2015 IEEE 14th International Conference on Machine Learning and Applications

A Study of the Use of Complexity Measures in the Similarity Search Process Adopted by *k*NN Algorithm for Time Series Prediction

Antonio Rafael Sabino Parmezan, Gustavo E. A. P. A. Batista
Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo
São Carlos, SP, Brazil
{antoniop, gbatista}@icmc.usp.br

***Abstract*—In the last two decades, with the rise of the Data Mining process, there is an increasing interest in the adaptation of Machine Learning methods to support Time Series non-parametric modeling and prediction. The non-parametric temporal data modeling can be performed according to local and global approaches. The most of the local prediction data strategies are based on the *k*-Nearest Neighbor (*k*NN) learning method. In this paper we propose a modification of the *k*NN algorithm for Time Series prediction. Our proposal differs from the literature by incorporating three techniques for obtaining amplitude and offset invariance, complexity invariance, and treatment of trivial matches. We evaluate the proposed method with six complexity measures, in order to verify the impact of these measures in the projection of the future values. Besides, we face our method with two Machine Learning regression algorithms. The experimental comparisons were performed using 55 data sets, which are available at the ICMC-USP Time Series Prediction Repository. Our results indicate that the developed method is competitive and the use of a complexity-invariant distance measure generally improves the predictive performance.**

Keywords*—*Time Series Prediction, Similarity-Based Methods, Machine Learning, Data Mining.

I. Introduction

In the last two decades, with the rise of the Data Mining (DM) process, there is an increasing interest in the adaptation of Machine Learning (ML) methods, especially those for regression tasks, to support Time Series (TS) non-parametric modeling and prediction [1]. Due to their simplicity and comprehensibility, the ML methods have established themselves as serious candidates to the classical parametric models, which are based on autoregression and moving averages [2].

The non-parametric temporal data modeling does not presuppose the nature of the data distribution and can be performed according to the local and global approaches [3]. In the global approach, the predictive models are constructed from a training procedure that takes as input all observations of the series. Differently, in the local approach, the original series are partitioned into subsequences whose closest or most important values about the current value are considered to predict future observations.

In this context, one of the local prediction data strategies consists of a modification of the *k*-Nearest Neighbor (*k*NN) method. The *k*NN is an instance-based ML algorithm that consists on finding, according to some similarity measure, the k examples that are the nearest to an unlabeled example. The new example classification is decided on the labels of those k nearest examples [4].

In this article we propose a novel modification of the *k*NN algorithm for TS prediction, named the *k*NN - Time Series Prediction with Invariances (*k*NN-TSPI). Our proposal differs from the literature by incorporating three techniques for obtaining amplitude and offset invariance [5], complexity invariance [6], and treatment of trivial matches [7]. As we discuss with more details throughout this paper, these three modifications allow a more meaningful matching between the reference query and the TS subsequences.

This study has three major contributions:

- We describe for the first time our similarity-based method for TS prediction (*k*NN-TSPI) and discuss the relevance of using the invariances to complexity, offset and amplitude, as well as the elimination of trivial matches;
- As complexity can be measured according to different paradigms, we evaluate six complexity measures for TS. The investigated complexity measures use concepts from information theory, Kolmogorov complexity and chaos complexity;
- We perform one of the most comprehensible empirical evaluations ever done for TS prediction. We employ a set of 55 TS from diverse areas. We built an online archive so that other researchers can replicate our results and evaluate their own methods. We named our archive ICMC-USP Time Series Prediction Repository (ICMC-USP TSPR) [8].

Our results show that using complexity invariance generally improves the similarity-based TS prediction. From the complexities measures compared, all but one outperformed the prediction without complexity invariance.

Additionally, we face the *k*NN-TSPI with the best performing complexity invariance to the two well-known ML regression algorithms: Multilayer Perceptron (MLP) and Support Vector Machines (SVM). The results indicate that the proposed method outperforms MLP but is outperformed by SVM, with to the statistical difference.

In general, we believe the *k*NN-TSPI performs well considering its simplicity and the reduced number of parameters. The

978-1-5090-0287-0/15 $31.00 © 2015 IEEE
DOI 10.1109/ICMLA.2015.217

two parameters necessary are the window size and quantity of the nearest neighbors. However, both parameters can be easily estimated using seasonality as a reference.

We are also highlighting the importance of evaluation in a set of publicly available TS from the ICMC-USP TSPR. The data in this repository pose some of the problems that one usually encounters in a typical one or multi-step ahead prediction tasks such as the growing trend, non-stationarity, outliers and multiple overlying seasonalities. In addition, the method used for this experimental comparison is based on the guidelines advocated in related literature.

