Scalable Dictionary Classifiers for Time Series Classification

Matthew Middlehurst(✉), William Vickers, and Anthony Bagnall

School of Computing Sciences, University of East Anglia, Norwich, UK
M.Middlehurst@uea.ac.uk

Abstract. Dictionary based classifiers are a family of algorithms for time series classification (TSC) that focus on capturing the frequency of pattern occurrences in a time series. The ensemble based Bag of Symbolic Fourier Approximation Symbols (BOSS) was found to be a top performing TSC algorithm in a recent evaluation, as well as the best performing dictionary based classifier. However, BOSS does not scale well. We evaluate changes to the way BOSS chooses classifiers for its ensemble, replacing its parameter search with random selection. This change allows for the easy implementation of contracting (setting a build time limit for the classifier) and check-pointing (saving progress during the classifiers build). We achieve a significant reduction in build time without a significant change in accuracy on average when compared to BOSS by creating a fixed size weighted ensemble selecting the best performers from a randomly chosen parameter set. Our experiments are conducted on datasets from the recently expanded UCR time series archive. We demonstrate the usability improvements to randomised BOSS with a case study using a large whale acoustics dataset for which BOSS proved infeasible.

Keywords: Time series · Classification · Dictionary · Contracting

1 Introduction

Dictionary based learning is commonly employed in signal processing, computer vision and audio processing to capture recurring discriminatory features. The approach has been successfully applied to time series classification (TSC) in a variety of ways. An extensive experimental study [1] found that the best dictionary approach was the ensemble classifier the Bag of Symbolic Fourier Approximation Symbols (BOSS). It was shown that dictionary based classifiers detect a fundamentally different type of discriminatory features than other TSC approaches, and the addition of the BOSS ensemble to the meta ensemble the Hierarchical Vote Collective of Transformation-based Ensembles (HIVE-COTE) [12] leads to a significant improvement in accuracy. BOSS is an ensemble classifier that evaluates a range of parameter combinations over a grid, then retains all classifiers that are within 92% of the best combination, as measured by a leave-one-out cross-validation on the train data.

H. Yin et al. (Eds.): IDEAL 2019, LNCS 11871, pp. 11–19, 2019.
https://doi.org/10.1007/978-3-030-33607-3_2

The BOSS ensemble has some drawbacks. Firstly, the need to cross-validate each parameter combination means that it scales poorly. Secondly, the fact that it retains a variable number of base classifiers means that it is often very memory intensive. Thirdly, the histograms can be very large, so storing them all for each base classifier also requires a significant memory commitment for large problems. One proposed method to solve this, the BOSS vector space (BOSS-VS) classifier [15], has been shown to be significantly less accurate than the full BOSS [13,16]. We investigate whether we can mitigate against these problems without this significant loss in accuracy. Our primary contribution is to propose a new classifier, cBOSS, that uses an alternative ensemble mechanism to provide an order of magnitude speed up without loss of accuracy. Our secondary contributions include reproducing results from a related study [16] and making cBOSS both contractable (i.e. it is able to build the best possible classifier in a fixed amount of time) and check-pointable (i.e. the build classifier stage can be stopped and restarted). Our experiments are easily reproducible and we have released a Weka compatible version of BOSS, cBOSS and other tested classifiers that is integrated into the UEA Codebase[1]. There is also a Python version of the classifiers developed for the Alan Turing Institute sktime package[2].

The rest of this paper is structured as follows. Section 2 gives a brief description of dictionary based classification algorithms. Section 3 describes the alterations we make to the BOSS algorithm to make cBOSS. Section 4 presents the results of our experimental evaluation, and Sect. 5 concludes and offers some ideas for future work.

2 Time Series Dictionary Based Classifiers

Classifiers that use frequency of words as the basis for finding discriminatory features are often referred to as dictionary based classifiers [1]. They have close similarities to bag-of-words based approaches that are commonly used in computer vision. Informally, a dictionary based approach will be useful when the discriminatory features are repeating patterns that occur more frequently in one class then other classes.

