HIVE-COTE: The Hierarchical Vote Collective of Transformation-based Ensembles for Time Series Classification

Jason Lines
University of East Anglia
Norwich, United Kingdom
j.lines@uea.ac.uk

Sarah Taylor
University of East Anglia
Norwich, United Kingdom
s.l.taylor@uea.ac.uk

Anthony Bagnall
University of East Anglia
Norwich, United Kingdom
ajb@uea.ac.uk

***Abstract*—There have been many new algorithms proposed over the last five years for solving time series classification (TSC) problems. A recent experimental comparison of the leading TSC algorithms has demonstrated that one approach is significantly more accurate than all others over 85 datasets. That approach, the Flat Collective of Transformation-based Ensembles (Flat-COTE), achieves superior accuracy through combining predictions of 35 individual classifiers built on four representations of the data into a flat hierarchy. Outside of TSC, deep learning approaches such as convolutional neural networks (CNN) have seen a recent surge in popularity and are now state of the art in many fields. An obvious question is whether CNNs could be equally transformative in the field of TSC. To test this, we implement a common CNN structure and compare performance to Flat-COTE and a recently proposed time series-specific CNN implementation. We find that Flat-COTE is significantly more accurate than both deep learning approaches on 85 datasets.**

These results are impressive, but Flat-COTE is not without deficiencies. We improve the collective by adding new components and proposing a modular hierarchical structure with a probabilistic voting scheme that allows us to encapsulate the classifiers built on each transformation. We add two new modules representing dictionary and interval-based classifiers, and significantly improve upon the existing frequency domain classifiers with a novel spectral ensemble. The resulting classifier, the Hierarchical Vote Collective of Transformation-based Ensembles (HIVE-COTE) is significantly more accurate than Flat-COTE and represents a new state of the art for TSC. HIVE-COTE captures more sources of possible discriminatory features in time series and has a more modular, intuitive structure.

I. Introduction

Time series classification (TSC) problems arise across a rich and diverse range of domains. We may consider any ordered data to be a time series, which allows the definition to encompass data from various fields including, but not limited to, finance, biology, medicine, and engineering. The diversity of such data is easily apparent when visiting the University of California, Riverside/University of East Anglia (UCR/UEA) time series classification repository [1][1]. The UCR/UEA datasets consist of 85 varied and freely available problems that are used throughout the TSC literature by many researchers.

[1]www.timeseriesclassification.com

Given the ubiquitous nature and easy availability of data, many researchers have proposed algorithms for solving TSC problems. The greatest research emphasis has been focused on classifying problems in the time domain through using the raw series, typically by defining new elastic distance measures to couple with nearest neighbour classifiers [2], [3], [4]. However, other approaches have also been proposed, including dictionary and interval-based techniques [5], [6], [7], [8], ensemble algorithms [9], [10], and transformation-based approaches [11], [12], [13], [14].

With the wealth of solutions that one could choose from when attempting a new TSC problem, it raises the question of which technique(s) should be considered? A recent empirical evaluation was carried out in [1] where 20 published TSC algorithms were implemented and tested. Experimentation was extensive; each algorithm was tested with 100 different resamples of the 85 UCR/UEA datasets, producing one of the largest ever studies in machine learning with approximately 35 million individual experiments being completed. The results of this study showed that, while there are many competitive TSC algorithms with their own merits, one approach significantly outperformed all other algorithms in terms of classification accuracy. This approach, the Collective of Transformation-based Ensembles (COTE) [10], combines classifiers built on four alternate representations of TSC problems, where the most effective ensembling strategy was found to combine all classifiers into a flat hierarchy (Flat-COTE).

While this study evaluated the leading TSC algorithms published in the literature, it did not include any deep learning methods. Deep learning approaches have seen a recent surge in popularity in other fields, with convolutional neural networks (CNN) in particular garnering state-of-the-art results across tasks such as image processing, natural language processing, and speech recognition [15], [16], [17]. It is therefore only natural to ask whether CNNs could have such an impact on the field of TSC.