The remaining of this article is organized as follows: in Section II, we briefly describe the fundamentals on TS prediction and related work. In Section III, we present our similarity-base Time Series prediction method. In Section IV, we introduce concepts about the complexity-invariant distance measure. In Section V, we specify the configuration of the experiments, as well as considered data sets. Results and discussion are shown in Section VI while in Section VII we present the conclusions and directions for future work.

II. Background and Related Work

The methods for TS prediction are essentially based on the idea that historical data include intrinsic patterns, usually difficult to identify and not always directly interpretable, which discovered may help in the future description of the phenomenon investigated. This description is one of the main goals of the TS processing, it aims to answer in which circumstances the patterns found will repeat and what types of changes they may suffer over time [9].

The design of a model for TS prediction focuses on the application of algorithms which perform assumptions about the data, in order to capture the variables involved and modeling the existing dynamic relations, summarizing them in a robust and potentially flexible mathematical structure. The structure, besides helping to understand the process that originated the data, can also be used to predict future data. This prediction is obtained from extrapolating the generated model for a future time, so that the new data are projected for the later period to series of values used for model setting.

Recently, a few studies showed that the similarity-based methods can be useful to predict highly nonlinear and complex TS patterns. The potential of this strategy was also observed in other equally important tasks. For example, in classification, *k*NN algorithm provides results that are very difficult to beat [10]; in clustering, recent work suggests that for the TS clustering, the choice of the clustering algorithm is much less important than the choice of the distance measure used, with Dynamic Time Warping (DTW) providing excellent results [11]; in anomaly detection, a survey has shown that similarity-based methods provide the best overall results [12]. We believe that the superiority of the similarity-based methods is largely due to the community constant work on distance invariances such as warping, baseline, occlusion and rotation [6].

The general idea of the similarity-based prediction is very intuitive. Given a TS $Z = (z_1, z_2, \ldots, z_m)$ where $z_t \in \Re$, the problem is to predict the value z_{m+h}, where h indicates the prediction horizon. In practical terms, the prediction of time z_{m+h} is typically denoted by $\hat{z}(m,h)$ or $\hat{z}(h)$. For simplicity, but without loss of generality, the idea will be discussed in a unitary horizon ($h = 1$), *i.e.*, considering the prediction of the next value in the series.

As mentioned, from the series Z, the objective is to predict the next (unobserved) data point $\hat{z}(m,1)$. The simplest method uses the TS last l observations as query Q, and searches for the k most similar subsequences to Q, using a sliding window of size l. Let $S^{(1)}_{1..l}, \ldots, S^{(k)}_{1..l}$ be the k most similar subsequences, we use the next observations of each subsequence $S^{(j)}_{l+1}$ with $1 \leq j \leq k$ to predict $\hat{z}(m,1)$. Thus, the values of $S^{(j)}_{l+1}$ are provided as input to a prediction function f, for example the average (Equation 1), which aims to approximate the value of $\hat{z}(m,1)$.

$$f(S) = \frac{1}{k} \sum_{j=1}^{k} S^{(j)}_{l+1} \tag{1}$$

In Equation 1, f matches prediction function, S denotes the set of the most similar subsequences and $S^{(j)}$ refers to j^{th} nearest neighbor. This is the simplest way of combining the predictions, since the average of the predictions considers that all prediction values are equally probable to occur in the future.

Figure 1 displays an example of applying the aforementioned method with $k = 3$ and $l = 25$.

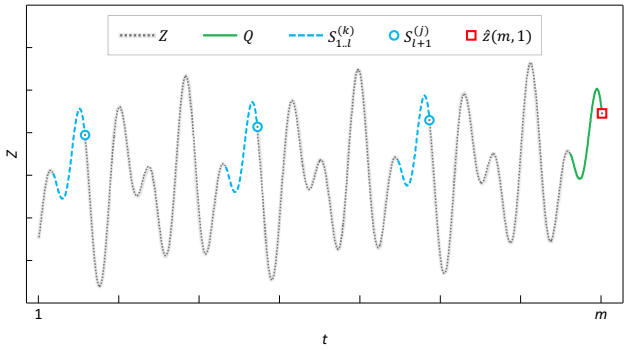

Fig. 1. An application example of the similarity-based TS prediction method with parameters $k = 3$ and $l = 25$

In this plot, the dotted line in gray represent the observations that belong to the TS; the green line indicates the subsequence of the length 25 taken as reference query; the blue dotted lines express the most similar subsequences found by the method using some measure of similarity, in this case the Euclidean Distance (ED); the blue circles correspond to the observations used for making the prediction; and the red square reflects the value to be predicted.

The similarity-based method defines an observation on the basis of the previous l observations. Thus, the dependence is restricted to a limited number of previous observations, since usually a certain value is not influenced by observations that happened a long time ago.

