Can automated smoothing significantly improve benchmark time series classification algorithms?

James Large, Paul Southam, and Anthony Bagnall

University of East Anglia, Norwich Research Park, UK
{James.Large, Paul.Southam, ajb}@uea.ac.uk
http://www.timeseriesclassification.com

Abstract. tl;dr: no, it cannot, at least not on average on the standard archive problems. We assess whether using six smoothing algorithms (moving average, exponential smoothing, Gaussian filter, Savitzky-Golay filter, Fourier approximation and a recursive median sieve) could be automatically applied to time series classification problems as a preprocessing step to improve the performance of three benchmark classifiers (1-Nearest Neighbour with Euclidean and Dynamic Time Warping distances, and Rotation Forest). We found no significant improvement over unsmoothed data even when we set the smoothing parameter through cross validation. We are not claiming smoothing has no worth. It has an important role in exploratory analysis and helps with specific classification problems where domain knowledge can be exploited. What we observe is that the automatic application does not help to improve classification performance and that we cannot explain the improvement of other time series classification algorithms over the baseline classifiers simply as a function of the absence of smoothing.

Keywords: time series, classification, smoothing, benchmark

1 Introduction

Time Series Classification (TSC) is differentiated from standard classification by the fact that the ordering of the attributes may be important in finding discriminatory features. Standard vector classifiers such as rotation forest and standard time dependent approaches such dynamic time warping with 1-NN are strong benchmark algorithms to compare against the range of bespoke TSC algorithms that have been proposed in recent years. Some of these achieve impressive performance and are significantly better than the benchmarks. Nevertheless, there has always been a suspicion that sensible standard preprocessing of the data would perhaps increase the accuracy of benchmark classifiers and that would make at least some of the bespoke algorithms redundant [1]. Broadly speaking, there are four types of preprocessing that may improve classifier performance: normalisation; smoothing; dimensionality reduction; and discretization. We address the question of whether smoothing series can significantly improve the accuracy of benchmark classifiers. Smoothing is the process of reducing the noise

in the series to make patterns in the data more apparent and is generally used as part of an exploratory analysis.

It is important to stress we are only concerned with class independent noise, since class dependent noise is possibly useful as a discriminatory feature. This is where we diverge from the majority of signal processing research into noise modeling and reduction. We are not necessarily trying to "clean up" a signal. Instead, we are trying to remove artifacts that may confound the classifier.

We test whether six smoothing algorithms improve three base classifiers. These are described in detail in Section 2. It is clearly important to set the parameters of the algorithm when smoothing so that it is relevant to a specific problem. Because we are attempting to smooth to improve classification, we set parameters through cross validation on the train data using the base classifier we are testing. The experimental design is described in Section 3. Our experiments address the following two questions.

1. Does smoothing with default parameters increase the accuracy of benchmark classifiers?
2. Can we learn smoothing parameters on the train data to significantly improve benchmark TSC algorithms?

A priori, we believed it unlikely that systematic smoothing would improve accuracy over the diverse data sets in the archive, since many of the series have very little noise. However, we thought that supervised smoothing, where no smoothing was an option, would improve performance albeit at the large computational cost of the parameter search. Our results, presented in Section 4 show that in fact smoothing makes very little difference, even when supervised. We discuss these results in Section 5 and conclude in Section 6.

2 Background

2.1 Time Series Classification

A large number of new classification problems have been proposed in the last ten years. While not exhaustive by itself, it is important to evaluate new algorithms against sensible benchmark classifiers on standard test problems in order to ascertain the usefulness of new research. The UCR archive is a widely used archive of test problems [8]. The archive is a continually growing collection of real valued TSC datasets[1] which come from a range of different domains and have a range of characteristics, in terms of size, number of classes, imbalances, etc. Most TSC publications benchmark against a 1-NN classifier using either Euclidean distance (ED) or Dynamic Time Warping (DTW) distance. DTW compensates for potential misalignments amongst series of the same class. DTW has a single parameter, the maximum warping window, and DTW performs significantly better when this parameter is set through cross validation. A recent comparative study [2] found that the classifier rotation forest [16] (RotF) was

