Class-Based Attribute Weighting for Time Series Classification

Bálint CSATÁRI[1], Zoltán PREKOPCSÁK[2]

[1,2] Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics, Magyar Tudósok körútja 2., H-1117, Budapest, Hungary

csatari@tmit.bme.hu[1], prekopcsak@tmit.bme.hu[2]

Abstract. *In this paper, we present two novel class-based weighting methods for the Euclidean nearest neighbor algorithm and compare them with global weighting methods considering empirical results on a widely accepted time series classification benchmark dataset. Our methods provide higher accuracy than every global weighting in nearly half of the cases and they have better overall performance. We conclude that class-based weighting has great potential for improving time series classification accuracy and it might be extended to use with other distance functions than the Euclidean distance.*

Keywords

Data mining, time series data, classification, attribute weighting.

1. Introduction

Attribute weighting has a long tradition in data mining. These methods are mostly used for feature selection [5] and for the alternation of distance functions in lazy classifiers [12]. In this paper, we are focusing on weighting for time series classification, because in this area, lazy classifiers are among the most accurate and robust [14], so there is a room for improvement with weighting methods. Classic distance functions like the Euclidean distance, or the Dynamic Time Warping [11] (DTW) consider every attribute the same importance, so with many noisy attributes present these distance functions fail. Weighting methods give lower weights for noisy attributes and higher for the most informative ones according to certain criteria. We call these *global weighting methods* because they use one weighting vector during the whole classification process.

In this paper, we propose two novel weighting methods for time series classification, which calculate different weighting vectors for each class, and we call them *class-based weighting methods*. The rest of the paper is structured as follows. In Section 2, we briefly review related work and mention similar research efforts, then in Section 3 we formulate the problem, describe our methods and the implementation details, while in Section 4, we present empirical results on a benchmark dataset. Finally, we conclude and outline some ideas for future work.

2. Related work

A lot of interest has been paid on time series classification in the past decade [14]. There has been numerous efforts to improve classification accuracy [2, 11] and even a data mining competition has been held [7] to compare different methods. In spite of these efforts, simple k-nearest neighbor classifier with Euclidean distance is still considered as one of the most accurate and robust general solution [14]. It has been shown that Euclidean distance outperforms many other time series distance functions when faced with datasets from diverse domains [6, 2]. One viable alternative is the Dynamic Time Warping (DTW) and its extensions [11], but it increases complexity and results in a slower algorithm than the simple Euclidean distance calculation. For distance measure comparisions, one nearest neighbor (1-NN) is an accepted objective evaluation method [6], so we use this in the empirical evaluation.

Attribute weighting has been an active research area in the 1990s, but as feature selection methods became independent of weighting and lazy classifiers have been outperformed by other methods in most areas, attribute weighting became a less mainstream area of data mining research. This explains why we have not found research papers for time series weighting. Previously, many global weighting methods have been proposed for general data mining problems [12], like the ones based on statistical and correlation tests (e.g. information gain [1]) or instance distances (e.g. Relief [5], ReliefF [9]). Furthermore, local weighting methods have also been introduced, which assign different weights for groups of instances or for each instance. Our class-based method is a special case of local weighting methods, and some papers had a similar approach for datasets with discrete attributes [4], while our work focuses on continous attributes, as time series data is usually represented this way.

3. Attribute weighting

In this section, we present our method, which utilizes attribute weighting to improve classification of time series data. Our goal is to emphasize specific ranges of sequences while suppressing others to make class-specific characteristics more obvious for the classifier. The question is what is the most appropriate way of determining the weights for this.

The main ideas behind our method for weight calculations are the so-called *internal* and *external* average distances. An attribute has high value during the classification, if it has a similar value compared with other sequences from the same class, in other words, if its internal distance is low. Meanwhile, an attribute showing huge deviation from other classes, can have a role in differentiating that given class from others. In other words we could say the external distance for that attribute is high. In order to formalize this idea, we have introduced the so-called *global internal* and *global external* average distances. These can be calculated as

$$D^a_{int} = \frac{\sum_k \sum_{i,j \in C_k} |X^a_i - X^a_j|}{\sum_k |C_k| \cdot (|C_k| - 1)} \tag{1}$$

$$D^a_{ext} = \frac{\sum_{i,j \in N} |X^a_i - X^a_j| - \sum_k \sum_{i,j \in C_k} |X^a_i - X^a_j|}{|N|^2 - \sum_k |C_k|^2} \tag{2}$$

where X^a_i is the value for the a-th attribute of the i-th sequence, $|C_k|$ is the population count of a class k and $|N|$ is the total number of sequences ($\sum_k |C_k| = |N|$).