Several surveys were conducted to analyze the performance of the presented method with different prediction functions [13] and various distance measures [14]. In this context,

it is important to note that one of the most frequent choices of similarity measure is the L^p space, being the most used ED. Although, similarity-based TS prediction methods have been researched in the recent past, we believe that previous research has failed to identify the correct invariances required for this task. As we shall demonstrate in Section III, the right combination of amplitude, offset and recently-proposed complexity invariance, combined with a policy to avoid trivial matches, leads to more precise and meaningful predictions.

III. The kNN-TSPI Algorithm

There are three major issues with the naïve similarity search procedure shown in Figure 1, and the combination of these issues shows that the k's most similar subsequences to a query Q are frequently very different from Q.

The first issue is the lack of invariance to amplitude and offset. When comparing the query Q to a subsequence S, both should be made invariant to the amplitude and offset. This invariance can be obtained by several means, but a simple way is through z-normalization, which is defined by Equation 2.

$$z'_t = \frac{z_t - \mu}{\sigma} \tag{2}$$

In Equation 2, z'_t and z_t refer respectively to the normalized value and the observation of the TS Z, both at time t. Likewise, μ indicates the average and σ the standard deviation of the values of a given subsequence that includes the observation z_t. The z-normalization which transforms the data to ensure zero average and unit standard deviation, has been strongly advocated in tasks that need to search for TS subsequences [5].

Figure 2 illustrates a subsequence $S^{(1)}$ that is the exact match to the query Q. The subsequence $S^{(1)}$ has the exact same values as Q, with just one difference, a small offset increase. We also included a second subsequence $S^{(2)}$ that is completely different from Q. In our example, $S^{(2)}$ is a straight line in order to make our argument stronger, but also for reasons that will become clearer when we discuss the complexity invariance. According to ED, the most similar subsequence to Q is $S^{(2)}$ (ED$(Q, S^{(2)}) = 84.26$ e ED$(Q, S^{(1)}) = 114.54$), even though the offset difference between Q and $S^{(1)}$ is just 10 units. The reason is very simple: Small offset differences quickly accumulate making the final ED grow very fast.

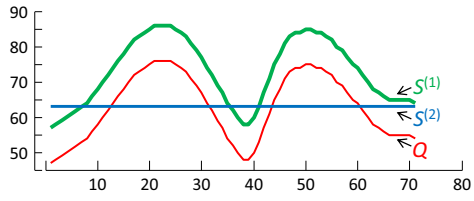

Fig. 2. Small offset difference makes ED consider Q more similar to $S^{(2)}$ than to $S^{(1)}$

Although for some applications, the amplitude and the offset may constitute a relevant feature to characterize the subsequence, in the majority of the application domains searching without invariance to offset and amplitude does not lead to meaningful results. That happens because the similar subsequences hardly occur in the exact same offset. We should notice that even small offset differences are enough to cause incorrect matches. In the case of the example in Figure 2, such offset difference between Q and $S^{(1)}$ is just 10 units. However, this is a best-case scenario, since $S^{(1)}$ is a perfect copy of Q. If we introduce noise, amplitude differences and warping, the offset difference for a mismatch will be much smaller. You should also notice that we used ED to illustrate this problem. However, other popular TS distances, such as DTW, would result in the exact same ordering for this example.

The second issue addresses the need for complexity invariance when comparing subsequences. In summary, the problem lies in the fact that pairs of complex objects, even those that subjectively may seem very similar, tend to be further apart under current distance measures than pairs of simple objects [6]. A general idea of this problem is exemplified in Figure 3, where the ED between Q and $S^{(1)}$ is greater than the distance between Q and the straight line $S^{(2)}$. The reason is that simple shapes such as $S^{(2)}$ usually present a good average behavior that match well with complex shapes. In contrast, complex shapes usually have several features such as peaks and valleys that make them difficult to match each other, even when the shapes look similar to the human eye.

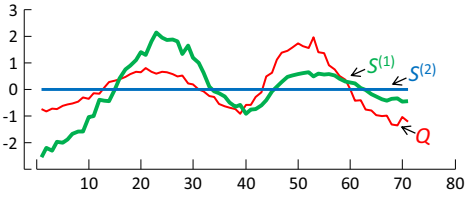

Fig. 3. The simple-shaped subsequence $S^{(2)}$ is considered the best match for Q, even though $S^{(1)}$ has a similar behavior

As we scan along the TS with a sliding window looking for a match for Q, the subsequences with huge diversity of shapes will appear and the current distance measures will tend to choose simpler shapes as the best matches. However, overly simple shapes, such as straight lines, will hardly be useful for prediction purposes. The solution to this problem lies in the use of complexity-invariant distance.

A third issue we should take care of is known as trivial matches. A subsequence taken from a sliding window that starts at observation m is very similar to the subsequence that starts at observation $m+1$ (or $m-1$). That happens because these subsequences share all but two observations. We illustrate this idea in Figure 4, in which the query Q is shifted first to the right and then to the left just by changing one of the observations.