Our work seeks to address two questions: first, is Flat-COTE more accurate than deep learning approaches for TSC? We implement a CNN using a common framework and conduct experiments on 85 datasets. We also consider recently published results on 44 datasets from a TSC-specific CNN implementation [18]. We demonstrate that Flat-COTE is significantly better than both deep learning approaches. However, despite

its impressive performance Flat-COTE has certain deficiencies. This leads to our second question; can we improve on the structure used in Flat-COTE to make a better classifier? We address this question by defining a new hierarchical probabilistic voting structure, defining a new spectral ensemble classifier, and assimilating classifiers from two further data representations. The resulting classifier, the Hierarchical Vote Collective of Transformation-based Ensembles (HIVE-COTE), contains modules that capture similarity with whole series measures (Elastic Ensemble (EE) [9]), phase independent subseries (Shapelet Transform ensemble (ST) [11]), an interval based ensemble (Time Series Forest (TSF) [6]), a dictionary ensemble (Bag-of-SFA-Symbols (BOSS) [5]), and the new spectral ensemble. HIVE-COTE captures more sources of possible discriminatory features in time series and has a more modular, intuitive structure. More importantly, HIVE-COTE is significantly more accurate than Flat-COTE and represents a new state of the art for TSC. All of our code and data is available from a public code repository and accompanying website[2].

To summarise, our contributions are as follows:

1) We evaluate two deep learning solutions for TSC: a standard CNN and a bespoke CNN for TSC. We demonstrate that the standard approach is no better than dynamic time warping, and both are significantly less accurate than the current state of the art.
2) We propose a new probabilistic hierarchical structure that is modular, encapsulating predictions from different representations into a single vote.
3) We define a new spectral-based classifier that is significantly more accurate than the alternative spectral methods.
4) We assimilate three new classifiers into the collective and create a new classifier, HIVE-COTE.

II. Time Series Classification Background

1) Whole series: Whole series techniques compare two series either as a vector (as with traditional classification) or by a distance measure that uses all data points. Most research effort has been directed at finding techniques that can compensate for small misalignments between series using **elastic** distance measures. The almost universal benchmark for whole series measures is Dynamic Time Warping (DTW) but numerous alternatives have been proposed. These involve alternative warping criteria [3], using versions of edit distance [2], [4] and transforming to use first order differences [13], [19].

2) Intervals: Rather than use the whole series, the interval class of algorithm select one or more phase-dependent intervals of the series. At its simplest, this involves a feature selection of a contiguous subset of attributes. However, the three most effective techniques generate multiple intervals, each of which is processed and forms the basis of a member of an ensemble classifier [6], [20], [21].

3) Shapelets: Shapelet [8] approaches are a family of algorithms that focus on finding short patterns that define a class and can appear anywhere in the series. A class is distinguished by the presence or absence of one or more shapelets somewhere in the whole series. Two common ways of finding shapelets are through enumerating the candidate shapelets in the training set [11] or searching the space of all possible shapelets with a form of gradient descent [22].

4) Dictionary-based: Some problems are distinguished by the frequency of repetition of subseries, rather than by their presence or absence. Dictionary-based methods form frequency counts of recurring patterns, then build classifiers based on the resulting histograms [23], [5].

5) Spectral: The frequency domain will often contain discriminatory information that is hard to detect in the time domain. Methods include constructing an autoregressive model [24], [25] or combinations of autocorrelation, partial autocorrelation and autoregressive features [10].

6) Combinations of the previous: Two or more of the above approaches can be combined into a single classifier. For example, concatenating different feature spaces [14], forward selection of features for a linear classifier [26], and transformation into a feature space that represents each group and ensembling classifiers together [10].

