Time-Series Classification with COTE: The Collective of Transformation-Based Ensembles

Anthony Bagnall, Jason Lines, Jon Hills, and Aaron Bostrom

Abstract—Recently, two ideas have been explored that lead to more accurate algorithms for time-series classification (TSC). First, it has been shown that the simplest way to gain improvement on TSC problems is to transform into an alternative data space where discriminatory features are more easily detected. Second, it was demonstrated that with a single data representation, improved accuracy can be achieved through simple ensemble schemes. We combine these two principles to test the hypothesis that forming a collective of ensembles of classifiers on different data transformations improves the accuracy of time-series classification. The collective contains classifiers constructed in the time, frequency, change, and shapelet transformation domains. For the time domain, we use a set of elastic distance measures. For the other domains, we use a range of standard classifiers. Through extensive experimentation on 72 datasets, including all of the 46 UCR datasets, we demonstrate that the simple collective formed by including all classifiers in one ensemble is significantly more accurate than any of its components and any other previously published TSC algorithm. We investigate alternative hierarchical collective structures and demonstrate the utility of the approach on a new problem involving classifying *Caenorhabditis elegans* mutant types.

Index Terms—Time series classification, ensemble, shapelet

1 Introduction

Time-series classification (TSC) problems, where we may consider any ordered data to be time-series data, arise in a wide range of disciplines. The establishment of the UCR repository for TSC problems [1] has engendered growth in the number of algorithms proposed for TSC (for example, see [2], [3], [4], [5], [6], [7], [8], [9]). These algorithms are often evaluated on the same datasets, and the admirable trend of releasing source code makes it feasible to compare and test for significant differences in accuracy.

We propose a simple approach to TSC based on transformation and ensembling that is significantly more accurate than any other algorithm that we are aware of, including the standard benchmark nearest neighbour (NN) classifiers. We believe our classifier, which we call the collective of transformation-based ensembles (COTE), provides a new benchmark in accuracy performance against which other classifiers must be measured.

The motivation for COTE comes from our recent research [10], [11], [12] which has explored two key ideas for TSC. Our starting hypothesis was that the simplest way to gain improved accuracy on TSC problems is to transform into an alternative data space where the discriminatory features are more easily detected. In [10], we showed that classifiers constructed on the power spectrum (PS), auto-correlation function (ACF) and time domain were more accurate than any of the constituent classifier parts. More recently, we demonstrated that the best way of using *shapelets* (short, discriminatory subsequences first defined for TSC in [13]) is as a shapelet transformation, which forms a new data space [11]. Our second hypothesis was that we can improve TSC performance through ensembling. Although the value of ensembling is well known, our approach is unusual in that we inject diversity by adopting a heterogeneous ensemble rather than by using resampling schemes with weak learners. Our approach is in fact a meta-ensemble, since two of the components (random forest and rotation forest) are themselves ensembles. We have demonstrated the effectiveness of heterogeneous ensembles in the time domain. Elastic distance measures such as dynamic time warping (DTW) are by far the most popular approach for TSC. In [12] we showed that the variety of elastic distance measures that have recently been proposed [4], [8], [9] are not individually significantly better than DTW, but when combined into a heterogeneous elastic ensemble (EE), the assimilated elements contribute to an overall significantly greater accuracy than any of the constituent parts.

We now take the logical next step of combining transformations and ensembles. COTE contains classifiers constructed in the time, frequency, change, and shapelet transformation domains combined in alternative ensemble structures. The details of the transformations are described in Section 3. For the time domain, we use the nearest neighbour classifiers from the elastic ensemble [12]. For the other domains, we use a range of standard classifiers described in Section 4. Classification involves a weighted vote of members of the collective. We evaluate accuracy on the benchmark 46 UCR datasets. We believe that EE was the first classifier to be significantly more accurate than DTW on the UCR datasets, and yet we show that COTE is significantly more accurate than EE. We extend the study of the classification ability of COTE to a further 26 datasets that have been used in the literature but are not part of the standard UCR data set, including two completely new problems involving

• *The authors are with the School of Computing Sciences, University of East Anglia, Norwich, Norfolk, United Kingdom.*
E-mail: {ajb, j.lines, j.hills, a.bostrom}@uea.ac.uk.

Manuscript received 4 Nov. 2014; revised 12 Feb. 2015; accepted 15 Mar. 2015. Date of publication 25 Mar. 2015; date of current version 3 Aug. 2015.
Recommended for acceptance by T. Palpanas.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2015.2416723

classifying *Caenorhabditis elegans* types based on motion capture data. Our contributions can be summarised as follows:

1) There has been a recent glut of new TSC algorithms using a wide range of techniques (see Section 2 for a review). The results presented in Section 7.1 show that COTE is significantly more accurate than them all, including our own algorithm presented in [12].
2) We propose a simple heterogeneous ensemble that reduces classification induced variance.
3) We describe new results for the shapelet transform using the heterogeneous ensemble. This approach is significantly better than 1NN-DTW. We compare our results with those found using alternative shapelet algorithms.
4) We propose a new way of using the autocorrelation function transform for TSC, involving concatenating autocorrelation, partial autocorrelation and autoregressive (AR) features.
5) We propose a novel way of choosing between transformation spaces. We test the hypothesis that the transformation is more important than the classifier through a series of experiments and demonstrate that the interaction between classifier and transform is more complex than we initially thought.

In total, we utilise 35 classifiers. The simplest way of combining these classifiers, which we call flat-COTE, ensembles all 35 classifiers proportionate to their training set cross validation accuracy. This approach is the most accurate, but the least explanatory. We investigate ways of forming hierarchical ensembles through choosing subsets of data representations to use based on training set performance. Based on our previous research [10], our *a priori* hypothesis was that if we could choose the best transformation we would arrive at a better classifier, because we assumed the choice of representation was more important than the choice of classifier. It turns out that the truth is more complex than that. Many of the data sets have discriminatory features in multiple domains, and choosing the transformation based on train set performance actually makes COTE significantly worse. We investigate alternative hierarchical collective structures that use weighting schemes and selection schemes between ensembles on different transforms. We demonstrate that although most approaches give significantly worse accuracy than the flat approach of a single ensemble, a collective of transform-based ensembles where inclusion is determined by a Mann-Whitney rank sign test is not significantly worse.

The structure of this paper is as follows. In Section 2 we provide some background into time series classification and the algorithms that have been proposed in the literature. In Section 3 we describe the data transforms used in the ensemble and in Section 4 we identify the classifiers we use on each data representation. Section 5 outlines the datasets we use for the experimentation, the results of which are presented in Sections 6, 7 and 8. Finally, we conclude and highlight future directions in Section 9.

2 Time Series Classification Background

We define time series classification as the problem of building a classifier from a collection of labelled training time series. We limit our attention to problems where each time series has the same number of observations. Suppose we have a set of n time series, $\mathbf{T} = \{T_1, T_2, \ldots, T_n\}$, where each time series has m ordered real-valued observations $T_i = < t_{i1}, t_{i2}, \ldots, t_{im} >$ and a class value c_i. The objective is to find a function that maps from the space of possible time series to the space of possible class values.

The key characteristic that differentiates TSC problems from the general classification task is that the ordering of the attributes is important. The best discriminatory features for classification might be masked by the length of the series, confounded by noise in the phase of the series or embedded in the interaction of observations. Hence, TSC generally requires techniques specific to the nature of the problem. The alternative approaches to TSC are best understood by considering how the data is represented or, equivalently, how similarity between series is quantified. Similarity between series can be based on several discriminating criteria, such as: similarity in time, spectra or autocorrelation structure; global or local similarity; and data driven or model based similarity.

2.1 Similarity in the Time Domain

Similarity in time is characterised by the situation where the series from each class are observations of an underlying common curve. Variation around this underlying common shape may be caused by noise or possible phase shift. The majority of research into TSC has concentrated on data driven global similarity in time. The commonly used benchmark classification algorithm is 1-NN with an elastic distance such as dynamic time warping or edit distance to allow for small shifts in the time axis. As first identified in [14] and confirmed through extensive experimentation [15], 1-NN DTW with the warping window size set through cross-validation on the training data, is surprisingly hard to beat. A number of new elastic measures have been proposed that are variations of the time warp and edit distance approaches [4], [8], [9]. Two main classes of technique have been proposed for detecting localised, phase independent, similarity in time. The first involves finding shapelets in the dataset [16]. Shapelets are discriminatory subseries in the data. We discuss recent shapelet research in more detail in Section 3.1.

The second popular localised approach involves deriving features from varying size intervals of the series [2], [3], [5]. Lin et al. [2] propose a bag-of-patterns (BoP) approach that involves converting a time series into a discrete series using symbolic aggregate approximation (SAX) [17], creating a set of SAX words for each series through the application of a short sliding window, then using the frequency count of the words in a series as the new feature set. An alternative is to use summary statistics calculated over different width intervals of a series. For a series of length m, there are $m(m-1)/2$ possible contiguous intervals. Deng et al. [5] calculate three statistics over these intervals: the mean, standard deviation, and slope on each of the possible intervals. They use these features to construct classifiers. Rather than generate the entire new feature space of $3m(m-1)/2$ attributes, they employ a random forest classifier, with each member of the ensemble assigned a random subset of features from the interval feature space.