Bag of SFA Symbols (BOSS)

A single BOSS [14] base classifier proceeds as follows. For each series, it extracts the windows sequentially, normalising the window if the parameter p is true. It then applies a Discrete Fourier Transform (DFT) to the resulting subseries, ignoring the first coefficient if p is true. The DFT coefficients are truncated to include only the first $l/2$ Fourier terms (both real and imaginary). The truncated samples are then discretised into α possible values using an algorithm called Multiple Coefficient Binning (MCB) (see [14]). MCB involves a preprocessing step to find the discretising break points by estimating the distribution of the Fourier coefficients. Consecutive windows producing the same word are only counted as

[1] https://github.com/TonyBagnall/uea-tsc.
[2] https://github.com/alan-turing-institute/sktime.

a single instance of the word. A bespoke BOSS distance function is used with a nearest neighbour classifier to classify new instances. The distance function is non-symmetrical, only including the distance for features that are non-zero in the first feature vector given. The BOSS base classifier has four parameters: window length w, word length l, whether to normalise each window p and alphabet size α. The BOSS ensemble (also referred to as just BOSS), evaluates all BOSS base classifiers in the range $w \in \{10 \ldots m\}$, $l \in \{16, 14, 12, 10, 8\}$ and $p \in \{true, false\}$. This parameter search is used to determine which base classifiers are used in the ensemble. Following [14], the alphabet size is fixed to 4 for all experiments. The number of window sizes is a function of the series length m. All BOSS base classifiers with a training accuracy within 92% of the best performing base classifier are kept for the ensemble. This dependency on series length and variability of ensemble size is a factor that can significantly impact on efficiency. Classification of new instances is then done using majority vote from the ensemble.

Word Extraction for Time Series Classification (WEASEL) WEASEL [16] is a dictionary based classifier that is an extension of BOSS. WEASEL is a single classifier rather than an ensemble. WEASEL concatenates histograms for a range of parameter values of w and l, then performs a feature selection to reduce the feature space. Like BOSS, WEASEL performs a Fourier transform on each window. DFT coefficients are no longer truncated and instead the most discriminative real and imaginary features are retained, as determined by an ANOVA F-test. The retained values are then discretised into words using information gain binning, similar to the MCB step in BOSS. WEASEL does not remove adjacent duplicate words as BOSS does. The word and window size are used as keys to index the histogram. A further histogram is formed for bigrams. The number of features is reduced using a chi-squared test after the histograms for each instance are created, removing any words which score below a threshold. WEASEL uses a logistic regression classifier to make the predictions for new cases. WEASEL performs a parameter search for p and a reduced range of l and uses a 10-fold cross-validation to determine the performance of each set. The alphabet size α is fixed to 4 and the *chi* parameter is fixed to 2.

3 BOSS Enhancements (cBOSS)

Our changes to BOSS mainly focus on the ensemble technique of the classifier, which is computationally expensive and unpredictable. Ensembling has been shown to be an essential component of BOSS, resulting in significantly higher accuracy [8]. We assess whether we can replace the current ensemble mechanism with a more stable and efficient scheme without a significant reduction in accuracy. We found that complete randomisation, i.e. selecting random parameter combinations for a fixed number of base classifiers, worked reasonably well but on some data performed very badly. Hence, we retain an internal evaluation of each possible member through leave-one-out cross validation on the train data, then use this value to both select and weight classifier votes. The primary difference to BOSS is that we do not determine which members to retain after the complete search. Rather, we introduce a new parameter, k, the fixed ensemble size, and maintain a list of weights. The option to replace the ensemble size k with a time limit t through contracting is made available through this change in building process, when set the classifier will continue building until the time since building started is greater than t. Even with weighting, classifiers with poor parameters for a particular problem can degrade the overall classifier. Because of this we set a max ensemble size s to filter these out, with any classifier built past this value replacing the current lowest accuracy member of the ensemble if its accuracy is higher. To further diversify the ensemble and increase efficiency we take a randomly selected 70% subsample of the training data for each individual classifier. The parameter space we randomly sample for cBOSS is the same as that which BOSS searches exhaustively through. For classifying new instances, we adopt the exponential weighting scheme used in the Cross-validation Accuracy Weighted Probabilistic Ensemble (CAWPE) [9] to amplify small difference in weights and results in significantly improved performance. cBOSS is more formally described in Algorithm 1.

