Neural Processing Letters (2024) 56:13
https://doi.org/10.1007/s11063-024-11531-1

TSCF: An Improved Deep Forest Model for Time Series Classification

Mingxin Dai[1,2] · **Jidong Yuan**[1,2] · **Haiyang Liu**[1,2] · **Jinfeng Wang**[1,2]

Accepted: 8 January 2024 / Published online: 7 February 2024

Abstract
The deep forest presents a novel approach that yields competitive performance when compared to deep neural networks. Nevertheless, there are limited studies on the application of deep forest to time series classification (TSC) tasks, and the direct use of deep forest cannot effectively capture the relevant characteristics of time series. For that, this paper proposes time series cascade forest (TSCF), a model specifically designed for TSC tasks. TSCF relies on four base classifiers, i.e., random forest, completely random forest, random shapelet forest, and diverse representation canonical interval forest, allowing for feature learning on the original data from three granularities: point, subsequence, and summary statistics calculated based on intervals. The major contribution of this work, is to define an ensemble and deep classifier that significantly outperforms the individual classifiers and the original deep forest. Experimental results show that TSCF outperforms other forest-based algorithms for solving TSC problems.

Keywords Time series classification · Deep forest · Shapelet · Interval

1 Introduction

Time series is a continuous collection of data with a temporal relationship that presents unique classification challenges compared to traditional data. In recent years, numerous methods dedicated to time series classification (TSC) have emerged. For example, dynamic

✉ Jidong Yuan
yuanjd@bjtu.edu.cn

Mingxin Dai
22125174@bjtu.edu.cn

Haiyang Liu
haiyangliu@bjtu.edu.cn

Jinfeng Wang
22120428@bjtu.edu.cn

1 Key Laboratory of Big Data and Artificial Intelligence in Transportation, Ministry of Education, Beijing, China

2 School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China

time warping (DTW) and its variants have been shown to be competitive similarity measures in solving TSC problems [1–3]. Shapelets [4], discriminating and interpretable subsequences found within time series, are generally used as the basis for decision tree node splitting, with proven advantages in improving accuracy. Calculating summary statistics based on intervals [5, 6] is also one of the main TSC methods, which find discriminating features on different intervals by randomly extracting intervals in sequences and calculating various statistics.

Decision trees' performance has made them a popular choice for classification tasks, leading researchers to propose several forest-based TSC algorithms. Traditional methods, such as random forest (RF) and completely random forest (CRF), consider the original time series at the point-level and select a single attribute in the decision tree node, that is, a point in the time series as the splitting basis. Generalized random shapelet forest (RSF) [7] randomly selects both the training data and subsequences to construct the basic decision trees, which focuses on feature learning at the subsequence-level. Time series forest (TSF) [5], on the other hand, summarizes the interval statistics as input for decision trees. However, TSF only uses three simple summary statistics: the mean; standard deviation; and slope. Canonical interval forest (CIF) [6] combines TSF and Catch22 [8], including 25 features to sample randomly. Diverse representation canonical interval forest (DrCIF) [9], as an extension of CIF, extracts intervals from three representations of time series and uses 29 statistics to form a pool of feature candidates. Nonetheless, the forest-based methods above only consider a single granularity feature of time series, i.e., point, subsequence, or summary statistics. As such, each algorithm overlooks time series characteristics demonstrated by other granularities, resulting in limitations in classification accuracy.

The gcForest [10] uses forests to emulate neural network connectivity and has fewer hyperparameters than deep neural networks (DNNs). The model's structure has two modules: multi-grained scanning, which extracts features using varying-sized windows, and cascade forest, a layer-by-layer processing system with two RFs and two CRFs. We conducted experiments using gcForest on UCR datasets,[1] but it did not outperform existing forest-based methods. Although gcForest considers information of different granularities through sliding windows with different sizes, it fails to fully consider the characteristics of the time series itself, i.e., temporal order, slope information, similarity between different time series, etc. Examining how to enhance the performance of gcForest to meet the needs of TSC tasks is a crucial topic warranting our attention.

Considering the challenges mentioned above, our aim is to improve the forest-based approach by embedding forest-based classifiers suitable for TSC based on the cascade structure of gcForest. This paper proposes a novel forest-based ensemble algorithm called time series cascade forest (TSCF). TSCF integrates four base classifiers, including RF, CRF, RSF, and DrCIF. By using them, while ensuring the diversity of the model, multi-granular feature learning of time series is realized. Our main contributions are as follows.

- We propose a deep forest model specifically designed to solve the TSC task, which embeds four forest-based classifiers into cascade layers to form the cascade forest.
- The model implements multi-granular feature learning by considering point-level features, discriminative subsequences, different statistics for phase dependent intervals.
- Experiments on 113 UCR datasets show that TSCF outperforms existing forest-based methods and is highly competitive with other baselines.