Baydogan et al. [3] describe a bag-of-features approach that combines interval and frequency features. The algorithm, called time series based on a bag-of-features representation (TSBF), involves separate feature creation and classification stages. The feature creation stage involves generating random intervals and then creating features representing the mean, variance, and slope over the interval. The start and end point of the interval are also included as features in order to retain the possibility of detecting temporal similarity. There is then a further feature transform that involves supervised learning. The features of each interval form an instance, and each time series represents a bag. A classifier is used to generate a class probability estimate for each instance. The probability estimates of all instances for a given time series (bag) are discretised, and a histogram for each possible class value is formed. The resultant concatenated histograms form the feature space for the training set of a classifier. A random forest classifier is used for the labelling, and a random forest and support vector machine for the classification.

2.2 Similarity in the Frequency Domain

Similarity in spectra relates to the situation where the relevant discriminatory features are in the frequency domain of each series. Data driven approaches commonly use the periodogram, or power spectrum of the whole series derived from the Fourier transform [10].

2.3 Similarity in Autocorrelation

The autocorrelation function describes the correlation within the series over a range of lags. The Fourier transform of the ACF of a series is in fact the power spectrum, but the ACF is more useful than the spectrum for detecting lower order relationships between series terms. In time series forecasting, the ACF is most commonly used in conjunction with the partial ACF (PACF) to fit an auto regressive moving average (ARMA) model to a series. In time series data mining, its primary usage has also been to fit ARMA models, the parameters of which are then used as discriminatory features [18]. Other research has used the ACF and PACF as the features for a classifier [10], [19]. We use a combination of these features in a way detailed in Section 3.2.

An overview of some of the ways the periodogram and ACF can be used for time-series classification is given in [20]. Our approach is described in Section 3.2.

Another thread of research that is harder to classify examines using complexity measures of the series to differentiate classes. Batista et al. [7] propose an alternative distance measure based on difference in complexity. Silva and de Souza [21] propose using recurrence plots in conjunction with a Kolmogorov complexity based distance measure.

Finally, and perhaps most relevant to our work, Fulcher and Jones [22] define a massive feature space involving time, frequency and autocorrelation features then use a greedy forward feature selection method with a linear discriminant classifier.

We compare the results for the all of these classifiers against COTE in Section 7.1.

3 Data Transformations

3.1 Localised Similarity in Shape in the Time Domain: Shapelet Transform

A shapelet [13] is a time-series subseries used for time-series classification. A good shapelet discriminates between classes using *shapelet distance* (*sDist*). For a shapelet S of length l, and a time series T, the $sDist$ is the minimum euclidean distance between the shapelet and any length l subseries of T. Let the set of length l subseries of T be denoted W_l, then

$$sDist(S,T) = min_{w \in W_l}(dist(s,w)).$$

A good shapelet will have small $sDist$s to instances of one class, and large $sDist$s to instances of any other class. We transform the original data using the best shapelets as features, where attribute i in instance j of the transformed data is $sDist(S_i, T_j)$, where S_i is the ith best shapelet and T_j is the jth instance of the original data.

The algorithm we use to discover shapelets and transform the data is described in Algorithm 1. It makes a single pass through the original data, taking each subseries of each series as a shapelet candidate. The set of $sDist$ values for each candidate is found using $findDistances$ and assessed using the *f-stat* quality measure in the $assessCandidate$ procedure. The best k shapelets are returned, after removing overlapping candidates in the method $removeSelfSimilar$. We use the length estimation procedure described in [23] to determine the appropriate values to use as the minimum and maximum shapelet lengths, and generate a maximum of $k = 10n$ shapelets, where n is the size of the training set of the original data.

Algorithm 1. ShapeletCachedSelection($\mathbf{T}$, min, max, k)

```
1:  $kShapelets \leftarrow \emptyset$
2:  for all $T_i$ in $\mathbf{T}$ do
3:    $shapelets \leftarrow \emptyset$
4:    for $l \leftarrow min$ to $max$ do
5:      $W_{i,l} \leftarrow generateCandidates(T_i, l)$
6:      for all subsequence $S$ in $W_{i,l}$ do
7:        $D_S \leftarrow findDistances(S, \mathbf{T})$
8:        $quality \leftarrow assessCandidate(S, D_S)$
9:        $shapelets.add(S, quality)$
10:   $sortByQuality(shapelets)$
11:   $removeSelfSimilar(shapelets)$
12:   $kShapelets \leftarrow merge(k, kShapelets, shapelets)$
13: return $kShapelets$
```

The aim of the research described in [11] was to demonstrate that transformation was better than using a shapelet tree by evaluation of a range of classifiers on transformed datasets. Further experimentation has allowed us to draw stronger conclusions about the utility of the shapelet transform. These are described in Section 6.

3.2 Frequency Domain: Periodogram Transform

For a real-valued time series $T = < t_1, t_2, \ldots, t_m >$, the discrete fourier transform (DFT) represents T as a linear combination of sinusoidal functions with amplitudes a, b and phase w,

$$t_x = \sum_{k=1}^{m} (a_k \cos(2\pi \cdot w_k \cdot x) + b_k \sin(2\pi \cdot w_k \cdot x)).$$

The periodogram (or spectrum) is the series

$$P =< p_1, p_2, \dots, p_m >,$$

where

$$p_i = \sqrt{a_i^2 + b_i^2}.$$

The periodogram is the Fourier transform of the ACF. The spectrum and ACF are different characterizations of the same information. The ACF is more useful for finding low-order dependencies between the terms; the periodogram is more useful for detecting lower-frequency correlations than the ACF. The first DFT coefficient of a series with zero mean will be zero. Since we always work with normalised series, we can ignore this term. In addition, the DFT of a real-valued series is symmetric, so that $(a_i, b_i) = (a_{m-i-1}, b_{m-i-1})$. This means we can discard half of the periodogram. The periodogram transform is then $P =< p_2, p_3, \dots, p_{m/2} >$.

3.3 Autocorrelation-Based Transform

The Autocorrelation function measures the interdependence of terms in the time domain, and is commonly used in statistics and speech processing to model data where there is a dependency between observations over a short period of time. Positive autocorrelation in a series generally indicates some form of persistence, in that the series tends to remain in the previous state, whereas negative autocorrelation is indicative of high volatility. The ACF of time series T is $\rho =< \rho_1, \rho_2, ..., \rho_{m-l} >$ (where l is the maximum lag), where

$$\rho_k = \frac{E[(t_i - \mu_i) \cdot (t_{i+k} - \mu_{i+k})]}{\sigma_i \cdot \sigma_{i+k}}.$$

ρ_k are usually estimated from data by r_k, where

$$r_k = \frac{\sum_{i=1}^{m-k} (t_i - \bar{t})(t_{i+k} - \bar{t})}{\sum_{i=1}^{m} (t_i - \bar{t})^2}.$$

The quantity r_k is the autocorrelation coefficient at lag k and has range $[-1, 1]$. If the series T has been normalised to zero mean and unit variance, the calculation of r_k simplifies to

$$r_k = \sum_{i=1}^{m-k} (t_i \cdot t_{i+k}).$$

The autocorrelation function is often used to fit an autoregressive model to a time series. An AR model is of the form

$$t_i = c + \sum_{j=1}^{p} \phi_i t_{t-j} + \varepsilon_i,$$

where c is a constant, ϕ_i are model parameters and ε_i are random variables (usually assumed to be independent and identically distributed). Estimates of the parameters ϕ_i are found by first estimating the partial autocorrelation function (PACF). The PACF describes the autocorrelation between variables t_i and t_{i+k}, with the linear dependence between t_{i+1} and t_{i+k-1} removed. The sample PACF is calculated from the sample ACF. For any given value of p there are an associated set of parameters $\Lambda_p = (\lambda_1, \lambda_2, \dots, \lambda_p)$ that satisfy

$$R_p = \Lambda_p \Phi_p,$$

where $R_p = (r_1, r_2, \dots, r_p)$ are the first p terms of the ACF and Φ_p is a Toepliz matrix of ACF terms defined as

$$\Phi_p = \begin{bmatrix} 1 & r_1 & r_2 & \cdots & r_{p-1} \\ r_1 & 1 & r_2 & \cdots & r_{p-2} \\ r_2 & r_1 & 1 & \cdots & r_{p-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{p-1} & r_{p-2} & r_{p-3} & \cdots & 1 \end{bmatrix}.$$

The system of linear equations defined by $R_p = \Lambda_p \Phi_p$ can be solved for Λ_p,

$$\Lambda_p = \Phi_p^{-1} R_p.$$

So for all values of p we have

$$\Lambda = \begin{bmatrix} \Lambda_1 \\ \Lambda_2 \\ \vdots \\ \Lambda_{m-l} \end{bmatrix} = \begin{bmatrix} \lambda_{1,1} & & & & \\ \lambda_{2,1} & \lambda_{2,2} & & & \\ \lambda_{3,1} & \lambda_{3,2} & \lambda_{3,3} & & \\ \vdots & & & & \\ \lambda_{m-l,1} & \lambda_{m-l,2} & \lambda_{m-l,3} & \cdots & \lambda_{m-l,m-l} \end{bmatrix},$$

where l is the maximum lag. The PACF is defined as the vector of values

$$L =< \lambda_{1,1}, \lambda_{2,2}, \dots, \lambda_{m-l,m-l} > .$$

Finding L involves solving $m - l$ systems of linear equations. However, since Φ is a Toepliz matrix (all the diagonals are constant), the equations can be efficiently solved using the Durbin-Levinson algorithm.

The parameters of an AR model of order p are estimated from the Λ row p, i.e. $W =< w_1, w_2, \dots, w_p >$ where $w_i = \lambda_{p,i}$. The order of the model, p, is usually chosen to minimize some criteria such as the Akaike Information Criterion (AIC), or the Bayes Information Criterion (BIC).

The maximum lag, l, determines the length of the series R, L, and W. By definition, the higher the lag, the less data is available for the estimate, and the higher the variability in the ACF. It is common to restrict the maximum lag severely, and in all experiments we use the maximum lag size of $m/4$ or 100, whichever is smaller.