Algorithm 1. cBOSS_build(A list of n cases length m, $\mathbf{T} = (\mathbf{X}, \mathbf{y})$)

```
Parameters: the ensemble size $k$, the max ensemble size $s$
Let $w$ be window length, $l$ be word length, $p$ be normalise/not normalise and $\alpha$ be alphabet size.
Let $\mathbf{C}$ be a list of $s$ BOSS classifiers $(\mathbf{c}_1, \ldots, \mathbf{c}_s)$
Let $\mathbf{E}$ be a list of $s$ classifier weights $(\mathbf{e}_1, \ldots, \mathbf{e}_s)$
Let $\mathbf{R}$ be a set of possible BOSS parameter combinations
$i \leftarrow 0$
$lowest\_acc \leftarrow \infty, lowest\_acc\_idx \leftarrow \infty$
while $i < k$ AND $|\mathbf{R}| > 0$ do
    $[l, a, w, p] \leftarrow random\_sample(\mathbf{R})$
    $\mathbf{R} = \mathbf{R} \setminus \{[l, a, w, p]\}$
    $\mathbf{T}' \leftarrow$ subsample_data($\mathbf{T}$)
    $cls \leftarrow$ build_base_BOSS($\mathbf{T}', l, a, w, p$)
    $acc \leftarrow$ LOOCV($cls$) { train data accuracy}
    if $i < s$ then
        if $acc < lowest\_acc$ then
            $lowest\_acc \leftarrow acc$, $lowest\_acc\_idx \leftarrow i$
        $c_i \leftarrow cls$, $e_i \leftarrow acc^4$
    else if $acc > lowest\_acc$ then
        $c_{lowest\_acc\_idx} \leftarrow cls$, $e_{lowest\_acc\_idx} \leftarrow acc^4$
        $[lowest\_acc, lowest\_acc\_idx] \leftarrow$ find_new_lowest_acc($\mathbf{C}$)
    $i \leftarrow i + 1$
```

4 Results

We compare cBOSS to the dictionary base classifiers BOSS [14] and WEASEL [16] in terms of accuracy and speed. Two known dictionary based classifiers not included in our comparison are Bag of Patterns (BOP) [10] and SAX-VSM [17]. Both of these classifiers were found to be significantly worse than the common time series classification benchmark, one nearest neighbour dynamic time warping [1]. Each of the classifiers are tested on the same 30 random resamples from the 114 datasets without missing values in the UCR repository. For these experiments we set the values for the cBOSS k and s values to 250 and 50 respectively. Figure 1 shows the critical difference diagram for these three classifiers. Solid bars indicate cliques where there is no significant difference between classifiers. Tests of difference are performed using pairwise Wilcoxon signed rank tests with the Holm correction. The results confirm that WEASEL is indeed significantly better than BOSS. They also demonstrate that there is no significant difference between BOSS and cBOSS.

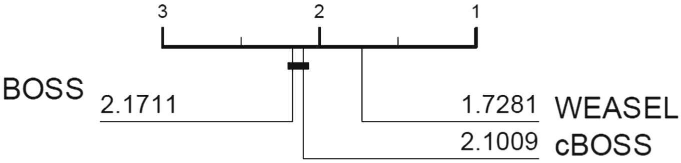

Fig. 1. Critical difference diagram showing accuracy ranks and cliques for the three dictionary based classifiers.