Results from a recent experimental evaluation of the leading TSC algorithms in these groups [1] are summarised in the critical difference (CD) diagram in Figure 1. CD diagrams were introduced in [27]. They show the average ranks of multiple classifiers over multiple datasets and summarise a significance test between the ranks. The horizontal black bars are *cliques*; if two classifiers are in the same clique, their ranks are not significantly different. If they are not in the same clique, they are significantly different. The main conclusion from [1] is that Flat-COTE is significantly more accurate than all the other classifiers evaluated, representing the current state of the art for TSC.

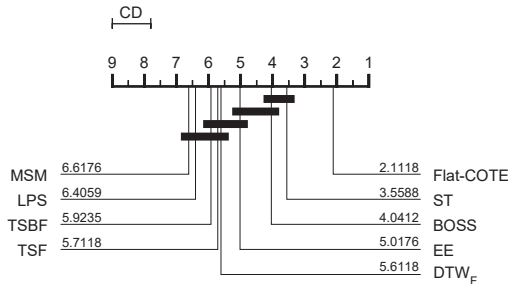

Fig. 1. The top 9 classifiers in [1]: Flat-COTE, ST, BOSS, EE, DTWF (DTW Features), TSF, TSBF (Time Series Bag-of-features), LPS (Learned Pattern Similarity), and MSM (Move-Split-Merge).

A. Flat-COTE

Flat-COTE combines 35 classifiers into a single ensemble. It consists of 11 whole series classifiers (EE), 8 shapelet classifiers (ST), and 16 spectral classifiers (8 built on autocorrelation features, 8 using the power spectrum). Each classifier is built independently and produces separate training accuracies. Given a test instance, each individual classifier outputs a single class prediction and weights it by training accuracy. Weighted test predictions are then pooled, and the class with the highest combined vote is the predicted class by Flat-COTE. The generic proportional ensemble scheme is outlined in Algorithm 1.

[2]www.timeseriesclassification.com/icdm2016.php

Algorithm 1 ProportionalEnsemble(*classifiers*, *train*, *test*)

```
1: trainAccs = ∅;
2: for i ← 1 to |classifiers| do
3:     trainAccs_i = loocv(train, classifiers[i])
4:     classifiers_i.buildClassifier(train)
5: testPreds = ∅
6: for i ← 1 to |test| do
7:     votes = ∅
8:     bsfWeight = −1;
9:     bsfClass = −1;
10:    for c ← 1 to |classifiers| do
11:        p = classifiers_c.classify(test_i)
12:        votes_p = votes_p + trainAccs_c;
13:        if votes_p > bsfWeight then
14:            bsfWeight = votes_p
15:            bsfClass = p
16:        testPreds_i = bsfClass
17: return testPreds
```

The results in [1] demonstrated that Flat-COTE is significantly more accurate than any of the other algorithms that were evaluated, but the conclusions give rise to several questions:

1) Could a classifier from a different area of machine learning do better? The evaluation in [1] considered many such approaches, but an obvious omission were deep learning algorithms.
2) Does the flat structure cause a lack of robustness? For example, the EE component contains 11 classifiers while the other representations only have 8 each. This gives EE a higher weight in Flat-COTE. Also, EE contains full DTW and windowed DTW; in cases where the optimal window is 100%, these classifiers will be identical but have two votes. Conversely, if we included a tree-based ensemble (such as TSF) with 500 classifiers, do we give it one compound vote, or 500 individual votes? Either is undesirable.
3) The ACF and PS classifiers make up almost 50% of Flat-COTE, yet the features are deterministically linked and derived from the whole series.
4) Flat-COTE only contains classifiers from three of the five groups that we identified in Section II.

In Section III we define HIVE-COTE, a modular collective with a new hierarchical structure that includes a novel spectral ensemble and two new modules derived from other published algorithms. We then compare against Flat-COTE and deep learning algorithms in Section VI.