To make use of these assumptions, with a so-called *weighting function* we calculate weights for every attribute using D_{int} and D_{ext} values. It is easy to guess that $\frac{D_{ext}}{D_{int}}$ exploits both internal coherence and external deviation, but a weighting function relying only on internal distances could be $\frac{1}{D_{int}}$. A truly positive side of this global weighting approach is that – as we will see – it can improve the error rate of an instance based 1-NN Euclidean distance comparison by extending the process with a simple preprocessing step.

However, we propose a class-based attribute weighting method which can improve the matching efficiency further. By creating a customized weighting vector for each class, the sometimes opposing influence of different classes on the global weighting vector can be eliminated. In order to create class-based weighting vectors, we have to determine the internal and external distances individually for each class. Figure 1 depicts the comparisons for this class-based method, which can be formalized as follows:

$$d^{k,a}_{int} = \frac{\sum_{i,j \in C_k} |X^a_i - X^a_j|}{|C_k| \cdot (|C_k| - 1)} \tag{3}$$

$$d^{k,a}_{ext} = \frac{\sum_{i \in C_k} \sum_{j \in N - C_k} |X^a_i - X^a_j|}{|C_k| \cdot (|N| - |C_k|)} \tag{4}$$

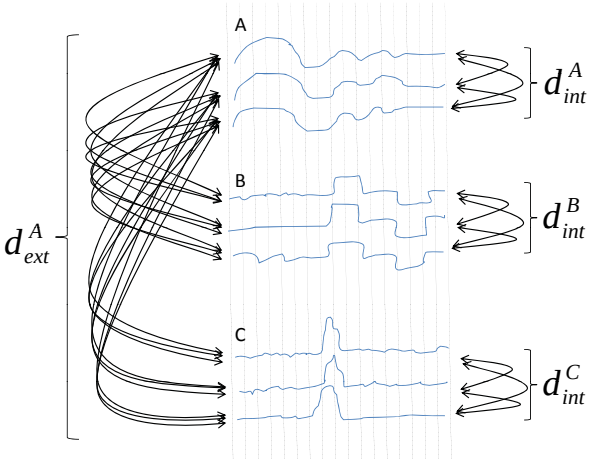

Fig. 1. Demonstration of class-based internal and external distances.

Similarly to global weighting functions, there are many possibilities of combining these distances in order to create weighting vectors. In the next section we demonstrate our benchmark results using two simple and obvious weighting functions similiar to the ones mentioned earlier, $\frac{d_{ext}}{d_{int}}$ and $\frac{1}{d_{int}}$. While in the case of global weighting, the implementation is straightforward, the utilization of class-based weighting vectors demand more than a simple preprocessing step. Here we explain the steps of our modified classification method, in which we have modified an instance based 1-NN Euclidean distance comparison in order to make the most of class-based weighting.

1. After the training set has been acquired, the sequences are *z-normalized*, so each sequence has an average of zero and a standard deviation of one.

2. The $d^{k,a}_{int}$ and $d^{k,a}_{ext}$ values are calculated based on the training set elements for each k class and each a attribute. With this, the individual weighting vectors are determined for each class (using the $\frac{d_{ext}}{d_{int}}$ or the $\frac{1}{d_{int}}$ weighting function). The length of this vector is equal to the number of attributes.

3. The weighting vectors have to be normalized so the sum of the elements is equal to the number of attributes. This step ensures that none of the elements make extreme modification to the sequence. Note that with the regular 1-NN Euclidean distance comparison the weights are 1, thus giving the same sum.

4. The training set elements are multiplied with the weighting vector corresponding to their class, so the important ranges are emphasized and the incoherent parts are omitted. After that, the processing of the test elements can commence, with the following steps performed on each incoming sequence.

(a) The sequence is z-normalized to bring the attributes to the same range as the training set instances.

(b) As the class of the sequence is unknown, it has to be multiplied with the weighting vectors independently. This data is then filled into a $\max(k) \times A$ sized T matrix where $\max(k)$ denotes the number of classes, and A denotes the number of attributes.

(c) Since the class of the train set elements are given, each train set element is compared to the corresponding row of T, using Euclidean distance measure. The lowest distance train element's class is matched with the tested sequence.

Before moving on the the next section and presenting benchmark results, we would like to emphasize that the weights calculated by our method are invariant to offsetting and constant multiplication thanks to the definition of the weighting functions. This is important, because the weighting stays the same after any type of data normalization.

4. Results

In this section we present how our attribute weighting method performs against other weighting operators on a benchmark dataset. The operators we tried and managed to compete with are built in weighting operators of Rapidminer [10] and Weka [13]: the regular 1-NN Euclidean distance, Gini-index [3], information gain ratio, information gain [1], Relief [5], ReliefF [9], SVM weighting [13], single rule weighting and OneR [13]. We omitted the results of two other operators, namely PCA weighting and standard deviation weighting, which in most of the cases provided worse results than the old but strong strawman, regular 1-NN. Based on our novel methods discussed in the previous section, we let two global, and two class-based weighting functions ($\frac{1}{D_{int}}$, $\frac{D_{ext}}{D_{int}}$, $\frac{1}{d_{int}}$, $\frac{d_{ext}}{d_{int}}$, combining the distances shown in Eq. (1)-(4)) to compete against these.