We have several options as to what variables to use to capture discriminatory features in the change domain. We could use the ACF (R), the PACF (L), or the AR model (W), individually or in any combination, with p set arbitrarily or through some selection criteria. We evaluate these alternatives in Section 6.2.

4 CLASSIFIERS

4.1 Heterogeneous Ensemble

The classifiers used are the WEKA [24] implementations of k Nearest Neighbour (where k is set through cross validation), Naive Bayes, C4.5 decision tree [25], support vector machines [26] with linear and quadratic basis function kernels, Random Forest [27] (with 100 trees), Rotation Forest [28] (with 10 trees) and a Bayesian network. Each classifier is assigned a weight based on the cross validation training accuracy, and new data are classified with a weighted vote. The set of classifiers were chosen to balance simple and complex classifiers that use probabilistic, tree based and kernel based models. With the exception of k-NN, we do not optimise parameter settings for these classifiers via cross validation. Our primary justification for forming heterogenous ensembles of strong classifiers is to minimize the variance of the classifiers over different transformations.

We chose to do this to reduce the complexity of the algorithm and to keep the focus of this research on the importance of transformation in TSC. Furthermore, we do not perform any model selection through classifier selection based on training performance. This extra level of cross validation may yield improved classifiers, but introduces a computational overhead.

4.2 Elastic Ensemble

We use the heterogeneous ensemble of eight classifiers for datasets in the frequency, change, and shapelet transformation domains. For the time domain, we use Elastic Ensemble classifier [12]. The EE is a combination of nearest neighbour classifiers that use elastic distance measures. There is a general consensus that *"simple nearest neighbor classification is very difficult to beat"* [7]. dynamic time warping with warping set through cross-validation (DTWCV) is the commonly used benchmark. There have been a number of variants of DTW. These include a weighted version of DTW (WDTW) [8] that replaces the warping window with a weight function to penalise against large warpings. Alternative elastic measures based on edit distance have also been proposed. These include a distance measure based on the Longest Common Subsequence (LCSS) problem, Edit Distance with Real Penalty (ERP) [29], Time Warp Edit (TWE) distance [9] and Move-Split-Merge (MSE) [4]. These are all constituents in the elastic ensemble.

In [12], we show that none of these individual measures significantly outperforms DTWCV. However, we demonstrate that by combining the predictions of 1-NN classifiers built with these distance measures and using a voting scheme that weights according to cross-validation training set accuracy, we can significantly outperform DTWCV. The 11 classifiers in EE are 1-NN with euclidean distance (ED), full dynamic time warping, DTW with window size set through cross validation (DTWCV), derivative DTW with full window and window set through cross validation (DDTW and DDTWCV), weighted DTW (WDTW) and derivative weighted DTW (WDDTW) [8], longest common subsequence, Edit Distance with Real Penalty [29], Time Warp Edit distance [9], and the Move-Split-Merge distance metric [4]. EE outperforms a heterogenous ensemble constructed by treating the time-series as vector features. Fig. 1 shows the scatter plot of accuracies of the EE classifier against the heterogeneous ensemble classifier constructed in the time domain. The EE is significantly better than the time-based heterogeneous ensemble, winning on 46 datasets, losing on 23, with 3 ties. Further experimental comparison of time-based and NN elastic ensembles can be found in [30].

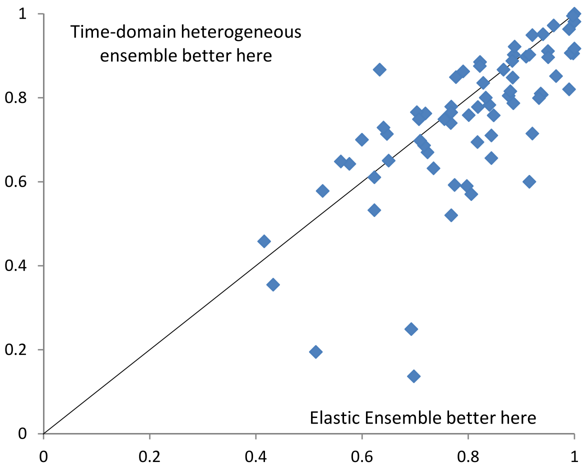


Fig. 1. Test accuracy of the elastic ensemble versus heterogeneous ensemble in the time domain over 72 problems.

5 Datasets

We have collected 72 datasets, the names of which are shown in Table 1. Forty six of these are available from the UCR repository [1], 24 were used in other published work [6], [11], [12], [23] and two are new datasets we present for the first time. Further information and the datasets we have permission to circulate are available from [31]. We have removed the dataset ECG200 from all experiments, because an error in data processing means that it can be perfectly classified with a single rule on the sum of squared values for each series (see [10] for further details). Furthermore, as also recommended in [10], we have normalised the datasets Coffee, Olive Oil, and Beef.

5.1 Classifying Mutant Worms

Caenorhabditis elegans is a roundworm commonly used as a model organism in the study of genetics. The movement of these worms is known to be a useful indicator for understanding behavioural genetics. Brown et al. [32] describe a system for recording the motion of worms on an agar plate and measuring a range of human-defined features [33]. It has been shown that the space of shapes *Caenorhabditis elegans* adopts on an agar plate can be represented by combinations of four base shapes, or *eigenworms*. Once the worm outline is extracted, each frame of worm motion can be captured by four scalars representing the amplitudes along each dimension when the shape is projected onto the four eigenworms (see Fig. 2). Using data collected for the work described in [32], we address the problem of classifying individual worms as wild-type or mutant based on the time series of the first eigenworm, down-sampled to second-long intervals. We have 257 cases, which we split 70 percent/30 percent into a train and test set. Each series has 900 observations, and each worm is classified as either wild-type (the N2 reference strain—109 cases) or one of four mutant types: goa-1 (44 cases); unc-1 (35 cases); unc-38 (45 cases) and unc-63 (25 cases). The data were extracted from the *C. elegans* behavioural database [34]. The formatted classification problems are available from the website associated with this paper [31].

Our primary goal is to use these data sets to test the hypothesis that ensembling across transformations

TABLE 1
Datasets Grouped by Problem Type

		Image Outline Classification			
DistPhalanxAge	DistPhalanxOutline	DistPhalanxTW	FaceAll	FaceFour	WordSynonyms
MidPhalanxAge	MidPhalanxOutline	MidPhalanxTW	OSULeaf	Phalanges	yoga
ProxPhalanxAge	ProxPhalanxOutline	ProxPhalanxTW	Herring	SwedishLeaf	MedicalImages
Symbols	Adiac	ArrowHead	BeetleFly	BirdChicken	DiatomSize
	FacesUCR	fiftywords	fish		
		Motion Classification			
CricketX	CricketY	CricketZ	UWaveX	UWaveY	UWaveZ
GunPoint	Haptics	InlineSkate	ToeSeg1	ToeSeg2	MutantWorms2
		MutantWorms5			
		Sensor Reading Classification			
Beef	Car	Chlorine	Coffee	Computers	
FordA	FordB	ItalyPower	LargeKitchen	Lightning2	Lightning7
StarLightCurves	Trace	wafer	RefrigerationDevices	MoteStrain	Earthquakes
ElectricDevices	SonyRobot1	SonyRobot2	OliveOil	Plane	Screen
		SmallKitchen			
		Human Sensor Reading Classification			
	TwoLeadECG	ECGFiveDays	ECGThorax1	ECGThorax2	
		Simulated Classification Problems			
	MALLAT	CBF	SyntheticControl	TwoPatterns	

The actual file names are in a string array in the supporting code.

significantly improves accuracy. Our secondary goal is to explore alternative ways of combining classifiers and ensembles to try and improve the accuracy of the overall classifier and provide exploratory insights into a particular classification problem. All datasets are split into a training and testing set, and all parameter optimisation is conducted on the training set only. We have made every effort to remove bias. We made all design decisions prior to evaluation on the test data and have selected data sets through collaboration with domain experts rather than to optimise performance. For the majority of our experiments, we use a single train/test split. We do this for for two reasons. First, it is almost universal practice to do so with the UCR datasets (for example [2], [3], [4], [6], [7], [8], [9], [22], [35], all perform single train/test experiments) and it makes sense for us to do so also in order to allow for a fair comparison. Second, some of the data sets are designed so the train/test split removes bias. For example, the electric devices problem involves repeated readings from electrical devices in several households. The train/test split is constructed so that all the data from a particular household is either in the train set or the test set. If we allow readings from a specific device to be in both train and test sets we introduce bias, because matching a specific device is easier than learning to classify all devices of a given type. Hence, the majority of the results we present are for the standard train/test splits. However, we also recognise that the field of time series classification should move towards evaluation through resampling and/or cross-validation. In Section 7.4 we present results of a resampling experiment using a subset of the 72 data sets.