[1] http://www.timeseriesclassification.com

also a strong benchmark. It is able to discover relationships in time through the internal principle component transformation it uses, and is not significantly worse than DTW with window set through cross validation (DTWCV henceforth). The same study compared 22 TSC algorithms on 85 of the UCR archive data and found that just nine out of twenty two TSC algorithms were significantly more accurate than both a rotation forest and DTW classifier. Some of these TSC algorithms are highly complex and both memory and computationally expensive. A case was made that the superior algorithms achieved higher accuracy because the representation they use allows for the detection of discriminatory features that the benchmarks cannot find. This was further demonstrated on the archive and through data simulation [14]. We wish to test whether simple preprocessing can significantly improve the benchmarks and hence narrow the gap between DTW and rotation forest and the nine significantly better TSC algorithms.

2.2 Time Series Smoothing

Given a time series $T =< t_0, \ldots, t_{m-1} >$, a smoothing function produces a new series $S =< s_0, \ldots, s_{p-1} >$, where $p \leq m$ (we index from zero to make the equations simpler). Most algorithms employ a sliding window, of length w, along the series, resulting in a series of length $p = m - w$. The simplest form of smoothing is to take the **moving average (MA)** [9], often called the Simple Moving Average,

$$s_j = \frac{\sum_{i=j-w}^{j} t_i}{w} \quad \text{for} \quad j = w \ldots m-1,$$

where w is the single parameter, window size. **Exponential smoothing (EXP)** [9] is a generalisaton of moving average smoothing that assigns a decaying weight to each element rather than averaging over a window.

$$s_0 = t_0 \quad \text{and} \quad s_j = \alpha \cdot t_j + (1-\alpha) \cdot t_{j-1}$$

where $0 \leq \alpha \leq 1$. For consistency with other smoothing algorithms, EXP is often given a window size w, then the decay weight is set as $\alpha = \frac{2}{w+1}$.

A **Gaussian filter (GF)** [9] applies a fixed convolution over a window

$$s_j = \sum_{i=j-w}^{j} t_i \cdot c_i,$$

where the convolution values c_i are derived from a standard normal distribution over the window w, the single parameter.

Like GF, the **Savitzky-Golay (SG)** filtering method is a convolutional method of smoothing. Instead of using a fixed convolution, it estimates a different convolution on each window based on local least-squares polynomial approximation.

$$s_j = \sum_{i=j-w}^{j} t_i \cdot c_{i,j}$$

Since its initial introduction [17], it has been used successfully and pervasively across many signal processing domains for different purposes, particularly in chemometrics [7, 12]. SG has two parameters, window size w and polynomial order n. For accessible explanations of how the polynomial coefficients are calculated, we refer the reader to [18].

Discrete Fourier Approximation (DFT)[10] smooths the series by first transforming into the frequency domain, discarding the high frequency terms, then transforming back to the time domain. DFT has a single parameter, r, the proportion of Fourier terms to retain.

The **Recursive Median Sieve (SIV)** is a one-dimensional recursive median filter [3] that filters the data by removing extrema of specific scales. The sieve uses morphological scale-space operations, specifically openings and closings, or combinations of them, to filter an input signal. It does this by applying flat structuring elements to an input signal, which unlike conventional morphological operators such as those used in granulometries, have a fixed size but variable shape. They were introduced as a one-dimensional non-linear scale-space decomposition algorithm in [5], but can be extend to n-dimensions by adopting techniques from graph morphology [4].