III. A New Collective: HIVE-COTE

We introduce a new version of COTE that we call the Hierarchical Vote Collective of Transformation-based Ensembles (HIVE-COTE). HIVE-COTE is an improved version of Flat-COTE that uses a modular hierarchical meta-ensemble structure. Given a problem where all classifiers across the four domains achieve similar training accuracies, Flat-COTE will be biased towards time domain classifiers over other domains simply because more classifiers are built in the time domain. HIVE-COTE overcomes this potential design bias by modularising the elements of each group of classifiers. It allows only a single probabilistic prediction from each domain (whole series; interval; shapelet; dictionary; and spectral). The components of a module (ensemble of classifiers on a certain type) then becomes an encapsulated design decision. From the top level, it does not matter if a module contains one classifier or five hundred. The overseer simply defines how to combine module predictions into a single overall estimate.

A. Hierarchical Voting Structure

More formally, suppose we have g modules for a problem with C classes, where $|C| = c$. Each module produces an estimate of the probability of the class variable y, $p_j(y = i)$ for $j = 1 \dots g$ and $i = 1 \dots c$. Furthermore, each module has a cross-validation weight w_j. Flat-COTE treats each classifier as a single module, performing a weighted vote with all modules to find the class value with the greatest weight. HIVE-COTE differs as it instead treats each constituent ensemble as a module. Each module outputs a probability estimate for each class, weighting estimates proportionally to training accuracy. These estimates are combined in a second layer to create a meta-ensemble, where the class value with the greatest weight across all modules is output as the prediction. This creates a more balanced and intuitive ensemble as each module corresponds to a different base ensemble, where each ensemble has an equal starting weight that is then weighted by training accuracy.

B. HIVE-COTE Modules

HIVE-COTE contains five modules: two modules from Flat-COTE for whole series and shapelets, two modules from other published research for interval and dictionary-based similarity, and one new module for spectral features.

1) Elastic Ensemble (EE) [9]: EE combines 1-nearest neighbour (1-NN) classifiers using various whole-series measures. The majority of research emphasis in TSC has been placed on defining similarity measures to couple with 1-NN classifiers. Given the wide choice in measures that could be used, a preliminary experiment in [9] showed that there was no similarity measure that significantly outperformed all others when coupled with 1-NN classifiers. However, it demonstrated that the classifiers did make predictions in significantly different ways. The EE was created to utilise this diversity by building 1-NN classifiers with these measures to combine into a proportional ensemble (as in Algorithm 1), which was significantly more accurate than any of its constituent parts. All parameter settings for the measures are set through cross-validation on the training data.

2) Shapelet Transform Ensemble (ST) [11]: ST separates shapelet discovery from the classifier by finding the top k shapelets on a single run. Data are transformed using the shapelets, where attributes in a new instance are the distances from an input series to each shapelet. We use the most recent version of ST [28] that balances the number of shapelets per class and evaluates each shapelet on how well it discriminates a single class. Following [10], [28] we construct a classifier from this dataset using a weighted ensemble of standard classifiers (as in Algorithm 1). We include k Nearest Neighbour (where k is set through cross-validation), Naive Bayes, C4.5 decision tree, Support Vector Machines (linear and quadratic kernels), Random Forest (with 500 trees), Rotation Forest (with 50 trees), and a Bayesian network. Each classifier is assigned a weight proportional to cross-validation training accuracy, and new data (after transformation) are classified with a weighted vote. With the exception of k-NN, we do not optimise parameter settings for these classifiers via cross-validation.

3) Bag-of-SFA-Symbols (BOSS) Ensemble [5]: Shapelet algorithms look for subseries patterns that identify a class through presence or absence. However, if a class is defined by the frequency of a pattern then shapelet approaches will be poor. Dictionary approaches address this by forming frequency counts of repeated patterns. They approximate and reduce the dimensionality of series by transforming into representative words, then compute similarity by comparing the distribution of words. The core process involves forming words by passing a sliding window of length w over each series, approximating each window to produce l values, then discretising these values by assigning each a symbol from an alphabet of size α. BOSS uses a truncated discrete Fourier transform to compress each window, then discretises through multiple coefficient binning. The resulting distribution of words forms the basis for 1-NN classification and uses a bespoke non-symmetrical distance function. BOSS also includes a parameter that determines whether the subseries are normalised or not. During the parameter search of window sizes, the BOSS ensemble retains all classifiers with training accuracy within 92% of the best. New instances are classified by a majority vote.