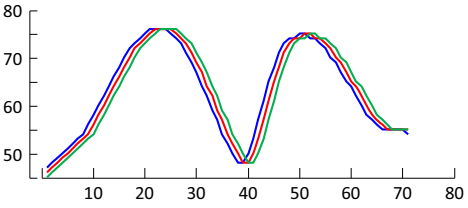

Fig. 4. The distance between the query Q and Q shifted to the right or left by one observation is very small, since these three subsequences share most of their observations

The example illustrated in Figure 4 shows that if we include Q as a part of the searchable data, the algorithm will almost always return, as most of the similar subsequences, trivial matches constituted by the query shifted by few observations. In most cases, search of the whole data is a waste of the time, since the results are very likely to be predefined.

However, this is not the only problem with the trivial matches. A similar problem occurs when the query matches any subsequence S of the TS. It is very likely that the distance between Q and S will be very similar to the distance between Q and S shifted to the left or to the right by few observations. Therefore, if S is one of the most similar subsequences to Q then the trivial matches of the S are also likely to appear among the most similar subsequences. This is a problem because the idea behind using k instead of the one most similar subsequence is to include some diversity and, therefore, to be more robust against an erroneous choice of the one similar subsequence. However, trivial matches are giving us exactly the opposite, *i.e.*, little diversity since we have several copies of the same subsequence with some small variation. One way to ensure such diversity is through exclusion of trivial matches by iterative checking.

We use the described techniques to improve the similarity-based TS prediction and create the Algorithm 1: *k*NN-TSPI.

Algorithm 1: *k*NN-TSPI

```
   /* Z represents a TS with m observations              */
   /* l is the query length in number of observations   */
   /* k indicates the number of similar subsequences    */
   Input: Z, l, k
   Output: ẑ(m, 1)
1  begin
       // S^(j)_{1..l} contains a subsequence of length l which
       // begins in observation j of the TS
2      S ← generate_subsequences(Z, l);
       // S^(j)' is the z-normalization of subsequence S^(j)
3      S' ← z_scores(S);
       // Obtaining the normalized query Q
4      Q ← S'_{(m−l+1)..m};
       // D^(j) contains the complexity-invariant distance
       // between Q and S^(j)', for 1 ≤ j ≤ m-l+1
5      D ← CID(Q, S');
       // Choosing of the k most similar subsequences
6      P ← search_nearest_neighbors(S', D, k);
       // Obtaining the next value of each of the k most
       // similar subsequences ∈ P, where S^(k)'_{l+1} indicates
       // the next z-normalized value
7      R' ← {S^(1)'_{l+1}, ..., S^(k)'_{l+1}};
       // Mapping of z-normalization to query values and
       // calculation of prediction
8      ẑ(m, 1) ← f(R);
9      return ẑ(m, 1);
10 end
```

In the 2nd line of Algorithm 1, all subsequences of the length l extracted from a TS Z are assigned to the variable S. Then, in the 3rd line, the z-normalization of all subsequences generated from the original series occurs. In the 4th line, the z-normalized subsequence Q is stored. It is important to note that the z-normalization is performed independently for each subsequence and also independently for the query. Therefore, the similarity search is made with invariance to amplitude and offset. In the 5th line, distances with complexity invariance between the normalized query Q and all subsequences of length l already z-normalized are calculated. From this, in the 6th line, the search for the k most similar subsequences occurs. These similar subsequences have no trivial matches, *i.e.* there is no overlapping with the reference query Q or between each other. Subsequently, in the 7th line, the z-normalized values of each k most similar subsequences are obtained. In the 8th line these values are mapped to the query values space according to the Equation 3 and used by the prediction function f (Equation 1) for calculating the future value.

$$S_{l+1}^{(k)} = \sigma(Q) \times S_{l+1}^{(k)'} + \mu(Q) \tag{3}$$

Additionally, the parameters k and l are very intuitive and easy to determine. For example, the value of l could be proportional to the seasonal pattern of the TS, since the nearest neighbors would be more significant in predicting. Furthermore, to adjust these parameters to a specific TS, it could be employed a training-testing validation method.

The time complexity of *k*NN-TSPI algorithm is $O(m \cdot l)$, where m is the size of TS, and l is the length of the subsequences.