As a benchmark database, we used the 20 datasets hosted by the University of California, Riverside (UCR) [8], which includes time series datasets from many areas of life. These include human movements (Gun_point), shape recognition via contour curves converted into pseudo time series (Leaf, Faces), word image matching (50words), in-line process control measurements of processing semiconductor wafers (Wafer), cardial electrical activity of humans (ECG), different types of lightning data (Lightning-2, Lightning-7) and also synthetic datasets (Trace, Synthetic_control). Spreading out this broadly, this database is proven to be a comprehensive benchmark for time series data classification.

We listed comparison results in Table 1, which shows classification error rates for the 9 former and the 4 newly proposed classification algorithms. As we expected, the global weighting functions seem to show similar accuracy as earlier weighting methods, having an average error of 20.77% and 20.62% on the 20 datasets. However the class-based weighting resulted in an average improvement of 2-3% (error rates are 19.13% and 18.20%). Also note, that in 9 cases our algorithm provided the best results among the classification methods.

As we expected, class-based weighting outperformed global weighting methods, proving that this approach has a measurable effect on classification error. Comparing our weighting functions, we can say that $\frac{1}{d_{int}}$ seemed to be better than the others, which means that internal coherence of the classes have a huge importance in time series classification, at least on this benchmark dataset.

Attribute weighting does not raise complexity during the classification, but it needs preprocessing while storing the training set. We have to calculate distances between all training instances, so it has a quadratic complexity on the number of instances and a linear complexity on the number of attributes. The quadratic complexity can be problematic in the case of huge datasets, but in that case sampling can be used for the approximation of the average distances and the weight vectors.

5. Conclusion

We have presented new global weighting functions for time series classification and a new approach for class-based attribute weighting. Our class-based methods provided higher accuracy than global weighting on a widely accepted time series benchmark dataset. We have to note that it has a quadratic complexity, so future work might be needed on calculating approximate weights or creating a linear weighting method.

Although we presented the operation with the instance based 1-NN comparison, attribute weighting is rather a preprocessing task. This means that it can be used independently or in combination with other preprocessing methods to improve classification performance. Besides, weighting can be extended for other distance functions, like Dynamic Time Warping, so we will keep on investigating class-based weighting for various cases.

Acknowledgements

We would like to thank the University of California, Riverside (UCR), especially Professor Eamonn Keogh for making the time series benchmarking dataset publicly available. Without this, our comprehensive analysis would have not been possible.

References

[1] DAELEMANS, W., VAN DEN BOSCH, A.: Generalization performance of backpropagation learning on a syllabification task. In *Proceedings of the 3rd Twente Workshop on Language Technology.* Enschede, The Netherlands, 1992, p. 27–37.

[2] DING, H., TRAJCEVSKI, G., SCHEUERMANN, P., WANG, X., KEOGH, E.: Querying and Mining of Time Series Data: Experimental Comparison of Representations and Distance Measures. In *Proceedings of the VLDB Endowment (Vol. 1.).* Auckland, New Zealand, 2008, p. 1542–1552.

[3] GINI, C.: Measurement of Inequality of Incomes. In *The Economic Journal*, 31., 1921, p. 124–126.

[4] HOWE, N., CARDIE, C.: Examining Locally Varying Weights for Nearest Neighbor Algorithms. In *Proceedings of the Second International Conference on Case-Based Reasoning*. Berlin, Germany, 1997, p. 455–466.

[5] KIRA, K., RENDELL, L. A.: The Feature Selection Problem: Traditional Methods and a New Algorithm. In *AAAI*. Cambridge, MA, USA, 1992, p. 129–134.

[6] KEOGH, E., KASETTY, S.: On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstration. In *the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* Edmonton, Alberta, Canada, 2002, p. 102–111.

[7] KEOGH, E., SHELTON, C., MOERCHEN, F.: Workshop and Challenge on Time Series Classification at SIGKDD 2007. http://www.cs.ucr.edu/ eamonn/SIGKDD2007TimeSeries.html (retrieved 5th March, 2010)

[8] KEOGH, E., XI, X., WEI, L., RATANAMAHATANA, C.: The UCR Time Series Classification/Clustering Page. http://www.cs.ucr.edu/ẽamonn/time_series_data/ (retrieved 5th March, 2010)

[9] KONONENKO, I.: Estimating attributes: Analysis and extensions of RELIEF. In *Proceedings of the European Conference on Machine Learning*. Catania, Italy, 1994, p. 171–182.