4) Time Series Forest (TSF) [6]: TSF overcomes the problem of the huge interval feature space by employing a random forest approach, using summary statistics of each interval as features. Training a single tree involves selecting $\sqrt{m}$ random intervals, generating the mean, standard deviation, and slope of the random intervals for every series. Trees are trained on the resulting $3\sqrt{m}$ features and classification is by majority vote.

5) Random Interval Features (RIF): The spectral component of Flat-COTE contains 8 classifiers built on autocorrelation features (ACF ensemble) and 8 classifiers trained using the power spectrum (PS ensemble). The ACF features involve concatenation of autocorrelation, partial autocorrelation, and autoregressive terms, and the PS terms are the truncated periodogram (squared Fourier terms). However, [1] showed that these two ensembles were significantly worse than the other components of Flat-COTE (EE and ST). As these ensembles contribute 16 of the 35 total constituent classifiers in Flat-COTE it would be expected that including these classifiers will negatively affect Flat-COTE on problems where discriminatory features are not in the spectral domain. To overcome this we propose a new classifier: the Random Interval Feature (RIF) Ensemble. RIF draws ideas from the interval feature classifier TSF [6] and we also construct a random forest classifier. However, instead of using time domain intervals, we use intervals from the data transformed into alternate representations. We create two versions: RIF_ACF for autocorrelation-transformed data and RIF_PS for the power spectrum-transformed data. We omit full results for brevity, but detailed analysis can be found on the accompanying website. Both RIF classifiers are on average 4% better than their Flat-COTE counterparts and the difference is significant. RIF_ACF beats ACF on 58 of the 85 datasets, RIF_PS wins on 68 over PS.

While both RIF variants are improvements over the original spectral components, we must be careful not to repeat the issue in Flat-COTE by placing too much weight on spectral domains. As the ACF and PS transforms are deterministically linked it makes sense to have a single module in HIVE-COTE to represent spectral methods. We can either combine RIF_ACF and RIF_PS into a single module or use only one. For simplicity we select the second option; we include RIF_ACF and omit RIF_PS. There is little to choose from between the two classifiers in terms of performance, but the ACF transform is built by concatenating ACF, partial-ACF, and autoregressive model terms. Hence this combined approach likely includes more information than only transforming into the power spectrum.

IV. Datasets

A. UCR/UEA Time Series Dataset Repository

We use the 85 datasets from the UCR/UEA repository. These have been commonly adopted by TSC researchers and the datasets are split into pre-defined train/test partitions to allow reproducible research. However, as discussed in [1], always using the same split risks overfitting on a single sample. As we are focused on the relative performance of classifiers, we adopt the same methodology as [1]: we resample each dataset 100 times and report the average accuracies over 100 folds for a dataset. We seed the resample so that it can be reproduced exactly. All code and data is available from the accompanying website, including a new ethanol level problem that we introduce as a case study for HIVE-COTE.

B. Ethanol Level

Up to 25% of licensed premises in some parts of the UK have been found to have counterfeit alcohol for sale. Brown-Forman, the company that makes Jack Daniels, estimates that around 30% of all alcohol in China is fake. This is a health risk to the consumer as illegally produced spirits may contain contaminants such as methanol, and an economic risk due to the avoidance of taxes. Forgeries can sometimes be detected through external appearance such as poor labelling, but currently there is no way to conclusively tell whether spirits are forged without opening the bottle. However, the alcohol level of genuine spirits is tightly controlled and must equal the level stated on the bottle, but forgeries generally do not have this level of quality control. This means one way of detecting forgeries is by measuring the level of alcohol. Currently, this can only be done by taking a sample and is not feasible for widespread screening. We are investigating non-intrusive ways of testing the alcohol level using spectroscopy. We have conducted experiments using 20 different bottle types and four levels of alcohol: 35%, 38%, 40%, and 45%. Each series is a a spectrograph of 1751 observations. To avoid experimental bias, we evaluate classifiers using a leave-one-bottle-out cross-validation.