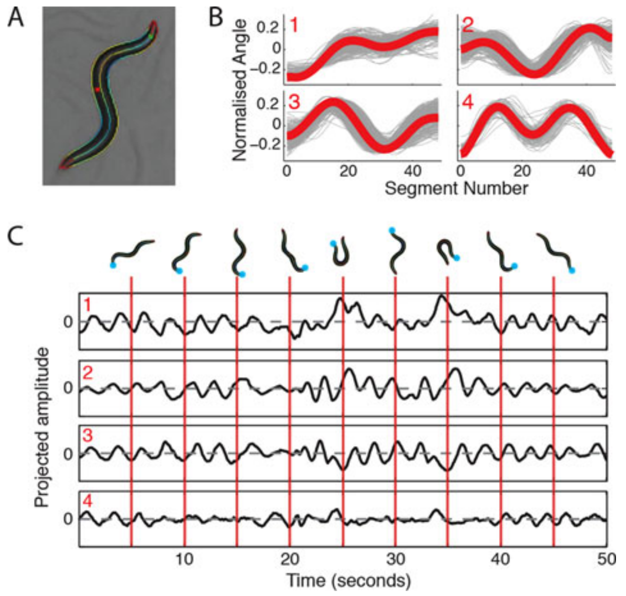


Fig. 2. (A) a worm on an agar plate. (B) four representative eigenworms. (C) example time series. Images taken with permission from [32]

6 Single Transform Results

6.1 Shapelet Ensemble (SE)

The shapelet transform used in conjunction with the heterogeneous ensemble described in Section 4.1 produces a classifier that is significantly more accurate than DTWCV, albeit only marginally. On the 72 datasets, the shapelet ensemble is better on 41, ties on 4 and is worse on 27. This gives a p value of 0.057 with the binomial test (BT) and 0.0152 with a Mann-Whitney test. If we restrict our attention to just the 46 UCR datasets, then SE is better on 25, worse on 17 and ties on 4. There is no significant difference between the classifiers on the UCR data. Thus we claim there is weak evidence that SE is better than DTWCV, but the overall difference is small. Full results are available from [31] and the UCR results are shown in Table 2 below for reference. Perhaps more relevant to COTE is the variability in the results. The

TABLE 2
Collated Published Results on the UCR Data Sets

	ED	DTW	TWED	WDTW	MSM	TSF	TSBF	BoP	CID	RPCD	FBL	LTS	SE	COTE	Best
Adiac	0.389	0.391	0.376	0.364	0.384	0.261	0.245	0.432	0.379	0.384	0.355	0.437	0.435	**0.233**	COTE
Beef	0.467	0.467	0.533	0.6	0.5	0.3	0.287	0.433	0.467	0.367	0.433	0.24	0.167	**0.133**	COTE
Car	0.267	0.233											0.267	**0.133**	COTE
CBF	0.148	0.004	0.007	0.002	0.012	0.039	0.009	0.013	0.001		0.289	0.006	0.003	**0.001**	CID
ChlorineCon	0.35	0.35				**0.26**	0.336		0.351	0.489		0.349	0.3	0.314	TSF
CinCECGTorso	0.103	0.07				0.069	0.262		0.054	**0.021**		0.167	0.154	0.064	RPCD
Coffee	0.25	0.179	0.214	0.133	0.236	0.071	0.004	0.036	0.179	**0**	**0**	0	0	**0**	Tie
CricketX	0.426	0.236				0.287	0.278		0.249	0.261		0.209	0.218	**0.154**	COTE
CricketY	0.356	0.197				0.2	0.259		0.197	0.292		0.249	0.236	**0.167**	COTE
CricketZ	0.38	0.18				0.239	0.263		0.205	0.292		0.201	0.228	**0.128**	COTE
DiatomSizeR	0.065	0.065				0.101	0.126		0.065	0.036		**0.033**	0.124	0.082	LTS
ECGFiveDays	0.203	0.203				0.07	0.183		0.218	0.136		**0**	0.001	**0**	Tie
FaceAll	0.286	0.192	0.189	0.257	0.189	0.231	0.234	0.219	0.144	0.19	0.292	0.218	0.263	**0.105**	COTE
FaceFour	0.216	0.114	0.024	0.136	0.057	0.034	0.051	**0.023**	0.125	0.057	0.261	0.048	0.057	0.091	BoP
FacesUCR	0.231	0.088				0.109	0.09		0.102	0.058		0.059	0.087	**0.057**	COTE
fiftywords	0.369	0.242	**0.187**	0.194	0.196	0.277	0.209	0.466	0.226	0.226	0.453	0.2323	0.281	0.191	TWED
fish	0.217	0.16	0.051	0.126	0.08	0.154	0.08	0.074	0.154	0.126	0.171	0.066	**0.023**	0.029	SE
GunPoint	0.087	0.087	0.013	0.04	0.06	0.047	0.011	0.027	0.073	**0**	0.073	0	0.02	0.007	RPCD
Haptics	0.63	0.588				0.565	0.488		0.571	0.614		0.532	0.523	**0.481**	COTE
InlineSkate	0.658	0.613				0.675	0.603		0.586	0.68		0.573	0.615	**0.551**	COTE
ItalyPower	0.045	0.045				0.033	0.096		0.044	0.157		**0.031**	0.048	0.036	LTS
Lightning2	0.246	0.131	0.213	**0.1**	0.164	0.18	0.257	0.164	0.131	0.246	0.197	0.177	0.344	0.164	WDTW
Lightning7	0.425	0.288	0.247	0.2	0.233	0.263	0.262	0.466	0.26	0.356	0.438	**0.197**	0.26	0.247	LTS
MALLAT	0.086	0.086				0.072	0.037		0.075			0.046	0.06	**0.036**	COTE
MedicalImages	0.316	0.253				**0.232**	0.269		0.258	0.289		0.271	0.396	0.258	TSF
MoteStrain	0.121	0.134				0.118	0.135		0.205	0.203		0.087	0.109	**0.085**	COTE
NonInvThorax1	0.171	0.185				0.103	0.138						0.1	**0.093**	COTE
NonInvThorax2	0.12	0.129				0.094	0.13						0.097	**0.073**	COTE
OliveOil	0.133	0.167	0.167	0.188	0.167	0.1	**0.09**	0.133	0.167	0.167	0.1	0.56	0.1	0.1	TSBF
OSULeaf	0.483	0.384	0.248	0.372	0.198	0.426	0.329	0.256	0.372	0.355	0.165	0.182	0.285	**0.145**	COTE
Plane	0.038	**0**											**0**	**0**	Tie
SonyAIBORobot	0.305	0.305				0.235	0.175		0.185	0.203		0.103	**0.067**	0.146	SE
SonyAIBORobotII	0.141	0.141				0.177	0.196		0.123	0.157		0.082	0.115	**0.076**	COTE
StarLightCurves	0.151	0.095				0.036	**0.022**		0.066	0.118			0.024	0.031	TSBF
SwedishLeaf	0.213	0.157	0.102	0.138	0.104	0.109	0.075	0.198	0.117	0.098		0.087	0.093	**0.046**	COTE
Symbols	0.1	0.062				0.121	**0.034**		0.059	0.096	0.227	0.036	0.114	0.046	TSBF
SyntheticControl	0.12	0.017	0.023	0.002	0.027	0.023	0.008	0.037	0.027		0.037	0.007	0.017	**0**	COTE
Trace	0.24	0.01	0.05	**0**	0.07	**0**	0.02	**0**	0.01		0.01	**0**	0.02	0.01	Tie
TwoLeadECG	0.253	0.132				0.112	0.046		0.138	0.126		**0.003**	0.004	0.015	LTS
TwoPatterns	0.09	0.0015	0.001	**0**	0.001	0.053	0.001	0.129	0.004		0.074	0.003	0.059	**0**	Tie
UWaveX	0.261	0.227				0.213	**0.164**		0.211	0.379		0.2	0.216	0.196	TSBF
UWaveY	0.338	0.301				0.288	**0.249**		0.278	0.383		0.287	0.303	0.267	TSBF
UWaveZ	0.35	0.322				0.267	**0.217**		0.293	0.407		0.269	0.273	0.265	TSBF
wafer	0.005	0.005	0.004	0.002	0.004	0.047	0.004	0.003	0.006	0.003	**0**	0.004	0.002	0.001	FBL
WordSynonyms	0.382	0.252				0.381	0.302		**0.243**	0.276		0.34	0.403	0.266	CID
yoga	0.17	0.155	0.13	0.165	0.143	0.157	0.149	0.17	0.156	0.134	0.226	0.15	0.195	**0.113**	COTE
# Data Sets	46	46	19	19	19	44	44	19	42	38	20	41	46	46	
# Best	0	1	1	3	0	3	6	2	1	3	2	8	4	22	

standard deviation in the difference of the error between SE and DTWCV is 8.9 percent, indicating that selecting between the techniques or combining predictions could yield significant improvement.

Three other shapelet approaches have been proposed. Logical shapelets [36], fast shapelets [6] and learnt shapelets (LST) [35]. The accompanying website for [6] provides results for logical and fast shapelets on 31 UCR data sets. SE is better than logical on 28 data sets and better than fast shapelets on 26. In both cases, SE is significantly better. Results for 41 data sets are presented on the website associated with learnt shapelets [37]. LST beats SE on 29 data sets, ties on 1 and loses on 11. The reported LST results are significantly better than SE and are clearly very encouraging. We believe LST is a promising approach to shapelet generation that requires further research and validation. The reported LST results are averaged over five runs with parameter tuning on each fold. We used the LST code from [37] to get results for 51 of our 72 data sets for a single run. These 51 were selected purely because of time and memory constraints. We found that LST was better on 25, tied on 2 and was worse on 24. Clearly there is no difference between the techniques on this sample of datasets. we also found that LST was not noticeably faster, and required more memory, than the shapelet transform with the optimizations included. These experiments are not conclusive, but equally

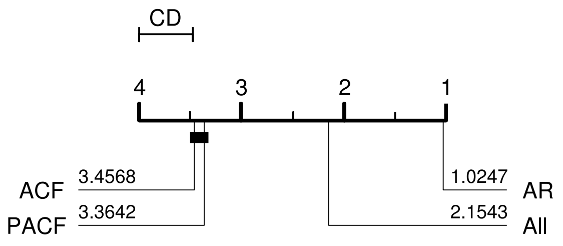


Fig. 3. Critical difference diagram for change based transforms on simulated AR data.

they do not lead us to believe that the LST approach is significantly better than the SE.