IV. Complexity-Invariant Distance

Complexity invariance uses information about complexity differences between two TS as a correction factor for existing distance measures. In practical terms, the CID measure may be defined from the ED according to Equation 4 [6].

$$CID(Q, C) = ED(Q, C) \times CF(Q, C) \tag{4}$$

In this equation, Q and C are two data sequences, ED comprises the Euclidean Distance, and CF is a complex correction factor defined by Equation 5, in which $CE(Z)$ reflects an estimate of the complexity of the TS Z.

$$CF(Q, C) = \frac{\max\left(CE(Q), CE(C)\right)}{\min\left(CE(Q), CE(C)\right)} \tag{5}$$

The original CID measure uses a fairly simple complexity estimation. It is based on the physical intuition that if we could "stretch" a TS until it becomes a straight line, a complex TS would result in a longer line than a simple TS. With this, we can assign greater distances to subsequences with different complexities. The complexity estimate can be computed using Equation 6.

$$CE(Q) = \sqrt{\sum_{i=1}^{n-1} (q_i - q_{i+1})^2} \tag{6}$$

There are dozens of complexity measures that are applicable to TS data. Most of these measures are variations of concepts from information theory (especially entropy), Kolmogorov complexity and chaos complexity. In this paper, we compare the complexity estimate measure used in original CID with five other complexity measures for TS. Below you can find a short description of each investigated measure.

Absolute Difference: This measure is similar to the one used in CID. However, we compute the absolute differences

between consecutive observations, instead of the squared difference. More formally, $CE(Q) = \sum |q_i - q_{i+1}|$;

Compression: This measure approximates the Kolmogorov complexity with the LempelZiv compression length. Each TS is first converted to symbols using SAX [15] and later compressed using a file compression utility. The complexity estimate of the TS is simply the compressed file size in bytes;

Edges: This measure uses the number of edges, that can also be interpreted as the number of trend changes or the number of times the first derivative changes sign, as a complexity estimate;

Zero-crossings: The number of zero-crossings is the number of times the signal crosses an imaginary zero line, *i.e.*, the signal changes sign. In the area of speech analysis it is frequently used to detect voiced segments apart from unvoiced sounds and noisy breaks [16];

Permutation Entropy: This complexity is calculated as the entropy of a set of patterns [17]. Those patterns are obtained by generating all permutations of natural numbers between 0 and $n-1$, n being a parameter value usually chosen in the interval [3, 7]. For instance, for $n = 3$, the valid permutations are {[0, 1, 2], [1, 0, 2], ..., [2, 1, 0]}. The pattern [0, 1, 2] should be interpreted as a sequence of three observations in a TS in which the second observation is greater than the first one, and the third observation is greater then the second one. The pattern [1, 0, 2] should be interpreted as a sequence of three values where the second observation is smaller than the first one and the third observation is greater than the first one; and so on. The probability of each pattern is obtained by running a sliding window of size n across the TS and counting the occurrences of each pattern. The complexity estimate is the entropy of the set of patterns.

All complexity estimates described can be used to compare short subsequences and therefore applied to the prediction task. It is important that this fact is understood because the compared subsequences in the similarity-based prediction are usually small, varying according to the number of observations that represent a seasonal station in TS.

V. Empirical Evaluation

We applied our similarity-based TS prediction method considering different complexity estimate measures, in order to verify the impact of these measures in the projection of future values. The experiments were organized in three steps, as outlined in Figure 5.

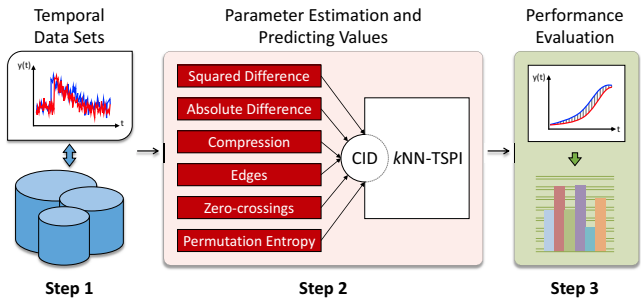

Fig. 5. Experimental setup

In Step 1, we have selected 55 benchmark data sets currently available at ICMC-USP TSPR [8], from the systematic reviews' results out of the publish works in TS prediction area. These data sets are frequently reported in the literature and come from different domains, including agriculture, engineering, finance, medicine, physics and tourism. A summary on the characteristics of the 55 data sets is shown in Table I. In this table, for each set, are described the type of data acquisition, the size of TS (m), the maximum number of observations that make up a seasonal variation (max_p) in the series, and the prediction horizon (h).

In Step 2, the measures specified in Section IV were adopted as an estimate of the complexity in CID used by *k*NN-TSPI. To make a fair comparison, the parameters of the algorithm were determined through a method analogous to the holdout validation procedure. In other words, the parameters are chosen in a training-testing process by minimizing the Mean Squared Error (MSE). We use values for k in the range of 1 to 9 in increments of 2, and l in the range of 3 to max_p also in increments of 2. As max_p is an upper bound for number of observations that correspond to a seasonal station, l will be proportional to the seasonality in the TS.

In Step 3, the results obtained were evaluated according to the predictive error calculated by using the Mean Absolute Percentage Error (MAPE), Theil's U (TU), and Prediction Of Change In Direction (POCID). This measures were calculated from prediction (one-step ahead with updating) the series.