The aim of cBOSS is to be not significantly less accurate than BOSS in significantly less time than BOSS. Figure 2 shows the average build time plotted against average rank for the four classifiers. cBOSS is on average 12 times faster than BOSS, with no significant loss of accuracy.

To reduce the risk of bias or any suggestion of cherry picking, we have conducted experiments with all the datsets in the archive. However, many of these problems are small. To examine the effect on larger problems, we take a closer look at the results for 16 problems on which BOSS takes over an hour to build. Table 1 shows the pattern of results is the same on these 16 data. cBOSS is not significantly worse than BOSS, but an order of magnitude faster.

Table 1. Average large dataset performance by classifier using accuracy rank, Area Under the Receiver Operating Characteristic curve (AUROC) and Negative Log-Likelihood (NLL). Included is total build time over all datasets relative to BOSS.

Classifier	AccRank	AUROC	NLL	Build time
WEASEL	1.375	0.9529	0.9753	85.48%
cBOSS	2.25	0.9558	0.7951	7.7%
BOSS	2.375	0.953	0.7923	100%

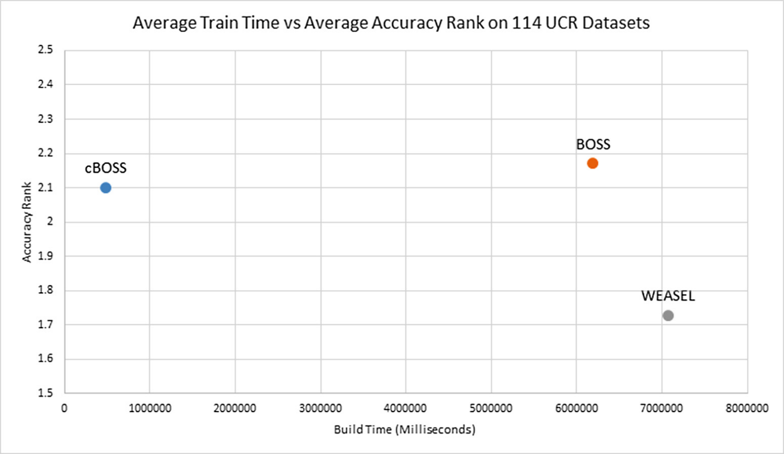

Fig. 2. Average build time against average accuracy rank for each dictionary classifier, with a lower value for both axis being a better performance.

To demonstrate the effectiveness of building cBOSS using a time contract Fig. 3 shows the change in accuracy over a range of t values on the 16 large problems. As shown an increase in time the classifier contracted for increases accuracy on average, though this increase slows as the parameter search progresses. The versatility of being able to build the best classifier within a train time limit is a large usability boost to the classifier, making this aspect much more predictable.

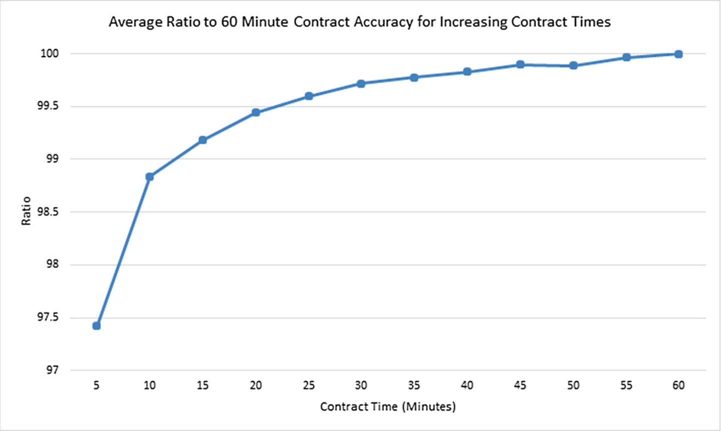

Fig. 3. Average accuracy value for a range of contract times on 16 large datasets. Times range from 5 to 60 min linearly spaced in increments of 5.