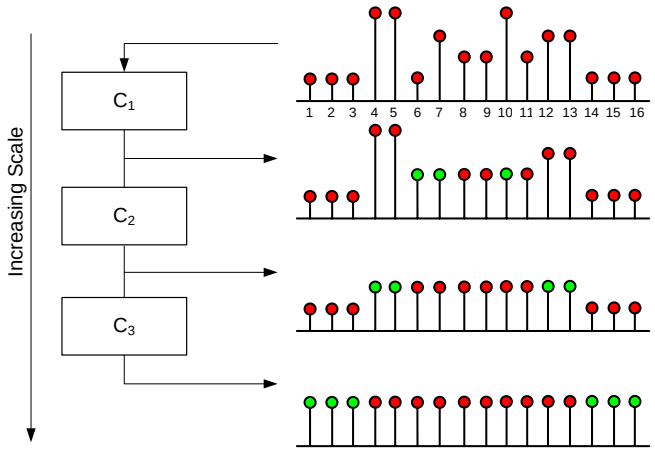

Fig. 1. An example sieve decomposition of a 1D signal. Green vertices are the vertices affected at each scale level.

The sieve performs a decomposition removing extrema (both maxima and minima) at different scales as shown in Figure 1. At the scale c_1 the maxima

and minima at points 6,7 and 10 are smoothed to equal the nearest value of the neighbours. At scale c_2 the pairs at (4,5) and (12,13) are smoothed. At the highest scale, the series is uniform. The sieve takes in a single parameter, c, which is the scale to smooth the signal to.

3 Experimental Setup

Table 1. The parameter spaces searched for each filtering method over the course of our experiments (default value in bold). m is the series length. For Savitzky-Golay (SG), all combinations of w, n are searched where $w > 2n$.

Method	Parameters and default values in bold
Moving Average (MA)	$w \in \{2, 3, \mathbf{5}, 10, 25, 50, 100, \sqrt{m}, \log_2(m)\}$
Exponential Smoothing (EXP)	$w \in \{2, 3, \mathbf{5}, 10, 25, 50, 100, \sqrt{m}, \log_2(m)\}$
Gaussian Filtering (GF)	$w \in \{2, 3, \mathbf{5}, 10, 25, 50, 100, \sqrt{m}, \log_2(m)\}$
Savitzky-Golay (SG)	$w \in \{\mathbf{5}, 9, 17, 33, 65\}$ $n \in \{\mathbf{2}, 3, 4, 8, 16, 32\}$
Fourier Approximation (DFT)	$r \in \{0.01, 0.05, \mathbf{0.1}, 0.25, 0.5, \log_2(m)/m\}$
Sieve (SIV)	$c \in \{\frac{1}{15} \cdot \log_{10}(m), \ldots, \frac{\mathbf{5}}{\mathbf{15}} \cdot \log_{\mathbf{10}}(\mathbf{m}), \ldots, \log_{10}(m)\}$

For each of pair of filter+classifier combination, we perform 10 stratified random resamples of each data set and report the average results across those resamples. The first resample, fold 0, is always the exact train/test split published on the UCR archive, to allow for easier comparison to existing work. To avoid ambiguity, we stress that in all cases the training of a classifier, including any parameter tuning and model selection required, is performed independently on the train set of a given fold, and the trained classifier is evaluated exactly once on the corresponding test set. We conduct 10 resamples on 76 of the (at the time of experimentation) 85 UCR archive TSC problems. We have omitted the largest problems - ElectricDevices, FordA, FordB, HandOutlines, NonInvasive-FatalECGThorax1, NonInvasiveFatalECGThorax2, PhalangesOutlinesCorrect, StarlightCurves, and UWaveGestureLibraryAll - as well as problems recently introduced into the expanded archive due to time constraints and a desire to maintain comparability with the results of [2]. Table 2 summarises the datasets used.

We average test accuracy over the 10 resamples, then present results in critical difference diagrams, which display the average ranks of the classifiers over all problems and group classifiers into cliques, within which there is no significant difference. For each resample, we perform a 10 fold cross validation (CV) on that resamples' train data to find smoothing parameters, such that no transfer learning of optimal parameters is performed.

For comparing multiple classifiers on multiple datasets, we follow the recommendation of Demšar [11] and use the Friedmann test to determine if there are

Table 2. The 76 UCR time series classification problems used in the experiments.