The MAPE measure is defined according to Equation 7, where z_t is the actual value observed and $\hat{z}_t$ is the predicted value. The result of this measure is a percentage value that relates the predicted value with the actual value of the TS.

$$MAPE = \frac{1}{h}\sum_{t=1}^{h}\left|\frac{z_t - \hat{z}_t}{z_t}\right| \times 100 \tag{7}$$

The TU coefficient, defined by Equation 8, is based on the MSE of the predictor, normalized by the prediction error of a naïve model. The naïve model assumes that the best value for time $t+1$ is the value obtained at time t.

$$TU = \frac{\sum_{t=1}^{h}(z_t - \hat{z}_t)^2}{\sum_{t=1}^{h}(z_t - z_{t-1})^2} \tag{8}$$

According to Equation 8, if TU $>$ 1, the naïve model outperformed the investigated algorithm; if TU $<$ 1, the algorithm outperformed the naïve model; and if TU $\leq$ 0.55, the algorithm is trusted to carry out future predictions.

The last considered evaluation measure was the POCID, which is expressed by the Equation 9. In this equation, the term D_t has the value 1 if $(\hat{z}_t - \hat{z}_{t-1})(z_t - z_{t-1}) > 0$, and is 0 otherwise.

$$POCID = \frac{\sum_{t=1}^{h} D_t}{h} \times 100 \tag{9}$$

From the values of these evaluation measures it was also possible to objectively compare different settings of

TABLE I. SUMMARY OF FEATURES AND SETTINGS OF THE BENCHMARK DATA SETS

ID	Data Set	Acquisition	*m*	*max_p*	*h*	ID	Data Set	Acquisition	*m*	*max_p*	*h*
01.A	Fortaleza	Annual	149	6	7	29.M	Darwin	Monthly	1400	12	36
02.A	Manchas	Annual	176	11	12	30.M	Dow Jones	Monthly	641	12	29
03.D	Atmosfera: Temperatura	Daily	365	7	31	31.M	Energia	Monthly	141	12	9
04.D	Atmosfera: Umidade Relativa do Ar	Daily	365	7	31	32.M	Global	Monthly	1800	12	36
05.D	Banespa	Daily	1499	7	88	33.M	ICV	Monthly	126	12	6
06.D	CEMIG	Daily	1499	7	88	34.M	IPI	Monthly	187	12	7
07.D	IBV	Daily	1499	7	88	35.M	Latex	Monthly	199	12	7
08.D	Patient Demand	Daily	821	7	90	36.M	Lavras	Monthly	384	12	12
09.D	Petrobras	Daily	1499	7	88	37.M	Maine	Monthly	128	12	8
10.D	Poluição: PM10	Daily	365	7	31	38.M	MPrime	Monthly	707	12	23
11.D	Poluição: SO2	Daily	365	7	31	39.M	OSVisit	Monthly	228	12	12
12.D	Poluição: CO	Daily	365	7	31	40.M	Ozônio	Monthly	180	12	12
13.D	Poluição: O3	Daily	365	7	31	41.M	PFI	Monthly	115	12	7
14.D	Poluição: NO2	Daily	365	7	31	42.M	Reservoir	Monthly	864	12	24
15.D	Star	Daily	600	7	25	43.M	STemp	Monthly	1896	12	36
16.D	Stock Market: Amsterdam	Daily	3128	7	92	44.M	Temperatura: Cananéia	Monthly	120	12	12
17.D	Stock Market: Frankfurt	Daily	3128	7	92	45.M	Temperatura: Ubatuba	Monthly	120	12	12
18.D	Stock Market: London	Daily	3128	7	92	46.M	USA	Monthly	130	12	6
19.D	Stock Market: Hong Kong	Daily	3128	7	92	47.M	Wine: Fortified White	Monthly	187	12	19
20.D	Stock Market: Japan	Daily	3128	7	92	48.M	Wine: Dry White	Monthly	187	12	19
21.D	Stock Market: Singapore	Daily	3128	7	92	49.M	Wine: Sweet White	Monthly	187	12	19
22.M	Stock Market: New York	Daily	3128	7	92	50.M	Wine: Red	Monthly	187	12	19
23.M	Bebida	Monthly	187	12	7	51.M	Wine: Rose	Monthly	187	12	19
24.M	CBE: Chocolate	Monthly	396	12	24	52.M	Wine: Sparkling	Monthly	187	12	19
25.M	CBE: Beer	Monthly	396	12	24	53.M	ECG: A	0.5s Intervals	1800	60	120
26.M	CBE: Electricity Production	Monthly	396	12	24	54.M	ECG: B	0.5s Intervals	1800	60	120
27.M	Chicken	Monthly	187	12	7	55.M	Laser	1s Intervals	1000	8	100
28.M	Consumo	Monthly	154	12	10						

*k*NN-TSPI algorithm. Such comparisons were analyzed using the Friedman's non-parametric statistical test for paired data and multiple comparisons, with significance level of 5% (p-value < 0.05), followed by Nemenyi post-hoc test[1].