6.2 The Change-Based Ensemble

The most common way to use the ACF in time-series data mining is to fit an AR model (i.e. use W with p set to minimize AIC) to each series, then base similarity on differences in model parameters, using, for example, euclidean distance (see, e.g., [38], [39], [40]).

Fitting the AR model provides the most explanatory power, but it does not necessarily capture the best discriminatory features between series. This is because the model selection criteria for the parameter p is fairly crude, and if different series from the same underlying model are modelled with different p values, then the distance between the series will be large. If the data is in fact generated by AR processes for each class, then clearly the feature set W will be optimal. Fig. 3 shows the average ranks of using ACF, AR and PACF features in isolation and in combination with the heterogeneous ensemble described in Section 4.1 on over 2,000 simulated data sets generated by the algorithm presented in [19].

Using just the AR parameters on this data produces significantly better results. The ACF and PACF do not capture the difference in classes, and the redundant features degrade the performance when all features are used together. This would seemingly lend argument to the standard practice of using the AR parameters as features.

However, Fig. 4 shows the same experiment repeated with the 72 data sets we use in later experimentation. The situation is now reversed. Using the AR parameters is significantly worse than the other approaches, and the classifier built on the concatenated feature sets perform the best. Clearly, many of the problems are not suitable for autocorrelation based features. However, some useful information may still be in the autocorrelation function which is not captured by the AR parameters. These experiments lead us to conclude that using the concatenation of ACF, PACF and AR features gives the most robust solution for TSC with change based features.

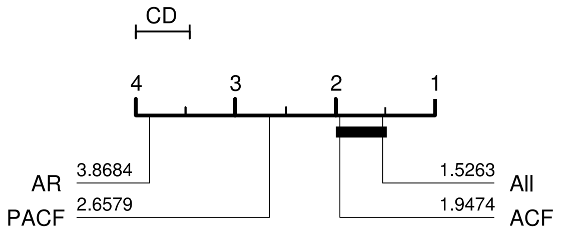


Fig. 4. Critical difference diagram for change based transforms on 72 data sets.

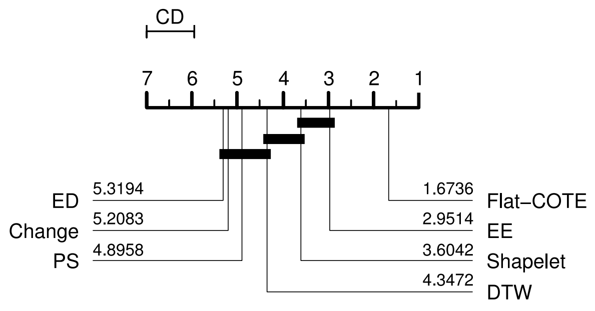


Fig. 5. Critical difference diagram for collective (flat-COTE) and the individual ensembles on the Change domain (Change), the power spectrum, Shapelet Transform (Shapelet) and the time domain Elastic Ensemble. Single classifiers 1-Nearest neighbour with euclidean distance and dynamic time warping distance with warping window set through cross validation (DTW) are included for contrast.

7 Flat-COTE Results

We deploy 35 different classifiers over four data representations. The most obvious ensemble approach is to include all possible classifiers in one ensemble. The flat collective of transform-based ensembles (flat-COTE) weights the vote of each classifier by its cross-validation accuracy on the training data. We compare the accuracy ranks of ensembles constructed on each transform domain, flat-COTE, and, for bench marking, 1-NN with euclidean distance and dynamic time warping with warping window set through cross validation. The mean rank of flat-COTE is significantly higher than all of the other classifiers (tested using the Friedman rank test). Fig. 5 shows the critical difference diagram, as described in [41]. The diagram shows the average ranks of the classifiers. The solid horizontal lines group classifiers into *cliques*, within which there is no significant difference in rank.

In [12], we demonstrate that the elastic ensemble of 1-NN classifiers is significantly more accurate than any one of the component distance measures. Flat-COTE is significantly better than DTWCV (Fig. 6) and EE (Fig. 7). The information provided by the shapelet transform and, to a lesser extent, the power spectrum and change domains, provides discriminatory features that are hard, if not impossible, to detect in the time domain.

This result raises two immediate questions. First, how good is flat-COTE in comparison to other TSC classification algorithms? Second, can we structure the collective so that it uses only the transforms appropriate for the problem domain?

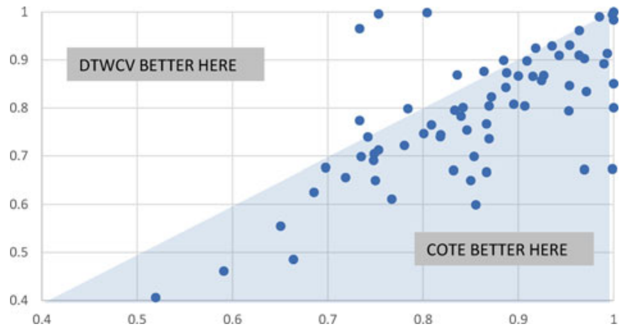


Fig. 6. Scatter plot of test accuracies of DTW (window size set through cross validation) against flat-COTE for all 72 data sets. DTW is better on 10 data sets, flat-COTE better on 60, and they tie on 2.

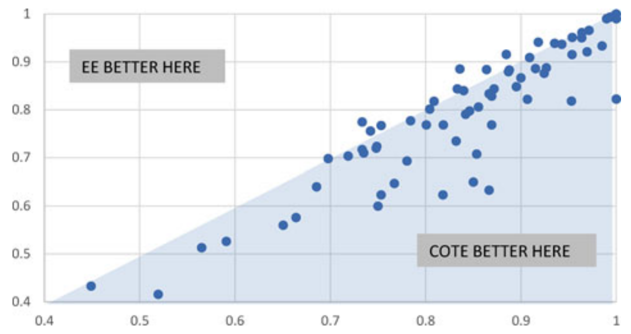


Fig. 7. Scatter plot of test accuracies of Elastic Ensemble [12] against COTE for all 72 data sets. EE is better on 10 data sets, COTE better on 54, and they tie on 8.

7.1 Comparison to Other TSC Algorithms

In Section 2, we described numerous TSC algorithms that have recently been proposed in the literature [2], [3], [5], [7], [8], [9], [21], [22], [36]. We have not as yet implemented these algorithms, but we can compare performance on the published UCR datasets. These are collated in Table 2. All results are rounded to three decimal places, for consistency across publications. Flat-COTE is the most accurate on 22 of the 46 data sets. Many of the differences between classifiers is tiny, but however we look at the data, it is clear that flat-COTE is outperforming the other algorithms. For example, if we restrict our attention to harder problems, where the best accuracy is over 5%, flat-COTE is the most accurate on 16 of 27 data sets (59 percent), and when the best accuracy is over 10 percent, flat-COTE better on 11 out of 19 data sets (58 percent).

Full comparative results are available on a spreadsheet on the accompanying website [31]. A pairwise comparison of each algorithm against flat-COTE is given below. We test for significant difference using the binomial test and the Wilcoxon sign rank test (WSR).

1) The feature based linear (FBL) classifier [22] is evaluated on 19 UCR datasets. Flat-COTE is better on 15 of these, ties on 3, and is worse on 1 (wafer, where COTE is 99.9 percent accurate, FBL 100 percent). Flat-COTE is significantly better at the 1 percent level. The p-values are 0.001 (BT) and 0.01 (WSR).
2) The results for LTS [35] for 41 UCR data sets are presented on the website [?]. Flat-COTE is better on 30 of these, ties on 2, and is worse on 9. Flat-COTE is significantly better at the 1 percent level. The p values are 0.0005 (BT) and 0.0022 (WSR).
3) The TSBF classifier [3] is evaluated on 44 UCR datasets. In comparison to the best version of TSBF (TSBF Rand), flat-Cote wins on 37 datasets and loses on 7. Flat-COTE is significantly better at the 1 percent level. The p-values are 2.65×10^{-6} (BT) and 8.86×10^{-6} (WSR).
4) Two versions of TSF [5], TSF entrance and TSF entropy, are assessed on 44 UCR datasets. In comparison to TSF entrance (the best version of TSF), flat-Cote wins on 35 datasets and loses on 9. Flat-COTE is significantly better at the 1 percent level. The p-values are 5.3×10^{-5} (BT) and 1.65×10^{-5} (WSR).
5) CID [7] is evaluated on 42 UCR datasets (the two Fetal ECG datasets are missing). Flat-COTE is more accurate on 41 and worse on 1. Flat-COTE is significantly better at the 1 percent level. The p-values are 9.78×10^{-12} (BT) and 1.12×10^{-8} (WSR).
6) RPCD [21] present test accuracy on 37 datasets (the UCR datasets without the simulated problems and the two fetal ECG datasets). Flat-Cote wins on 32 datasets, ties on 1, and loses on 4. Flat-COTE is significantly better at the 1 percent level. The p-values are 9.71×10^{-7} (BT) and 2.3×10^{-6} (WSR).
7) The BOP approach [2] is evaluated on 19 UCR datasets. Flat-COTE is better on 16 of these, ties on 1, and is worse on 2. Flat-COTE is significantly better at the 1 percent level. The p-values are 0.0007 (BT) and 0.002 (WSR).
8) TWE [9] with optimised parameters is evaluated on 19 UCR datasets. Flat-COTE is better on 16 of these, ties on 1, and is worse on 2. Flat-COTE is significantly better at the 1 percent level on these 19 datasets. The p values 0.0007 (BT) and 0.002 (WSR). It is also significantly better on all 72 datasets.
9) MSM [4] is evaluated on 19 UCR datasets. Flat-COTE is better on 16 of these, ties on 1, and is worse on 2. Flat-COTE is significantly better at the 1 percent level. The p values 0.0007 (BT) and 0.002 (WSR). It is also significantly better on all 72 datasets.
10) DTW has been used as the standard benchmark algorithm for UCR datasets in the vast majority of TSC research. For the 46 UCR problems flat-COTE is better on 40 of these, ties on 3, and is worse on 3. Over all 72 datasets, flat-COTE is better on 63, worse on 6 and draws on 3.
11) Euclidean distance is also still often used to support new TSC algorithms. Flat-COTE is better on 44 of these, ties on 1, and is worse on 1. Over all 72 datasets, flat-COTE is better on 69, ties on 1 and is very marginally worse on 2.