V. A Deep Learning Benchmark for TSC

Convolutional neural networks (CNN) are a type of feed-forward artificial neural network with one or more convolutional layers containing learnable filters. A typical CNN has one or more convolutional/max pooling layer pairs followed by one or more fully connected layers, and finally a softmax layer. CNNs are powerful classifiers due to their ability to automatically learn discriminative features from the input data. CNNs have been successfully applied to a wealth of learning tasks including image classification [15], natural language processing [16], and speech recognition [17]. The success in these fields naturally leads to the question of whether CNNs are competitive for

TSC. Though not currently widespread in TSC research, CNNs have been applied to TSC problems [18]. The authors used 44 of the 85 UCR/UEA datasets to evaluate a standard CNN, then defined a Multi-scale CNN (MCNN) that builds CNNs with representations of the input series under various rescaling factors.

VI. Results

A. Comparison to Deep Learning

The CNN and MCNN in [18] were evaluated on 44 of the 85 UCR/UEA datasets. However, the CNN was used only as a comparison to MCNN and the actual results of the CNN were not published, while the published MCNN results were only recorded on roughly half of the UCR/UEA datasets. We wish to run our own standard CNN over the 85 problems as a benchmark to understand how it compares to other competing approaches before comparing MCNN to the state of the art. We create CNNs in the Theano framework [29] using stochastic gradient descent with momentum with one convolutional layer, followed by a max-pooling layer and three fully connected layers. Each convolutional/fully connected layer contains 256 filters/units. The hyper-parameters (and the range of values we consider) that must be set are: the learning rate (0.1, 0.01, 0.001), filter size (0.05, 0.1, 0.2), pooling size (2, 3, 5), and the number of training epochs (50, 100, 200). We select parameters through minimising error in training, favouring smaller epochs in the event of ties to avoid overfitting. Figure 2 compares the CNN over the 85 datasets to the current state of the art, Flat-COTE, and the common benchmark of DTW 1-NN with warping set in training. Full results and code can be found on the accompanying website.

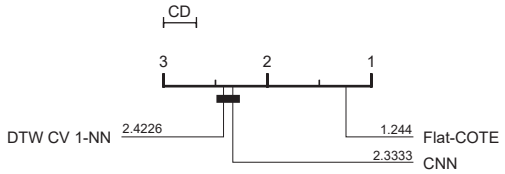

Fig. 2. Flat-COTE compared to DTW 1-NN and CNN.

Flat-COTE significantly outperforms the CNN on the UCR/UEA datasets, while the average rank of CNN is not significantly different to DTW. This result is also confirmed using pairwise tests (Binomial and Wilcoxon Signed-Rank). This demonstrates that the standard CNN is at least competitive for TSC in regard to DTW, but cannot match Flat-COTE. However, the results in [18] stated that MCNN significantly outperformed a standard CNN on 44 UCR/UEA datasets. We compare our CNN results with the standard train/test split to the published MCNN results over the common datasets. Our analysis agrees that MCNN is significantly better than the standard CNN, which naturally leads to the question of whether MCNN is better than Flat-COTE. We test this by comparing MCNN and Flat-COTE over the 44 datasets. We find that Flat-COTE wins on 28 datasets, MCNN on 14, and they tie on 2. The difference is significant according to both a binomial test and a Wilcoxon signed-rank test.

Though Flat-COTE is significantly better than MCNN, comparing MCNN to DTW 1-NN over the 44 datasets finds a significant difference in favour of MCNN. This is still an impressive feat, as only a handful of algorithms evaluated in [1] actually outperformed DTW. It demonstrates promise, and warrants further investigation of deep learning applications to TSC. However, the current state of the art is confirmed to be Flat-COTE and our next objective is to evaluate whether HIVE-COTE is a significant improvement.