[10] MIERSWA, I., WURST, M., KLINKENBERG, R., SCHOLCZ, M., EULER, T.: YALE: Rapid Prototyping for Complex Data Mining Tasks. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Phildaelphia, USA, 2006, p. 935–940.

[11] RATANAMAHATANA, C., KEOGH E.: Three Myths about Dynamic Time Warping. In *Proceedings of SIAM International Conference on Data Mining*. Newport Beach, CA, USA, 2005, p. 506-510.

[12] WETTSCHERECK, D., AHA, D. W., MOHRI, T.: A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. In *Artifical Intelligence Review*, Vol. 11., Issue 1-5, 1997, p. 273–314.

[13] WITTEN, I. H., FRANK, E.: *Data Mining: Practical Machine Learning Tools and Techniques*. Second Edition, Elsevier Inc., 2005.

[14] YE, L., KEOGH, E.: Time Series Shapelets: A New Primitive for Data Mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. Paris, France, 2009, p. 947–956.

About Authors...

Bálint CSATÁRI is a second year electrical engineering MSc student at Budapest University of Technology and Economics, specializing in infocommunication systems. He got involved in data mining with his project laboratory. In his free time, he's editing his faculty's newspaper.

Zoltán PREKOPCSÁK is a PhD student at Budapest University of Technology and Economics. His research topic is pattern classification in time series with applications in human-computer interaction and ubiquitous computing.

%	1-NN	GiniIndex	InfoGainRatio	InfoGain	Relief	ReliefF (Weka)	SVM weighting	SingleRuleWeighting	OneR (Weka)	$\frac{D_{ext}}{D_{int}}$	$\frac{1}{D_{int}}$	$\frac{d_{ext}}{d_{int}}$	$\frac{1}{d_{int}}$
50words	36.92	38.68	34.73	37.80	38.24	39.12	38.02	36.92	37.14	36.04	34.95	33.85	33.19
Adiac	38.87	41.43	41.18	32.74	39.64	37.34	32.99	39.64	38.87	38.11	37.85	35.55	35.04
Beef	33.33	33.33	33.33	30.00	26.67	30.00	30.00	36.67	36.67	33.33	33.33	36.67	30.00
CBF	14.78	14.67	13.00	12.56	12.56	17.00	15.33	14.33	12.67	13.44	14.00	6.56	4.67
Coffee	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	3.57	0.00	3.57
ECG200	12.00	8.00	12.00	10.00	11.00	10.00	10.00	12.00	12.00	11.00	10.00	10.00	7.00
FaceAll	28.64	19.35	24.97	19.53	20.95	20.06	21.18	28.34	25.62	25.98	31.78	27.81	26.27
FaceFour	21.59	18.18	17.05	15.91	17.05	17.05	13.64	22.73	17.05	19.32	14.77	13.64	9.09
Fish	21.71	22.86	23.43	24.57	23.43	26.29	20.00	19.43	22.29	21.14	18.29	19.43	17.14
Gun_Point	8.67	11.33	7.33	10.00	10.00	10.67	16.00	10.00	8.00	10.00	11.33	15.33	16.67
Lightning-2	24.59	26.23	22.95	26.23	26.23	26.23	21.31	14.75	21.31	26.23	24.59	26.23	27.87
Lightning-7	42.47	36.99	39.73	39.73	32.88	32.88	35.62	42.47	41.10	36.99	34.25	27.40	28.77
OliveOil	13.33	13.33	13.33	13.33	16.67	13.33	10.00	13.33	13.33	13.33	16.67	16.67	13.33
OSULeaf	47.93	45.45	45.87	48.35	49.17	47.52	47.11	46.69	44.63	47.93	45.87	48.35	46.69
SwedishLeaf	21.12	21.44	20.48	22.08	21.28	20.80	23.04	20.96	19.36	20.16	20.64	16.64	16.16
Synthetic_control	12.00	13.67	13.33	12.67	13.00	11.33	12.33	17.67	11.00	10.67	10.00	8.00	10.00
Trace	24.00	19.00	25.00	21.00	33.00	32.00	27.00	24.00	26.00	25.00	21.00	13.00	8.00
Two_Patterns	9.33	46.20	19.73	50.10	14.08	18.50	20.75	11.75	13.52	9.33	12.48	9.50	13.03
Wafer	0.45	0.97	0.68	0.58	0.57	0.44	0.57	0.65	0.63	0.44	0.45	0.58	0.54
Yoga	16.97	16.60	17.90	16.83	15.73	15.70	17.07	17.73	16.67	16.93	16.57	17.47	17.07
Average	21.44	22.39	21.30	22.20	21.11	21.31	20.60	21.50	20.89	20.77	20.62	19.13	18.20

Tab. 1. Classification error percentages of different algorithms.