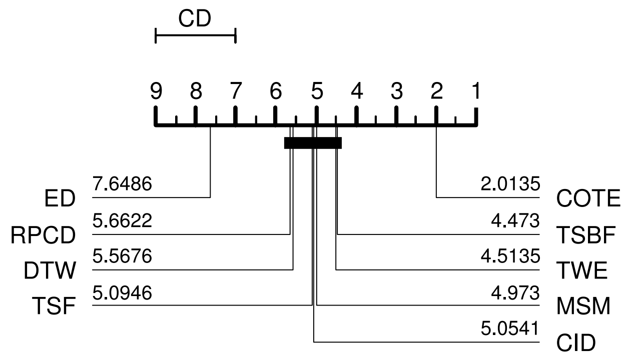


Fig. 8. Critical difference diagram for flat-COTE and the other TSC algorithms on 37 UCR datasets that are common across each paper. BOP and FBL are omitted due to only having results for 19 datasets available. The results for TWE and MSM were taken from [12] as the original work of [9] and [4] also report results on 19 datasets.

A critical difference diagram comparing the results of the flat-COTE to the other TSC algorithms for the UCR datasets is shown in Fig. 8. Flat-COTE is significantly more accurate than all recent algorithms for TSC that have been evaluated on the UCR datasets. To the best of our knowledge, these are the best results ever published on the UCR data.

Like flat-COTE, FBL generates a large set feature to capture alternative discriminatory factors. Given the similarity

TABLE 3
Test Classification Errors for the Two-Class and Five-Class Worm Problems with Five Different Ensembles

Dataset	Flat-COTE	EE	Shapelet	PS	Change
Worms5	0.25	0.38	0.30	0.26	0.19
Worms2	0.18	0.38	0.23	0.19	0.14

between COTE and FBL, it is worth considering why COTE is so much more accurate. We believe the difference is caused by two factors. First, we include shapelet features in our ensemble, and these are not present in FBL. Second, FBL relies on stepwise feature selection with a linear classifier. Such a simple procedure is likely to miss feature interactions that a more complex classifier such as rotation forest can pick up on. The interaction between transform and classifier is more complex than we initially thought, and we consider this in more detail in Section 8.

7.2 Algorithm Efficiency

Accuracy is not the only criteria for assessing TSC algorithms. COTE is a combination of complex transformations and classifiers and is no doubt slower than many of the algorithms it outperforms in terms of accuracy. We acknowledge this weakness but would mitigate it with two observations. First, the ensemble is easy to parallelise because there is no communication required between the components until the ensembling stage. We currently run COTE on a multiprocessor High Performance Computing Cluster and are developing a GPU version. Once parallelised, the ensemble is only as slow as its slowest component. This is undoubtedly the shapelet transform, for which the enumerative search is $O(n^2m^4)$. We take advantage of the shapelet speed ups involving alternative quality measures, early abandon and caching that have been proposed [11], [36]. Furthermore, heuristic search techniques such as that described in [35] offer the potential for speeding up the search without compromising quality. Our second observation is the most important criteria for assessing new TSC algorithms is classification accuracy. The majority of classification problems involve off line analysis where the domain experts would be happy to dedicate processor time to finding a good solution. We would suggest that a new algorithm proposed on the basis of accuracy alone would only be of interest if it is significantly more accurate than DTWCV and not significantly less accurate than COTE.

The flat-COTE approach that we have proposed is very simple to implement. However, the problem with an approach of using one large ensemble is that it gives little insight into the nature of the problem, and does not aid exploratory analysis. We demonstrate this point with our case study on worm motion, before describing a hierarchical classifier based on choosing one or more transform spaces, based on training-set performance.

7.3 Case Study: Classifying Mutant Worms

In Section 5.1 we described two new TSC problems involving classifying mutant worms based on their motion. The test accuracies for ensembles constructed on each representation and the flat-COTE are given in Table 3.

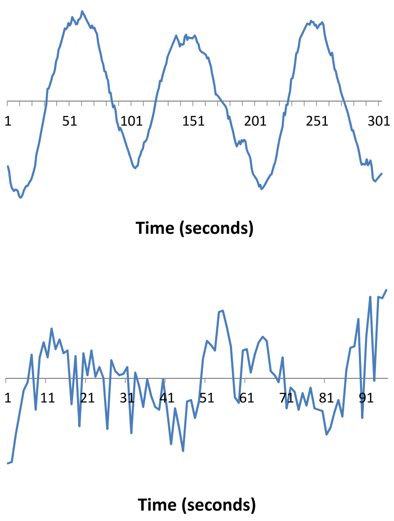


Fig. 9. The best shapelets for wild-type (top) and mutant worms (bottom) for the two class problem.

We observe that the time-domain classifier is the worst of all. This is unsurprising, given the nature of the data, but it is worth mentioning that time-domain classifiers are not always the best approach, given their prevalence in the literature. The flat ensemble is less accurate than the best approach, which is to use the change transform. This emphasises that it is desirable to be able to determine the best transform *a priori*. The shapelet ensemble is less accurate than the power spectrum and change ensembles, but shapelets offer the added bonus of greater explanatory power. Fig. 9 shows the best shapelets for the wild-type and mutant worm classification problem.

The best wild-type shapelet represents highly regular movement, in that the worm cyclically adopts the eigenworm1 shape. The mutant shapelet is much more erratic, with short localised variation from the regular pattern. This explains why the Change transform is the best. The low order ACF terms for the non mutants will be highly discriminatory, because the movement at one time step is highly correlated with the previous time step. This correlation is much weaker with the mutant type.

Fig. 10 shows the best shapelet for each of the five classes (wild type and four mutant classes). We see the same localised variance with the mutants as with the two class problem, but there is also some variability in the degree of deviation from the eigenworm between mutants. This preliminary study has demonstrated that time series classification could provide a useful way of automating what is currently a very labour intensive process, and that the ensemble approach gives very promising results.

7.4 Resampling Experiments

Even when using 72 data sets, relying on a predefined train/test split runs the risk of over-fitting that particular data split. To mitigate against this risk, we repeat our experiments on a subset of 20 dataset using resampling. These datasets were selected because they are the quickest to model and classify. We combine the train and test sets, then perform 50 random samples to form 50 train/test splits with the same train set size as the original train/test split.

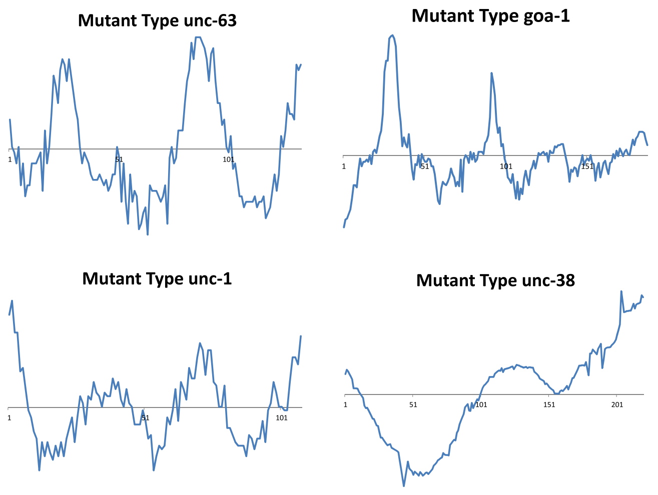


Fig. 10. The best shapelets for the four mutant classes.

Table 4 shows the train/test results for flat-COTE, the mean error for DTW and the mean error for flat-COTE. Flat-COTE is significantly better on 18 of the 20 data sets.

Full results are available from the associated website [31].

Fig. 11 shows the critical difference diagram for the resampled data. Flat-COTE is significantly better than each component ensemble except for the shapelet transform.

8 Alternative Ensemble Structures

The flat structure works well, but is less informative than an ensemble that could provide information about which transformation to use. We also thought that if we could choose the best transform based on training set performance we would be able to improve overall performance. To test this, we investigated several alternative ensemble structures that all involve first forming an ensemble on each transform independently, then combining these transforms to output predictions for the final classification.