Dataset	Atts	Classes	Train	Test	Dataset	Atts	Classes	Train	Test
Adiac	176	37	390	391	Meat	448	3	60	60
ArrowHead	251	3	36	175	MedicalImages	99	10	381	760
Beef	470	5	30	30	MidPhalOutAgeGroup	80	3	400	154
BeetleFly	512	2	20	20	MidPhalOutCorrect	80	2	600	291
BirdChicken	512	2	20	20	MiddlePhalanxTW	80	6	399	154
Car	577	4	60	60	MoteStrain	84	2	20	1252
CBF	128	3	30	900	OliveOil	570	4	30	30
ChlorineConcentration	166	3	467	3840	OSULeaf	427	6	200	242
CinCECGtorso	1639	4	40	1380	Phoneme	1024	39	214	1896
Coffee	286	2	28	28	Plane	144	7	105	105
Computers	720	2	250	250	ProxPhalOutAgeGroup	80	3	400	205
CricketX	300	12	390	390	ProxPhalOutCorrect	80	2	600	291
CricketY	300	12	390	390	ProximalPhalanxTW	80	6	400	205
CricketZ	300	12	390	390	RefrigerationDevices	720	3	375	375
DiatomSizeReduction	345	4	16	306	ScreenType	720	3	375	375
DisPhalOutAgeGroup	80	3	400	139	ShapeletSim	500	2	20	180
DisPhalOutCor	80	2	600	276	ShapesAll	512	60	600	600
DislPhalTW	80	6	400	139	SmallKitchApps	720	3	375	375
Earthquakes	512	2	322	139	SonyAIBORSurface1	70	2	20	601
ECG200	96	2	100	100	SonyAIBORSurface2	65	2	27	953
ECG5000	140	5	500	4500	Strawberry	235	2	613	370
ECGFiveDays	136	2	23	861	SwedishLeaf	128	15	500	625
FaceAll	131	14	560	1690	Symbols	398	6	25	995
FaceFour	350	4	24	88	SyntheticControl	60	6	300	300
FacesUCR	131	14	200	2050	ToeSegmentation1	277	2	40	228
FiftyWords	270	50	450	455	ToeSegmentation2	343	2	36	130
Fish	463	7	175	175	Trace	275	4	100	100
GunPoint	150	2	50	150	TwoLeadECG	82	2	23	1139
Ham	431	2	109	105	TwoPatterns	128	4	1000	4000
Haptics	1092	5	155	308	UWaveX	315	8	896	3582
Herring	512	2	64	64	UWaveY	315	8	896	3582
InlineSkate	1882	7	100	550	UWaveZ	315	8	896	3582
InsectWingbeatSound	256	11	220	1980	Wafer	152	2	1000	6164
ItalyPowerDemand	24	2	67	1029	Wine	234	2	57	54
LargeKitchApps	720	3	375	375	WordSynonyms	270	25	267	638
Lightning2	637	2	60	61	Worms	900	5	181	77
Lightning7	319	7	70	73	WormsTwoClass	900	2	181	77
Mallat	1024	8	55	2345	Yoga	426	2	300	3000

any statistically significant differences in the rankings of the classifiers. However, following recommendations in [6] and [13], we have abandoned the Nemenyi post-hoc test originally used by [11] to form cliques (groups of classifiers within which there is no significant difference in ranks). Instead, we compare all classifiers with pairwise Wilcoxon signed-rank tests, and form cliques using the Holm correction (which adjusts family-wise error less conservatively than a Bonferonni adjustment).

Our code[2] reproduces the splits used in this evaluation exactly, and full, reproducible results are available[3]. For all smoothing algorithms except the sieve, we used the standard MATLAB implementations and performed the smoothing and classification in separate stages. The default parameters given in Table 1 are those of the Matlab implementations. The sieve is implemented in C and was similarly isolated from the classification stage.

[2] https://github.com/TonyBagnall/uea-tsc
[3] http://www.timeseriesclassification.com/Smoothing.php

We use three baseline classifiers. 1-NN with Euclidean distance is a weak baseline in a TSC context, but it is still frequently used in research. 1-NN with DTW is the most common benchmark, although it is important to set the window through cross validation [15]. This is computationally expensive, although we use the DTW version described in [19] which speeds up the calculation by orders of magnitude. All the UCR data are normalised. For consistency, we renormalise each series after smoothing.

