162

[8] Korn, F., Jagadish, H & Faloutsos. C.1997. Efficiently supporting ad hoc queries in large datasets of time sequences. Proceedings of SIGMOD '97, Tucson, AZ,

pp 289-300

[9] Megalooikonomou, V., Wang, Q., Li, G. & Faloutsos, C. 2005.Multiresolution Symbolic Representation of Time Series. In proceedings of the 21st IEEE International Conference on Data Engineering (ICDE). Tokyo, Japan. (Apr 5-9. 2005).

[10] Wagner, R.,A., Fischer, M. J.1974. The String-to-String Correction Problem, Journal of the Association for Computing Machinery, Vol. 21, No. I, (January 1974), pp. 168--173

[11] Yi, B,K., & Faloutsos, C.2000. Fast time sequence indexing for arbitrary Lp norms. Proceedings of the 26st International Conference on Very Large Databases, Cairo, Egypt,

[12] UCR Time Series Classification/Clustering Page

http://www.cs.ucr.edu/~eamonn/time_series_data/

APPENDIX

Lemma : Let $\sum$ be a finite alphabet, $f_i^{(S)}$, $f_i^{(T)}$ be the frequency of the character i in S , T , respectively, where S and T are two strings represented by $\sum$.

Let; $$D(S,T) = |S| + |T| - 2\sum_i \min(f_i^{(S)}, f_i^{(T)})$$

Then $\forall S_1, S_2, S_3$ we have:

$$D(S_1,S_2) \le D(S_1,S_3) + D(S_3,S_2) \qquad \text{(A1)}$$

For all n, where n is the number of characters used to represent the strings

Proof: We will prove the above lemma by induction.

i) Basic step: $n = 1$, this is a trivial case.

Given three strings S_1, S_2, S_3 represented by the same character a .Let S_1^a, S_2^a, S_3^a be the frequency of a in S_1, S_2, S_3, respectively. We have six configurations in this case;

1) $S_1^a \le S_2^a \le S_3^a$, 2) $S_1^a \le S_3^a \le S_2^a$, 3) $S_2^a \le S_1^a \le S_3^a$

4) $S_2^a \le S_3^a \le S_1^a$, 5) $S_3^a \le S_1^a \le S_2^a$, 6) $S_3^a \le S_2^a \le S_1^a$

We will prove that relation (A1) holds in these six configurations.

1) $S_1^a \le S_2^a \le S_3^a$.

In this case we have: $\min(S_1^a, S_2^a) = S_1^a$,

$\min(S_1^a, S_3^a) = S_1^a$, $\min(S_2^a, S_3^a) = S_2^a$

$$D(S_1,S_2) \overset{?}{\le} D(S_1,S_3) + D(S_3,S_2)$$

By substituting the above values in this last relation we get;

$$S_1^a + S_2^a - 2S_1^a \overset{?}{\le} S_1^a + S_3^a - 2S_1^a + S_3^a + S_2^a - 2S_2^a \Rightarrow$$

$0 \overset{?}{\le} 2S_3^a - 2S_2^a$. This is valid according to the stipulation of this configuration. The proofs of cases 2), 3), 4), 5) and 6) are similar to that of case 1).

From 1)-6) we conclude that the lemma is valid for $n = 1$

ii) Inductive step: Let's assume that the lemma holds for $n-1$, where $n \ge 2$ and we will prove it for n. Since the lemma holds for $n-1$ then: $D(S_1,S_2) \le D(S_1,S_3) + D(S_3,S_2)$ (A2)

Where: $$D(S_1,S_2) = |S_1| + |S_2| - 2\sum_{i=1}^{n-1} \min(f_i^{(S_1)}, f_i^{(S_2)})$$

$$D(S_1,S_3) = |S_1| + |S_3| - 2\sum_{i=1}^{n-1} \min(f_i^{(S_1)}, f_i^{(S_3)})$$

$$D(S_3,S_2) = |S_3| + |S_2| - 2\sum_{i=1}^{n-1} \min(f_i^{(S_3)}, f_i^{(S_2)})$$

When a new character is added the strings represented by $n-1$ characters become represented by n characters.

Let the frequency of the newly introduced character be $f_n^{(S_1)}, f_n^{(S_2)}, f_n^{(S_3)}$ in S_1, S_2, S_3 respectively.

We have six configurations of the newly added character;

7) $f_n^{(S_1)} \le f_n^{(S_2)} \le f_n^{(S_3)}$,8) $f_n^{(S_1)} \le f_n^{(S_3)} \le f_n^{(S_2)}$

9) $f_n^{(S_2)} \le f_n^{(S_1)} \le f_n^{(S_3)}$,10) $f_n^{(S_2)} \le f_n^{(S_3)} \le f_n^{(S_1)}$

11) $f_n^{(S_3)} \le f_n^{(S_1)} \le f_n^{(S_2)}$,12) $f_n^{(S_3)} \le f_n^{(S_2)} \le f_n^{(S_1)}$

We will prove that relation (A1) holds in these six configurations.

7) $f_n^{(S_1)} \le f_n^{(S_2)} \le f_n^{(S_3)}$.

In this case we have:

$\min(f_n^{(S_1)}, f_n^{(S_2)}) = f_n^{(S_1)}$, $\min(f_n^{(S_1)}, f_n^{(S_3)}) = f_n^{(S_1)}$

$\min(f_n^{(S_2)}, f_n^{(S_3)}) = f_n^{(S_2)}$

$$D(S_1,S_2) \overset{?}{\le} D(S_1,S_3) + D(S_3,S_2) \Rightarrow$$

$$|S_1| + |S_2| - 2\sum_{i=1}^{n-1} \min(f_i^{(S_1)}, f_i^{(S_2)}) + f_n^{(S_1)} + f_n^{(S_2)} - 2 f_n^{(S_1)} \overset{?}{\le}$$

$$|S_1| + |S_3| - 2\sum_{i=1}^{n-1} \min(f_i^{(S_1)}, f_i^{(S_3)}) + f_n^{(S_1)} + f_n^{(S_3)} - 2 f_n^{(S_1)} +$$

$$|S_3| + |S_2| - 2\sum_{i=1}^{n-1} \min(f_i^{(S_3)}, f_i^{(S_2)}) + f_n^{(S_3)} + f_n^{(S_2)} - 2 f_n^{(S_2)}$$

$$\Rightarrow$$

$$|S_1| + |S_2| - 2\sum_{i=1}^{n-1} \min(f_i^{(S_1)}, f_i^{(S_2)}) \overset{?}{\le}$$

$$|S_1| + |S_3| - 2\sum_{i=1}^{n-1} \min(f_i^{(S_1)}, f_i^{(S_3)}) +$$

$$|S_3| + |S_2| - 2\sum_{i=1}^{n-1} \min(f_i^{(S_3)}, f_i^{(S_2)}) + 2 f_n^{(S_3)} - 2 f_n^{(S_2)}$$

Taking (A2) into account, we get: $0 \overset{?}{\le} 2 f_n^{(S_3)} - 2 f_n^{(S_2)}$, which is valid according to the stipulation of this configuration.

The proofs of cases 8), 9), 10), 11) and 12) are similar to that of case 7).

From 7)-12) we conclude that the lemma is valid for n .

From i) and ii), the lemma holds.