4.1 Whale Acoustics Use Case

Our interest in making BOSS scalable arose due to our desire to use it with an application in classifying animals based on acoustic samples. Recently, the protection of endangered whales has been a prominent global issue. Being able to accurately detect marine mammals is important to monitor populations and provide appropriate safeguarding for their conservation. North Atlantic right whales are one of the most endangered marine mammals with as few as 350 individual remaining in the wild [7]. We use a dataset from the Marinexplore and Cornell University Whale Detection Challenge[3] that features a set of right whale up-calls. Up-calls are the most commonly documented right whale vocalisation with an acoustic signature of approximately 60 Hz–250 Hz, typically lasting 1 s. Right whale calls can often be difficult to hear as the low frequency band can become congested with anthropogenic sounds such as ship noise, drilling, piling, or naval operations [3]. Each series is labelled as either containing a right whale or not with the aim to correctly identify the series that contain up-calls.

Previous work has been done in classifying the presence of whale species using acoustic data with whale vocalisations [6,18]. However, TSC approaches have not been applied. This problem is a good example of a large dataset for which it is infeasible to use BOSS. The dataset contains 10,934 train cases and 5885 test cases. Each case is a two second audio segment sampled at 2 kHz, giving a series length of 4000. We have done no preprocessing: the purpose of these experiments is to provide benchmark results for bespoke audio approaches and to test the scalability of cBOSS. The problem is large. It exceeds the largest train set from ElectricDevices of 8926 and series length from Rock of 2844 in the 128 UCR archive [4] (we will donate this data to the archive for the next release). For benchmarking alongside cBOSS on this dataset, we test the five classifiers that make up the HIVE-COTE ensemble [12], as well as two potential candidates for the ensemble WEASEL and Proximity Forest (PF) [13]. Only two of these classifiers, Time Series Forest (TSF) [5] and Random Interval Spectral Ensemble (RISE) [12], will complete within 28 days. The Shapelet Transform (ST) [2,11] is contractable, and we present results with a contracted time limit of five days. For fairness we ran PF on a single thread, but the capability to run using multiple threads is available and more likely to finish below the 28 day limit. BOSS did

Table 2. Accuracy and build time in hours for each of the potential HIVE-COTE components and cBOSS.

Classifier	Accuracy	Build time (hours)
cBOSS	0.8114	119.39
ST	0.818	120.05
RISE	0.7859	141.96
TSF	0.7712	16.23

[3] https://www.kaggle.com/c/whale-detection-challenge/data.

not complete, and also required huge amounts of memory (greater than 100 GB) to run at all. Attempts to build WEASEL ceased after a 300 GB memory limit was exceeded. cBOSS completed within 5 days, without a contract. Table 2 shows the accuracy and build time on the whales dataset each finished classifiers. These exploratory results suggest that both dictionary and shapelet approaches may be useful for this application.

5 Conclusion

We present cBOSS, a more scalable version of the BOSS classifier that uses a new ensemble mechanism. The replacement of the parameter search with randomly selected parameter sets provides a considerable speed up, and the introduction of subsampling for increased diversity and weighted voting means cBOSS is not significantly less accurate than BOSS. The inclusion of a fixed ensemble size, the ability to contract the build time and save progress with check-pointing make the classifier more robust and more predictable.