The algorithms employed in this research, including the method for parameter estimation, were implemented in MATLAB[2], which has considerable compatibility with the free and open source GNU Octave[3].

VI. RESULTS AND DISCUSSION

In addition to the estimated Squared Difference used by original CID, in this study were researched another five different complexity estimates: Absolute Difference, Compression, Edges, Zero-crossings and Permutation Entropy. These measures were applied to CID, in order to verify the impact of these combinations in the similarity search process adopted by *k*NN-TSPI. We have also compared these estimates with the same strategy, but without complexity invariance, *i.e.* using the Euclidean Distance.

Figure 6 presents the critical distance plot [18] for MAPE of the investigated measures. According to the scale represented in this figure, which indicates the average rank of each measure, the Squared Difference showed the best result, but not statistically better than the other complexity measures. Interestingly, the estimated Zero-crossings were competitive with the Squared Difference, although the latter is conceptually simpler. We have also observed that the estimated Edges provided, on average, the worst result. This means *a priori* that this complexity measure has little discriminatory power in the prediction task.

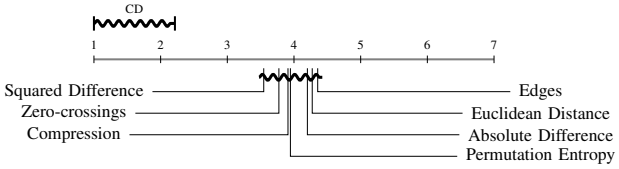

Fig. 6. Nemenyi post-hoc test result. Measures connected by a thick line have not presented statistically significant difference

For sake of space, we do not show the critical distance plots for TU and POCID measures. However, we created a webpage[4] that contains detailed results of our experiments. In this article, we presented some tables that allow greater view of results. Table II shows the average and standard deviation of the values obtained with the application of POCID measure, as well as the quantity of TU values smaller than 1 and less than or equal to 0.55.

TABLE II. SUMMARY RESULTS OF THE COMPLEXITY-INVARIANT DISTANCE MEASURES

Complexity Measure	POCID (%)	TU $<$ 1	TU $\leq$ 0.55
Euclidean Distance	57.44(18.62)	33	8
Squared Difference	59.11(18.34)	38	8
Absolute Difference	57.45(17.16)	38	10
Compression	58.61(18.06)	38	8
Edges	57.05(18.05)	34	5
Zero-crossings	58.30(17.06)	38	8
Permutation Entropy	57.92(17.24)	40	6

It is noted in Table II that the average and standard deviation values of POCID are distributed uniformly between the seven measures. This fact demonstrates that the use of any of these measures by the *k*NN-TSPI entailed in an average hit rate of 57.98% about the trend of the future values. Considering

[1]Statistical tests performed using KEEL Software Tool version for Windows, http://www.keel.es.

[2]http://www.mathworks.com.

[3]http://www.gnu.org/software/octave.

[4]http://sites.labic.icmc.usp.br/icmc_tspr/2015_icmla.

all complexity measures, Squared Difference presented the best result, scoring a 59.11% hit rate on the trends. According to the TU values, the Permutation Entropy performed consistently better than the trivial or naïve predictor for 40 data sets when TU $<$ 1. Looking at TU $\leq$ 0.55, the best performing measure was Absolute Difference, which presented significant results in 10 of the 55 data sets.

In order to demonstrate that the *k*NN-TSPI can provide as precise results as other ML algorithms for the same task, in this study we tested the proposed method using the original CID with two regression methods widespread in literature: MLP and SVM. These two algorithms were applied according to the global approach and their parameters were adjusted using 10-fold cross validation. In Figure 7, the critical distance plot built using the MAPE values resulting from the aforesaid experiment it is presented.

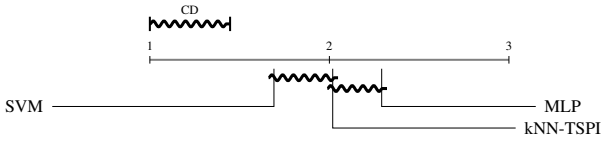

Fig. 7. Nemenyi post-hoc test result. Algorithms connected by a thick line have not presented statistically significant difference

As it can be seen in Figure 7, the *k*NN-TSPI algorithm showed the second best result, not presenting statistically significant difference when compared with the other methods. Although the SVM algorithm has had a prediction performance slightly higher than the *k*NN-TSPI, the method proposed in this paper is considerably simpler to adjust. While SVM has three parameters to be estimated, the similarity-based method has only two. Most importantly, these two parameters are totally intuitive and can be easily estimated just observing the seasonality of the data.