TABLE 4
DTW and Flat-COTE Errors from Train/Test Splits and from 50 Resampling Experiments

	DTW		flat-COTE	
Dataset	tr/te	resample	tr/te	resample
ArrowHead	0.840	0.775 ($\pm$ 0.006)	0.840	**0.817 ($\pm$ 0.006)**
Beef	0.633	0.607 ($\pm$ 0.012)	0.867	**0.855 ($\pm$ 0.015)**
BeetleFly	0.600	0.699 ($\pm$ 0.015)	0.750	**0.843 ($\pm$ 0.014)**
BirdChicken	0.650	0.714 ($\pm$ 0.014)	0.850	**0.879 ($\pm$ 0.014)**
CBF	0.998	0.966 ($\pm$ 0.006)	0.999	**0.988 ($\pm$ 0.006)**
Coffee	1.000	0.929 ($\pm$ 0.009)	1.000	**0.994($\pm$ 0.002)**
ECGFiveDays	0.822	0.835 ($\pm$ 0.008)	1.000	**0.974($\pm$ 0.004)**
FaceFour	0.909	0.862 ($\pm$ 0.009)	0.909	**0.917($\pm$ 0.009)**
FacesUCR	0.937	0.912 ($\pm$ 0.002)	0.943	**0.947($\pm$ 0.002)**
GunPoint	0.993	0.952 ($\pm$ 0.004)	0.993	**0.985($\pm$ 0.003)**
ItalyPD	0.961	0.950 ($\pm$ 0.002)	0.964	**0.959($\pm$ 0.001)**
Lightning7	0.767	0.738 ($\pm$ 0.006)	0.753	0.732($\pm$ 0.009)
MoteStrain	0.886	0.853 ($\pm$ 0.005)	0.915	**0.873($\pm$ 0.006)**
OliveOil	0.867	0.871 ($\pm$ 0.008)	0.900	0.885($\pm$ 0.009)
SonyI	0.707	0.878 ($\pm$ 0.008)	0.854	**0.946($\pm$ 0.008)**
SonyII	0.876	0.846 ($\pm$ 0.005)	0.924	**0.930($\pm$ 0.005)**
SynthControl	0.990	0.989 ($\pm$ 0.001)	1.000	**0.996($\pm$ 0.000)**
Toe1	0.921	0.853 ($\pm$ 0.008)	0.969	**0.947($\pm$ 0.002)**
Toe2	0.915	0.884 ($\pm$ 0.005)	0.885	**0.937($\pm$ 0.004)**
TwoLeadECG	0.933	0.837 ($\pm$ 0.010)	0.985	**0.951($\pm$ 0.006)**

Bold indicates that flat-COTE resampling experiment has significantly lower mean error than DTW.

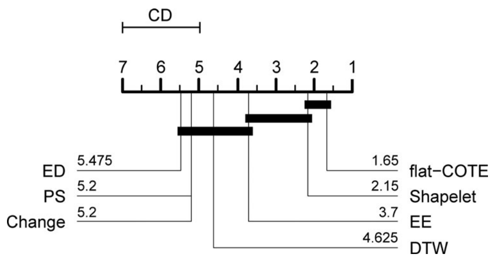


Fig. 11. Critical difference diagram for flat-COTE and the individual ensembles for the mean of 50 resamples on 20 datasets.

8.1 Predicting the Correct Transform Space

Suppose that we could determine which transformation was best beforehand. Surely that would make the overall classifier more accurate? Unfortunately our experiments show that it makes the classifier worse. Table 5 shows the distribution of best transform, as judged solely on the test set accuracy.

Imagine if we were able to perfectly predict this. The oracle classifier would only use the transform that gave the best test set accuracy. The results show that it would not in fact improve overall performance. The oracle-COTE is not significantly better than flat-COTE. Oracle-COTE wins on 27 dataset, flat-COTE on 36 and they tie on 9. Clearly, there is little to choose between them. Even if we can pick the best transform, it does not on average lead to greater accuracy. The result implies that many problems have discriminatory features in more than one domain. However, it may still be desirable to choose a single transform in order to obtain greater explanatory power. The question then arises, how well can we predict the correct transform? One obvious basis for selection would be training set cross validation accuracy. However, if we use the simple decision rule of choosing the transform ensemble that has the highest average individual training accuracy, we are correct just 55 percent of the time and the resulting classifier is significantly worse than flat-COTE. Choosing based on summary statistics such as the max or median fair no better.

Another possibility would be that we could use the problem type as a form of transfer learning in a Bayesian context to help predict the correct transform. Table 6 shows the proportion of each dataset type won by the different transforms. The numbers are small, so we cannot infer too much, but it would seem that the EE over performs on motion problems and that shapelets do disproportionately well on sensor data.

We experimented with a meta-classifier over the 72 datasets that used a range of training features such as component classifier ranks, accuracies and dataset characteristics, but our best accuracy for predicting the correct transform

TABLE 5
Frequency of Test Set Wins by Transform

Transform	Number of datassets
Elastic Ensemble	34
Shapelets	22
Power Spectrum	9
Change	7

TABLE 6
Percentages to Show Where Each Transform Space Produced the Lowest Error Rates, Broken Down by Problem Type

Dataset Type	EE	Shapelets	PS	Change
Human Sensor	20%	80%	0%	0%
Image	46%	21%	18%	14%
Motion	73%	27%	0%	0%
Sensor	35%	39%	17%	9%

was little more than 60 percent, and the overall accuracy of the resulting COTE classifier was significantly worse than flat-COTE. There is simply too much noise in the training set accuracy estimates for these datasets.

8.2 Weighting Each Transform

If we cannot (or do not want to) pick the best transform, the next logical step is to introduce a hierarchical collective with weighting for each transform ensemble. In the flat ensemble, each classifier is weighted by its cross validation accuracy on the training set. It would seem sensible then to introduce a hierarchy of ensembles that also weights each transform by a cross validation accuracy. However, the problem with this approach is that it requires a further level of cross validation, which for the shapelet transform and elastic ensemble in particular introduces an unacceptable time overhead. To estimate the error accurately we would have to perform the shapelet transform independently on each fold. For the elastic ensemble, we would have to estimate the parameters for each distance measure independently on each fold. Alternatively, we could weight by some summary statistic of the constituent members of each transform ensemble. We have tried a range of statistics, such as equal weighting, mean, median and the mean weighted by variance, but they are all significantly worse than flat-COTE (see Fig. 12).

8.3 Select a Subset of Transforms

Selecting a single transform discards useful information, and weighting is difficult because of the problem of finding an unbiased estimate of transform utility. Nevertheless, with many problems, certain transforms are clearly inappropriate and are likely to reduce the overall efficiency of the collective. Therefore it is desirable to have a mechanism that is able to select a subset of possible transforms based on within-transform classifier variation. We phrase the choice of whether to include a transform as a hypothesis test. If there is strong evidence that the median classifier accuracy on one transform is worse than that of the best transform, then we discard it. To test this hypothesis, for each dataset we select the best transform on the training data, then perform a two sample Mann-Whitney rank sum test at the 1 percent level against every other transform, where each sample consists of the training accuracy of the constituent classifiers. The selection process can thus choose 1, 2, 3 or 4 transforms to use. The resultant classifier, which we call Mann-COTE, wins on 25 dataset, flat-COTE on 23 and they tie on 24. There is no significant difference between them at the 1 percent level. Mann-COTE most frequently selects two transformations, and there is no apparent bias in performance against number of transforms selected (see Table 7).

TABLE 7
Frequency of Test Set Wins for Mann-COTE versus Flat-COTE, According to the Number of Transforms Selected for a Given Dataset by Mann-COTE

	Flat Wins	Tie	Mann Wins	Total
One transform	9	3	7	19
Two transforms	12	4	13	29
Three transforms	2	5	5	12
Four transforms	0	12	0	12
Total	23	24	25	72

Mann-COTE retains the accuracy of the flat-COTE but offers the possibility of greater exploratory power and likelihood of removing unsuitable data representations. With a larger population in each ensemble we would expect the performance of Mann-COTE to equal, and possibly exceed, the flat collective.

9 Conclusions and Future Work

We have proposed an ensemble scheme for TSC based on constructing classifiers on different data representations. The standard baseline algorithms used in TSC research are 1-NN with euclidean distance and/or dynamic time warping. We have conclusively shown that COTE significantly out-performs both of these approaches. We have shown it to be significantly better than all of the competing algorithms that have been proposed in the literature. We believe the results we present represent a new state of the art against which new TSC algorithms should be compared in terms of accuracy. Of course, accuracy is not the only criteria for assessing a classification algorithm. It is perfectly valid to propose algorithms that offer speed up or greater explanatory power, but no accuracy gains.

This result supports our belief that the best way to form better time series classifiers is to separate the data representation from the classification [11], and that the greatest improvement can be found through choice of data transformation, rather than classification algorithm [10]. However, further analysis of the performance of COTE variants shows that this is not as clear cut as we believed. Our expectation was that if we could choose the right transformation from the training data we would improve the overall performance. This was not the case even if we cheated and picked the best transformation on the test data. We think this is caused by two factors. First, it is apparent that problems can have discriminatory features in multiple representations. This is understandable, particularly with multi-class problems. Second, we were downplaying the importance of the

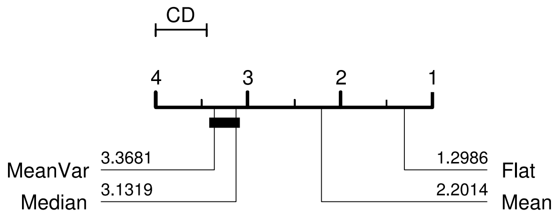


Fig. 12. Flat-COTE against alternative hierarchical weighted collectives. The flat scheme is significantly more accurate than weighting the ensemble according to any of the simple summary statistics that were tried.

classifiers. Algorithms such as rotation forest have very strong internal mechanisms for dealing with deceptive and/or redundant features. The interaction between classifier and transformation is more subtle than we supposed and is worthy of further investigation. This conclusion is supported by the fact that COTE is significantly more accurate than FBL, an algorithm that uses a massive feature space with a simple classifier.