4 Results

We present results for three baseline classifiers with both default smoothing and tuned smoothing through critical difference diagrams in Figure 2, and average accuracies are summarised in Table 3. For all three classifiers, smoothing of any kind provides no benefit. Tuning provides no benefit over using default values, and in many cases makes things worse due to overfitting.

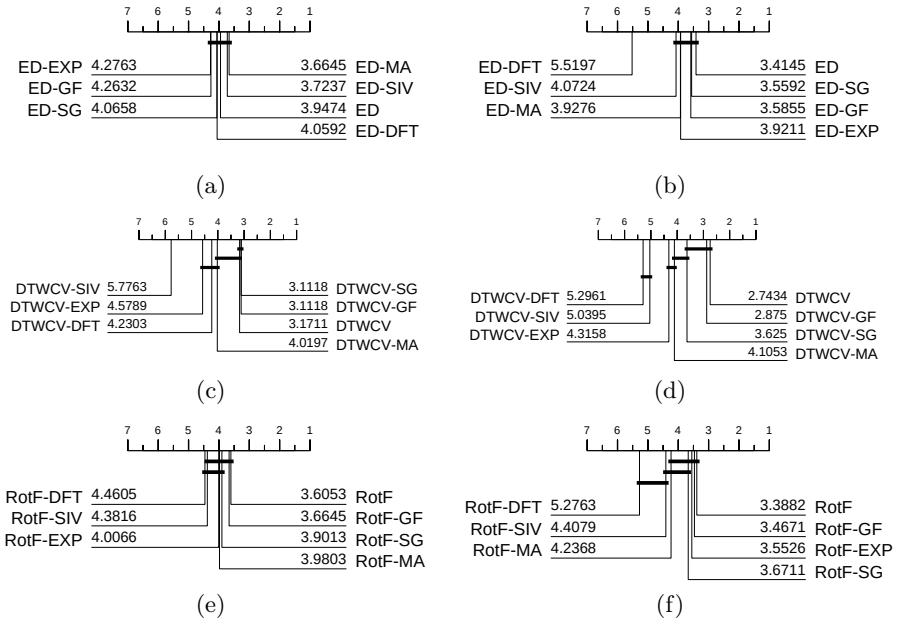

Fig. 2. Average ranks on 76 UCR problems while smoothing with the six methods described in Section 2, with default (left) and tuned (right) parameters using the parameters given in Table 1, for Euclidean distance ((a) and (b)), dynamic time warping ((c) and (d)) and rotation forest ((e) and (f)).

For all six experiments, the classifiers built on unsmoothed data are in the top clique. For four of the experiments, the unsmoothed classifier is the highest

Table 3. Average accuracies across the 76 datasets of the default (top) and tuned (bottom) smoothing methods when using the three benchmark classifiers. Within each group, classifier are ordered by rank to mirror Figure 2.

Untuned smoothing filters

	Accuracy		Accuracy		Accuracy
ED-MA	0.718	DTWCV-SG	0.77	RotF	0.769
ED-SIV	0.714	DTWCV-GF	0.769	RotF-GF	0.768
ED	0.714	DTWCV	0.771	RotF-SG	0.768
ED-DFT	0.716	DTWCV-MA	0.765	RotF-MA	0.767
ED-SG	0.716	DTWCV-DFT	0.764	RotF-EXP	0.768
ED-GF	0.716	DTWCV-EXP	0.763	RotF-SIV	0.761
ED-EXP	0.717	DTWCV-SIV	0.74	RotF-DFT	0.761