It is relevant to mention that the information summarized in Figure 7 also reflect the performance of the algorithms calculated by TU coefficient. In relation to the critical distance plot for POCID, the *k*NN-TSPI presented the best results.

VII. Conclusion

In this article, we present a similarity-based TS prediction method that uses of three techniques for obtaining offset and amplitude invariance, complexity invariance, and treatment of trivial matches. Our method was applied on 55 benchmark data sets and it was configured using six different complexity measures for obtaining invariance to this distortion. The results show that the use of CID with Squared Difference can be a good option, as seen in other tasks such as classification.

We have compared our method with two learning algorithms for regression: MLP and SVM. Our method outperformed the MLP algorithm and did not present statistically significant difference in respect to SVM. Nevertheless, the similarity-based method with invariances is considerably more intuitive and simpler to adjust.

In our future studies, we intend to explore the properties of the proposed method, among which are similarity measures and prediction functions. Additionally, we would like to compare the kNN-TSPI with state-of-the-art methods, for example Seasonal Autoregressive Integrated Moving Average (SARIMA).

VIII. Acknowledgments

This research was supported by FAPESP, grant 2013/109-78-8, and CNPq, grants 303083/2013-1 and 446330/2014-0.

References

[1] G. Ristanoski, W. Liu, and J. Bailey, "A time-dependent enhanced support vector machine for time series regression," in *International Conference on Knowledge Discovery and Data Mining*. New York: ACM, 2013, pp. 946–954.

[2] J. G. D. Gooijer and R. J. Hyndman, "25 years of time series forecasting," *Int. J. Forecasting*, vol. 22, no. 3, pp. 443–473, 2006.

[3] M. N. Islam and B. Sivakumar, "Characterization and prediction of runoff dynamics: A nonlinear dynamical view," *Advances in Water Resources*, vol. 25, no. 2, pp. 179–190, 2002.

[4] E. Fix and J. L. Hodges, "Discriminatory analysis, nonparametric discrimination, consistency properties," US Air Force School of Aerospace Medicine, Randolph Field, Tech. Rep. 4, Project 21-49-004, 1951.

[5] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh, "Searching and mining trillions of time series subsequences under dynamic time warping," in *International Conference on Knowledge Discovery and Data Mining*, Beijing, 2012, pp. 262–270.

[6] G. E. A. P. A. Batista, E. J. Keogh, O. M. Tataw, and V. M. A. Souza, "CID: An efficient complexity-invariant distance for time series," *Data Mining and Knowledge Discovery*, vol. 28, no. 3, pp. 634–669, 2014.

[7] A. Mueen, E. Keogh, Q. Zhu, and S. Cash, "Exact discovery of time series motifs," in *SIAM International Conference on Data Mining*, 2009.

[8] A. R. S. Parmezan and G. E. A. P. A. Batista, "ICMC-USP time series prediction repository," 2014, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos. Available at http://sites.labic.icmc.usp.br/icmc_tspr.

[9] C. Chatfield, *The analysis of time series: An introduction*. Boca Raton: Taylor & Francis, 2013.

[10] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh, "Querying and mining of time series data: Experimental comparison of representations and distance measures," in *International Conference on Very Large Data Bases*, vol. 1, no. 2. Auckland: VLDB Endowment, 2008, pp. 1542–1552.

[11] Q. Zhu, G. E. A. P. A. Batista, T. Rakthanmanon, and E. J. Keogh, "A novel approximation to dynamic time warping allows anytime clustering of massive time series datasets," in *SIAM International Conference on Data Mining*, California, 2012, pp. 999–1010.

[12] V. Chandola, D. Cheboli, and V. Kumar, "Detecting anomalies in a time series database," Department of Computer Science and Engineering, University of Minnesota, Minneapolis, Tech. Rep. 09-004, 2009.

[13] C. A. Ferrero, "Algoritmo kNN para previsão de dados temporais: funções de previsão e critérios de seleção de vizinhos próximos aplicados à variáveis ambientais em limnologia," 2009, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos.

[14] J. Aikes Junior, H. D. Lee, C. A. Ferrero, and F. C. Wu, "Estudo da influência de diversas medidas de similaridade na previsão de séries temporais utilizando o algoritmo kNN-TSP," in *Encontro Nacional de Inteligência Artificial*. Curitiba: SBC, 2012, pp. 1–12.

[15] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A symbolic representation of time series, with implications for streaming algorithms," in *Workshop on Research Issues in Data Mining and Knowledge Discovery*. New York: ACM, 2003, pp. 2–11.

[16] L. Rabiner and R. Schafer, *Digital processing of speech signals*. Englewood Cliffs: Prentice Hall, 1978.

[17] C. Bandt and B. Pompe, "Permutation entropy: A natural complexity measure for time series," *Phys. Rev. Lett.*, vol. 88, pp. 1–4, 2002.

[18] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.