We have independently recreated the published results for the WEASEL classifier and verified the findings in [16]. WEASEL is significantly better than both BOSS and cBOSS. However, it also the slowest classifier to build on average, and has an equally large memory footprint as BOSS. This indicates this it is a suitable BOSS replacement on smaller datasets, it has the same scalability issues as BOSS. cBOSS is a viable alternative for large problems if a dictionary based classifier is required.

cBOSS scales well up to problems with tens of thousands of cases. However, building models with several hundred thousand instances may still cause an issue in requirements for space and time. Simple expedients such as subsampling can facilitate building models on large data, but doing this automatically whilst maintaining accuracy is challenging. Furthermore, cBOSS is still comparatively memory intensive, since it uses a nearest neighbour classifier. Our attempts to use alternative less memory intensive classifiers have been unsuccessful. Instead, we intend to introduce a contract for memory, setting the sampling size and ensemble size accordingly. Investigations into the feasibility of this and any affect on the classifiers performance could be an interesting future work in further improving dictionary based scalability.

Acknowledgements. This work is supported by the UK Engineering and Physical Sciences Research Council (EPSRC) iCASE award T206188 sponsored by British Telecom. 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.

References

1. Bagnall, A., Lines, J., Bostrom, A., Large, J., Keogh, E.: The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Min. Knowl. Disc. **31**(3), 606–660 (2017)

2. Bostrom, A., Bagnall, A.: Binary shapelet transform for multiclass time series classification. Trans. Large-Scale Data Knowl. Centered Syst. **32**, 24–46 (2017)
3. Cox, T.M., et al.: Understanding the impacts of anthropogenic sound on beaked whales. Technical report, Space and Naval Warfare Systems Centre, San Diego, CA, USA (2006)
4. Dau, H., et al.: The UCR time series archive. ArXiv e-prints arXiv:1810.07758 (2018)
5. Deng, H., Runger, G., Tuv, E., Vladimir, M.: A time series forest for classification and feature extraction. Inf. Sci. **239**, 142–153 (2013)
6. Dugan, P.J., Rice, A.N., Urazghildiiev, I.R., Clark, C.W.: North Atlantic right whale acoustic signal processing: part i. comparison of machine learning recognition algorithms. In: 2010 IEEE Long Island Systems, Applications and Technology Conference, pp. 1–6. IEEE (2010)
7. Kraus, S.D., et al.: North Atlantic right whales in crisis. Science **309**(5734), 561–562 (2005)
8. Large, J., Bagnall, A., Malinowski, S., Tavenard, R.: On time series classification with dictionary-based classifiers. Intell. Data Anal. (2019)
9. Large, J., Lines, J., Bagnall, A.: A probabilistic classifier ensemble weighting scheme based on cross validated accuracy estimates. Data Min. Knowl. Disc. (2019)
10. Lin, J., Khade, R., Li, Y.: Rotation-invariant similarity in time series using bag-of-patterns representation. J. Intell. Inf. Syst. **39**(2), 287–315 (2012)
11. Lines, J., Davis, L., Hills, J., Bagnall, A.: A shapelet transform for time series classification. In Proceedings the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2012)
12. Lines, J., Taylor, S., Bagnall, A.: Time series classification with HIVE-COTE: the hierarchical vote collective of transformation-based ensembles. ACM Trans. Knowl. Disc. Data **12**(5), 52:1–52:35 (2018)
13. Lucas, B., et al.: Proximity forest: an effective and scalable distance-based classifier for time series. Data Min. Knowl. Disc. **33**(3), 607–635 (2019)
14. Schäfer, P.: The BOSS is concerned with time series classification in the presence of noise. Data Min. Knowl. Disc. **29**(6), 1505–1530 (2015)
15. Schäfer, P.: Scalable time series classification. Data Min. Knowl. Disc. **30**(5), 1273–1298 (2016)
16. Schäfer, P., Leser, U.: Fast and accurate time series classification with weasel. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 637–646. ACM (2017)
17. Senin, P., Malinchik, S.: SAX-VSM: interpretable time series classification using sax and vector space model. In Proceedings 13th IEEE International Conference on Data Mining (ICDM) (2013)
18. Shamir, L., et al.: Classification of large acoustic datasets using machine learning and crowdsourcing: application to whale calls. J. Acoust. Soc. Am. **135**(2), 953–962 (2014)