Tuned smoothing filters

	Accuracy		Accuracy		Accuracy
ED	0.714	DTWCV	0.771	RotF	0.769
ED-SG	0.717	DTWCV-GF	0.771	RotF-GF	0.769
ED-GF	0.716	DTWCV-SG	0.769	RotF-EXP	0.77
ED-MA	0.718	DTWCV-MA	0.763	RotF-SG	0.769
ED-EXP	0.719	DTWCV-EXP	0.764	RotF-MA	0.768
ED-SIV	0.716	DTWCV-SIV	0.748	RotF-SIV	0.76
ED-DFT	0.689	DTWCV-DFT	0.735	RotF-DFT	0.753

ranked. Setting the parameter through cross validation is if anything worse than using a default parameter. Given the order of magnitude more computation required to tune these parameters, this is surprising, particularly as *no smoothing* was one of the options. Further analysis shows that *no smoothing* was selected approximately 25% of the time. This could be an indication that the archive data are simply not suited to smoothing, however even in that case this suggests that the improvements being found by complex TSC classifiers cannot be easily explained away by simple, or even costly, attempts to smooth the data.

5 Analysis

We examine whether there are any characteristics of the data that could help determine whether any of the six types of smoothing would improve performance. We would expect that smoothing might be more useful for longer series. Figures 3 and 4 show the scatter plot of length against classifier rank for DTWCV and rotation forest. We find no obvious relationship between the performance of the unsmoothed classifier and series length. We repeated a similar analysis for number of instances and classes, but again found no correlation as one should expect when considering smoothing. Further, we find no significant areas where particular smoothing methods improve over the others.

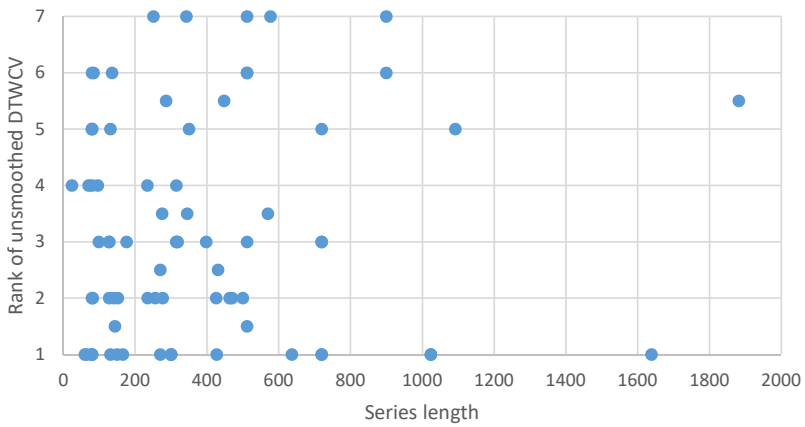

Fig. 3. Ranks per dataset on 76 UCR problems of unsmoothed DTWCV compared to six *untuned* smoothed versions plotted against series length. No obvious correlation is found.

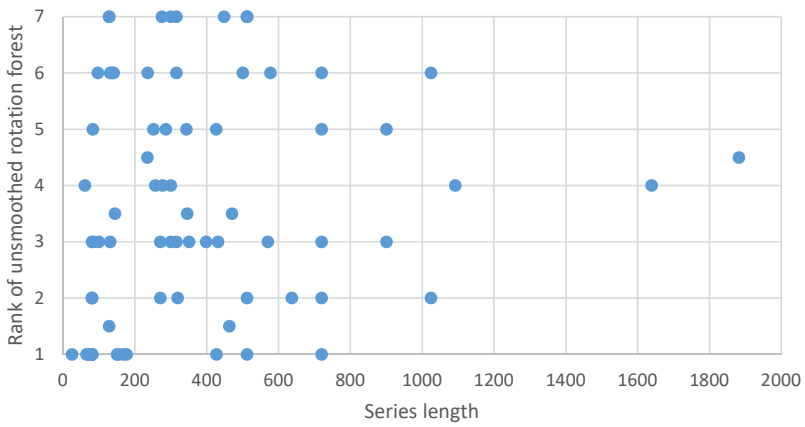

Fig. 4. Ranks per dataset on 76 UCR problems of unsmoothed rotation forest compared to six *untuned* smoothed versions plotted against series length. No obvious correlation is found.