There are several ways that we could improve the collective: Alternative transforms based on frequency counts, interval statistics, and complexity measures could all be assimilated into the collective at a future point, if they are found to add diversity; We could improve the existing transforms. In speech processing and related fields, it is common to use a spectral window, rather than transform the whole series. This offers the possibility of detecting localised discriminatory frequency features, which may be useful for long series classification problems; Our choice of classifiers in the heterogeneous ensemble is fairly arbitrary, and the inclusion of more complex classifiers, the exclusion of weaker classifiers, and the setting of parameters through cross validation might significantly improve overall performance; adapting hyper parameters of the data sets may improve the transform selection process.

Our research is incremental. Our primary concern is finding the best way to approach TSC problems, not to come up with the most ingenious and complex algorithm. We have proposed two novel ways of using shapelets and the ACF, but essentially we build on existing research by combining classifiers and representations that have been previously proposed in the literature. In our search for the best approach, we are classifier and representation neutral. If an algorithm can be shown to have value for some problem type, then we will absorb it into the collective. Our priority is in the accurate assessment and comparison of TSC algorithms to give guidance to those with real world TSC problems. If accuracy is the primary concern and the necessary computing resources are available, our advice to any practitioner is that ensembles over different data representations are the best approach to TSC and that COTE is on average the most accurate algorithm currently available. Our code and results can all be downloaded from [31].

Acknowledgments

This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) [grant number EP/M015087/1]. 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. Anthony Bagnall is the corresponding author.

References

[1] E. Keogh and T. Folias, The UCR time series data mining archive. (2015). [Online]. Available: http://www.cs.ucr.edu/ eamonn/TSDMA/

[2] J. Lin, R. Khade, and Y. Li, "Rotation-invariant similarity in time series using bag-of-patterns representation," *J. Intell. Inf. Syst.*, vol. 39, no. 2, pp. 287–315, 2012.

[3] M. Baydogan, G. Runger, and E. Tuv, "A bag-of-features framework to classify time series," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 11, pp. 2796–2802, Jun. 2013.

[4] A. Stefan, V. Athitsos, and G. Das, "The move-split-merge metric for time series," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 6, pp. 1425–1438, Jun. 2013.

[5] H. Deng, G. Runger, E. Tuv, and M. Vladimir, "A time series forest for classification and feature extraction," *Inf. Sci.*, vol. 239, pp. 142–153, 2013.

[6] T. Rakthanmanon and E. Keogh, "Fast-shapelets: A fast algorithm for discovering robust time series shapelets," in *Proc. 13th SDM*, 2013, pp. 668–676.

[7] G. Batista, X. Wang, and E. Keogh, "A complexity-invariant distance measure for time series," *Data Mining Knowl. Discovery*, vol. 28, no. 3, pp. 634–669, 2013.

[8] Y. Jeong, M. Jeong, and O. Omitaomu, "Weighted dynamic time warping for time series classification," *Pattern Recognit.*, vol. 44, pp. 2231–2240, 2011.

[9] P. Marteau, "Time warp edit distance with stiffness adjustment for time series matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 306–318, Feb. 2009.

[10] A. Bagnall, L. Davis, J. Hills, and J. Lines, "Transformation based ensembles for time series classification," in *Proc. 12th SDM*, 2012, vol. 12, pp. 307–318.

[11] J. Hills, J. Lines, E. Baranauskas, J. Mapp, and A. Bagnall, "Classification of time series by shapelet transformation," *Data Mining Knowl. Discovery*, vol. 28, pp. 851–881, 2014.

[12] J. Lines and A. Bagnall, "Time series classification with ensembles of elastic distance measures," *Data Mining Knowl. Discovery*, vol. 29, no. 3, pp. 565–592, 2015.

[13] L. Ye and E. Keogh, "Time series shapelets: A new primitive for data mining," in *Proc. 15th ACM Int. Conf. Knowl. Discovery Data Mining*, 2009, pp. 947–956.

[14] C. Ratanamahatana and E. Keogh, "Three myths about dynamic time warping," in *Proc. 10th SDM*, 2005, pp. 506–510.

[15] 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," *Proc. 34th VLDB Endowment*, vol. 1, pp. 1542–1552, 2008.

[16] L. Ye and E. Keogh, "Time series shapelets: A novel technique that allows accurate, interpretable and fast classification," *Data Mining Knowl. Discovery*, vol. 22, no. 1/2, pp. 149–182, 2011.

[17] J. Lin, E. J. Keogh, L. Wei, and S. Lonardi, "Experiencing SAX: A novel symbolic representation of time series," *Data Mining Knowl. Discovery*, vol. 15, no. 2, pp. 107–144, 2007.

[18] A. Bagnall and G. Janacek, "Clustering time series from ARMA models with clipped data," in *Proc. 10th ACM Int. Conf. Knowl. Discovery Data Mining*, 2004, pp. 49–58.

[19] A. Bagnall and G. Janacek, "A run length transformation for discriminating between auto regressive time series," *J. Classification*, vol. 31, pp. 154–178, 2014.

[20] J. Caiado, N. Crato, and D. Pena, "A periodogram-based metric for time series classification," *Comput. Statist. Data Anal.*, vol. 50, pp. 2668–2684, 2006.

[21] G. B. D. Silva, V. de Souza, "Time series classification using compression distance of recurrence plots," in *Proc. IEEE 13th Int. Conf. Data Mining*, 2013, pp. 687–696.

[22] B. Fulcher and N. Jones, "Highly comparative feature-based time-series classification," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 12, pp. 3026–3037, 2014.

[23] J. Lines, L. Davis, J. Hills, and A. Bagnall, "A shapelet transform for time series classification," in *Proc. 18th ACM Int. Conf. Knowl. Discovery Data Mining*, 2012, pp. 289–297.

[24] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. Witten, "The WEKA data mining software: an update," *ACM SIGKDD Explorations Newslett.*, vol. 11, no. 1, pp. 10–18, 2009.

[25] J. R. Quinlan, *C4. 5: Programs for Machine Learning*, vol. 1. San Mateo, CA, USA: Morgan Kaufmann, 1993.

[26] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[27] L. Breiman, "Random forests," *Mach. Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[28] J. Rodriguez, L. Kuncheva, and C. Alonso, "Rotation forest: A new classifier ensemble method," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 10, pp. 1619–1630, Oct. 2006.

[29] L. Chen, M. T. Özsu, and V. Oria, "Robust and fast similarity search for moving object trajectories," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2005, pp. 491–502.

[30] A. Bagnall and J. Lines, "An experimental evaluation of nearest neighbour time series classification," Dept. Comput. Sci., Univ. East Anglia, Norwich, United Kingdom, Tech. Rep. CMP-C14-01, 2014.

[31] A. Bagnall, Time series classification website. (2015). [Online]. Available: http://www.uea.ac.uk/computing/tsc

[32] A. Brown, E. Yemini, L. Grundy, T. Jucikas, and W. Schafer, "A dictionary of behavioral motifs reveals clusters of genes affecting *caenorhabditis elegans* locomotion," *Proc. Nat. Acad. Sci. United States Am.*, vol. 10, no. 2, pp. 791–796, 2013.

[33] E. Yemini, T. Jucikas, L. Grundy, A. Brown, and W. Schafer, "A database of *caenorhabditis elegans* behavioral phenotypes," *Nature Meth.*, vol. 10, pp. 877–879, 2013.

[34] T. Jucikas, A. Brown, and B. Bentle, C. elegans behavioural database. (2015). [Online]. Available: http://wormbehavior.mrc-lmb.cam.ac.uk/

[35] J. Grabocka, N. Schilling, M. Wistuba, and L. Schmidt-Thieme, "Learning time-series shapelets," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 392–401.

[36] A. Mueen, E. Keogh, and N. Young, "Logical-shapelets: An expressive primitive for time series classification," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2011, pp. 1154–1162.

[37] J. Grabocka, Learning time-series shapelets. (2015). [Online]. Available: http://fs.ismll.de/publicspace/LearningShapelets/

[38] M. Corduas and D. Piccolo, "Time series clustering and classification by the autoregressive metric," *Comput. Statist. Data Anal.*, vol. 52, pp. 1860–1872, 2008.

[39] A. Bagnall and G. Janacek, "Clustering time series from ARMA models with clipped data," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2004, pp. 49–58.

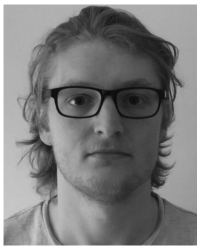
[40] K. Kalpakis, D. Gada, and V. Puttagunta, "Distance measures for effective clustering of ARIMA time-series," in *Proc. Int. Conf. Data Mining*, 2001, pp. 273–280.

[41] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learning Res.*, vol. 7, pp. 1–30, 2006.

Anthony Bagnall has been researching various aspects of data mining and agent-based modeling since 1994 as a PhD student and then a member of faculty in the University of East Anglia, Norwich, United Kingdom, where he is a senior lecturer in the School of Computing Sciences. His current primary research area is time series classification. He is a lifelong supporter of Arsenal Football Club and not really as young as he looks in this photograph.

Jason Lines has recently completed his PhD titled "Time Series Classification through Transformation and Ensembles" from the University of East Anglia. He is currently a senior research Assistant on an EPSRC funded project.

Jon Hills received the PhD degree in philosophy in 2010. He received distinction in MSc conversion course in computer science in 2011. He embarked on another PhD degree in the field of data mining at the University of East Anglia. He has recently completed his thesis titled "Mining Time-series Data using Discriminative Subsequences" and is currently a data scientist for Aviva.



Aaron Bostrom graduated in 2012 with first class honors in computer science prior to joined as a professional programmer in the games industry. He began his PhD degree in the field of time series classification using shapelets at the University of East Anglia in 2014.