B. HIVE-COTE

Figure 1 summarised the experimental evaluation in [1] that identified the leading algorithms in TSC. Those results were obtained using 100 seeded resamples of the 85 UCR/UEA datasets, which allows us to recreate the comparison and include HIVE-COTE. Due to space limitations we cannot publish full results here. However, full results are available on the accompanying website. A summary of the results are reported in Table I, and Figure 3 shows the CD diagram of HIVE-COTE compared to the leading TSC algorithms.

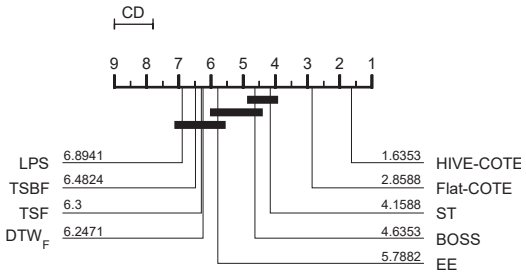

Fig. 3. The top TSC algorithms in [1] compared to HIVE-COTE.

HIVE-COTE is significantly more accurate than all alternatives, including Flat-COTE. Against all algorithms, HIVE-COTE wins on 45 out of the 85 datasets and is ranked within the top 3 classifiers on 83 problems. This underlines the utility of transformation-based ensembles and demonstrates the effectiveness of the new hierarchical structure. Finally, we could not include MCNN in the previous analysis as results were only published for 44 datasets. However, a pairwise comparison over the common datasets shows than HIVE-COTE wins on 30, MCNN on 11, and they tie on 3. The difference is significant.

C. Case Study: Ethanol Level

We can demonstrate the utility of HIVE-COTE by using the ethanol level problem. We build Flat and HIVE variants of COTE with a leave-one-bottle-out approach; we select a test bottle, train with all other bottles, then make test predictions. Repeating for all bottles gives us an overall accuracy.

As anticipated, this is a difficult problem and whole series approaches performed poorly. This is likely because data contains the whole spectrograph to incorporate as much information as possible, but it is likely that a narrow band contains the discriminatory features. As a result, the EE component of Flat-COTE only achieved only 26.6% accuracy, no better than random guessing. The ACF and PS ensembles performed slightly better with roughly 35% accuracy each, and ST performed best with over 50% accuracy. However, due to the structure of Flat-COTE, the poor learners in the time domain diluted the discriminatory power of ST and Flat-COTE barely surpassed 40% classification accuracy. HIVE-COTE

TABLE I. SUMMARY OF TEST RESULTS OVER 100 RESAMPLES OF THE 85 UCR/UEA DATASETS

	ST	BOSS	DTW_F	TSF	TSBF	LPS	EE	Flat-COTE	HIVE-COTE
Overall Average Accuracy	83.80	83.35	80.54	79.58	79.87	79.51	81.19	85.79	**86.88**
Overall Average Rank	4.16	4.64	6.25	6.30	6.48	6.89	5.79	2.86	**1.64**

faired better; the best individual constituent ensemble, RIF_ACF, achieved 66.7% accuracy, while TSF reported over 63% and BOSS recorded 52%. Combined, HIVE-COTE produced an accuracy of 64.1%. This result demonstrates two points. First, the constituent ensembles in HIVE-COTE were better than those in Flat-COTE for this problem, especially apparent by the superior performance of RIF_ACF compared to the original ACF. Second, the hierarchical structure of HIVE-COTE created a more balanced classifier, handling the poor time-domain classifiers without a significant degradation in accuracy.