6 Conclusion

It has long been a suspicion of many researchers in this field that much of the improvement seen in complex TSC algorithms could equally be achieved with comparatively simple preprocessing. Our experiments indicate for the case of smoothing, this is not true. We have taken six very popular smoothing algorithms and applied them using sensible default parameters and using extensive extra computation to discover optimal parameters through cross validation. We have found no significant difference between smoothed and unsmoothed classification with three benchmarks. The nature of the UCR data may explain this to a degree: the data from problems such as image processing will have less noise than, for example, financial data. We are not claiming that smoothing has no role to play in the analysis of time series data, merely that the automated application of smoothing without domain expertise does not on average improve the performance of baseline classifiers and that the absence of smoothing cannot explain the performance of algorithms that outperform the baselines.

Acknowledgement. This work is supported by the UK Engineering and Physical Sciences Research Council (EPSRC) [grant number EP/M015807/1] and Biotechnology and Biological Sciences Research Council [grant number BB/M011216/1]. The experiments were carried out on the High Performance Computing Cluster supported by the Research and Specialist Computing Support service at the University of East Anglia and using a Titan X Pascal donated by the NVIDIA Corporation.

References

1. Y. Chen B. Hu and E. Keogh. Time series classification under more realistic assumption. In *Proc. 13th SIAM International Conference on Data Mining*, 2013.
2. A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31(3):606–660, 2017.
3. J. Bangham. Data-sieving hydrophobicity plots. *Analytical biochemistry*, 174(1):142–145, 1988.
4. J. Bangham, R. Harvey, P. Ling, and R. Aldridge. Morphological scale-space preserving transforms in many dimensions. *Journal of Electronic Imaging*, 5:283–299, 1996.
5. J. Bangham, P. Ling, and R. Harvey. Scale-space from nonlinear filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(5):520–528, 1996.
6. A. Benavoli, G. Corani, and F. Mangili. Should we really use post-hoc tests based on mean-ranks? *Journal of Machine Learning Research*, 17:1–10, 2016.
7. G. Betta, D. Capriglione, G. Cerro, L. Ferrigno, and G. Miele. The effectiveness of Savitzky-Golay smoothing method for spectrum sensing in cognitive radios. *Proceedings of the 2015 18th AISEM Annual Conference*, pages 1–4, 2015.
8. Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen, and G. Batista. The UEA-UCR time series classification archive. `http://www.cs.ucr.edu/~eamonn/time_series_data/`, 2015.

9. Y. Chou. *Statistical Analysis*. Holt International, 1975.
10. J. Cooley, P. Lewis, and P. Welch. The fast fourier transform and its applications. *IEEE Transactions on Education*, 12(1):27–34, 1969.
11. J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.
12. B. Fernandes, G. Colletta, L. Ferreira, and O. Dutra. Utilization of Savitzky-Golay filter for power line interference cancellation in an embedded electrocardiographic monitoring platform. *Proc. IEEE International Symposium on Medical Measurements and Applications*, pages 7–12, 2017.
13. S. García and F. Herrera. An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons. *Journal of Machine Learning Research*, 9:2677–2694, 2008.
14. J. Lines, S. Taylor, and A. Bagnall. HIVE-COTE: The hierarchical vote collective of transformation-based ensembles for time series classification. In *Proc. IEEE International Conference on Data Mining*, 2016.
15. C. Ratanamahatana and E. Keogh. Three myths about dynamic time warping data mining. In *Proc. 5th SIAM International Conference on Data Mining*, 2005.
16. J. Rodriguez, L. Kuncheva, and C. Alonso. Rotation forest: A new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1619–1630, 2006.
17. A. Savitzky and M. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, 36(8):1627–1639, 1964.
18. R. Schafer. What is a Savitzky-Golay Filter? *IEEE Signal Processing Magazine*, 28(4):111–117, 2011.
19. C. Tan, M. Herrman, G. Forestier, G. Webb, and F. Petitjean. Efficient search of the best warping window for dynamic time warping. In *Proc. 18th SIAM International Conference on Data Mining*, 2018.