VII. CONCLUSIONS

We set out to address two questions. First, was the existing state of the art, Flat-COTE, significantly better than current deep learning approaches for TSC? To answer this question we implemented a convolutional neural network on 85 datasets. We found that the CNN compared well to other approaches, but Flat-COTE was significantly more accurate by a large margin. We then turned our attention to the results of a recently published TSC-specific CNN from the literature. Over the 44 dataset that results were reported for, Flat-COTE was significantly more accurate. This lead to our second question: could we create a new version of the collective that was significantly better? We introduced HIVE-COTE, a meta-ensemble that improves on Flat-COTE by incorporating three new classifiers into the collective, two from the literature and one new, and combines constituent ensembles through a novel hierarchical probabilistic voting structure. Finally, we compared HIVE-COTE to the best classifiers from a recent experimental evaluation in [1] on 100 resamples of 85 datasets. The results demonstrated that HIVE-COTE is significantly more accurate than all of the alternatives, including Flat-COTE. To the best of our knowledge, HIVE-COTE is the most accurate algorithm for TSC.

ACKNOWLEDGEMENTS

This work is 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. The authors would also like to thank James Large for recording the Ethanol Level problem data.

REFERENCES

[1] A. Bagnall, J. Lines, A. Bostrom, and J. Large, "The great time series classification bake off: An experimental evaluation of recently proposed algorithms," *DAMI*, vol. online first, 2016.

[2] P. Marteau, "Time warp edit distance with stiffness adjustment for time series matching," *IEEE PAMI*, vol. 31, no. 2, pp. 306–318, 2009.

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

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

[5] P. Schäfer, "The boss is concerned with time series classification in the presence of noise," *DAMI*, vol. 29, no. 6, pp. 1505–1530, 2015.

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

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

[8] L. Ye and E. Keogh, "Time series shapelets: a novel technique that allows accurate, interpretable and fast classification," *DAMI*, vol. 22, no. 1, pp. 149–182, 2011.

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

[10] A. Bagnall, J. Lines, J. Hills, and A. Bostrom, "Time-series classification with cote: The collective of transformation-based ensembles," *IEEE TKDE*, vol. 27, no. 9, pp. 2522–2535, 2015.

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

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

[13] T. Gorecki and M. Luczak, "Non-isometric transforms in time series classification using dtw," *Knowledge-Based Systems*, vol. 61, pp. 98–108, 2014.

[14] R. J. Kate, "Using dynamic time warping distances as features for improved time series classification," *DAMI*, vol. 30, no. 2, pp. 283–312, 2016.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, 2012, pp. 1097–1105.

[16] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," *arXiv:1404.2188*, 2014.

[17] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *ICASSP 2013*. IEEE, 2013, pp. 6645–6649.

[18] Z. Cui, W. Chen, and Y. Chen, "Multi-scale convolutional neural networks for time series classification," *arXiv:1603.06995*, 2016.

[19] G. Batista, E. Keogh, O. Tataw, and V. deSouza, "Cid: an efficient complexity-invariant distance measure for time series," *DAMI*, vol. 28, no. 3, pp. 634–669, 2014.

[20] M. Baydogan, G. Runger, and E. Tuv, "A bag-of-features framework to classify time series," *IEEE PAMI*, vol. 25, no. 11, pp. 2796–2802, 2013.

[21] M. G. Baydogan and G. Runger, "Time series representation and similarity based on local autopatterns," *DAMI*, pp. 1–34, 2015.

[22] J. Grabocka, N. Schilling, M. Wistuba, and L. Schmidt-Thieme, "Learning time-series shapelets," in *Proc. 20th ACM SIGKDD*, 2014.

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

[24] M. Corduas and D. Piccolo, "Time series clustering and classification by the autoregressive metric," *Computational Statistics & Data Analysis*, vol. 52, no. 4, pp. 1860–1872, 2008.

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

[26] B. D. Fulcher and N. S. Jones, "Highly comparative feature-based time-series classification," *IEEE TKDE*, vol. 26, no. 12, pp. 3026–3037, 2014.

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

[28] A. Bostrom and A. Bagnall, "Binary shapelet transform for multiclass time series classification," in *Big Data Analytics and Knowledge Discovery*, 2015, pp. 257–269.

[29] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints*, vol. abs/1605.02688, 2016.