The structure of this paper is as follows. In Sect. 2 we review various categories of relevant studies. In Sect. 3 we detail the individual classifiers in TSCF and its overall architecture. In

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

Sect. 4 we first present and analyse experimental comparisons of TSCF and the current state-of-the-art in TSC, then compare TSCF and its variants, and perform ablation experiments. Finally, in Sect. 5 we summarise our conclusions and discuss future work.

2 Related Work

In this section, we present a non-exhaustive review of the state-of-the-arts in TSC algorithms, grouping them as distance-based, shapelet-based, interval-based, dictionary-based, hybrid, and deep learning methods based on our research needs.

Distance-based methods utilize various distance calculation techniques to measure the similarity/dissimilarity between two time series. The DTW-based nearest neighbor classifier serves as a popular baseline for comparison. For the improvement of DTW, considerations such as the first derivative [1], imposing a penalty [3], and the local slope [2] have been explored. There are also several other distance measures, for example, move-split-merge (MSM) [11] transforms any time series into any other time series by three fundamental operations: move, split, and merge. The elastic ensemble (EE) [12] integrates the nearest neighbor classifiers based on 11 distance measures and proves that this ensemble method is better than any single composition method and DTW. Shape dynamic time warping (shapeDTW) [13], which enhances DTW by taking point-wise local structural information into consideration.

Shapelet-based approaches concentrate on distinctive local features of time series, which can be categorized into two types. The first uses shapelets [4] as a basis for decision tree node splitting. In addition to the previously mentioned RSF [7], random pairwise shapelets forest (RPSF) [14], unlike RSF which selects one shapelet at the node for splitting, uses a pair of shapelets from different classes to construct trees. To enhance the selection accuracy and calculation speed of shapelets, researchers have formulated several improved methods by using intelligent caching and exploiting a random projection technique on the symbolic aggregate approximation representation to search shapelet candidates. To optimize time complexity, the algorithm learns about shapelets in close proximity to the optimal ones [15] and employs local fisher discriminant analysis [16]. The second type is shapelet transformation (ST) initially proposed in [17]. Data are transformed into a new feature space by calculating the distance between the original data and the selected k best shapelet candidates, which is adaptive to different classifiers.

In addition to the TSF, CIF and DrCIF mentioned above, interval-based methods also include random interval features (RIF) [18], which does not use time domain intervals but two transformed representations of data, including RIF_ACF for autocorrelation-transformed data and RIF_PS for the power spectrum-transformed data. Unlike TSF, random interval spectral ensemble (RISE) [19] only randomly selects an interval for each base classifier, and then calculates the periodogram and auto-regression function based on the interval and stitches them into feature vectors and then builds trees. Supervised time series forest (STSF) [20] has supervised extraction intervals and screens the intervals that can be preserved with Fisher scores. There are also time series bag of features (TSBF) [21], learned pattern similarity (LPS) [22], etc.

Dictionary-based methods extracts words from a time series through a sliding window and classifies them based on those words. Bag of symbolic Fourier approximation symbols (BOSS) [23] transforms a time series into an unordered set of symbolic Fourier approximation (SFA) words. Word extraction for time series classification (WEASEL) [24] proposes a specific feature transformation method to learn a smaller but more discriminating feature set from time series.

Hybrid methods incorporate multiple perspectives of the time series. The hierarchical vote collective of transformation ensembles (HIVE-COTE) [18] is a various ensemble containing five classifiers each from a different representation. Numerous hybrid methods, such as random convolutional kernel transform (ROCKET) [25], time series combination of heterogeneous and integrated embeddings forest (TS-CHIEF) [26] and Catch22 [8], have also been developed for TSC. Interested researchers could refer to [27] for a detailed survey of the traditional method for TSC.

Deep learning, specifically DNNs, has been widely used in TSC problems. Time leNet (t-leNet) [28] first uses a convolutional neural network with temporal convolutions for TSC. Another effective approach is the multi-scale convolutional neural network (MCNN) [29] that downsamples the original sequence in the multi-scale and multi-frequency domains. Furthermore, fully convolutional neural networks (FCN), residual networks (ResNet) [30], time warping invariant echo state networks (TWIESN) [31], and InceptionTime [32] are all successfully applied to TSC. For a more comprehensive overview of deep learning methods for TSC, we recommend interested researchers refer to [33].

3 Time Series Cascade Forest

In this section we will elaborate TSCF in detail, starting with the problem definition, followed by the four forest-based classifiers that build up the model, and then the overall structure and complexity analysis. The running process of TSCF is described in Algorithms 1 and 2.

3.1 Problem Definition

Given a set of time series $D = \{S_i\}_{i=1}^{m}$ of m instances with labels Y, where each time series $S_i = \{x_j^i\}_{j=1}^{n}$ has n ordered real-valued observations and a discrete class label y from a range of c possible values. A classifier is a function or mapping from the space of possible inputs to a probability distribution over the class variable values. We consider the problem of univariate and equal length time series. The goal of TSCF is to learn the class probability vector $aug = \{v_1, v_2, v_3, v_4\}$, where v_1, v_2, v_3, and v_4 represent the prediction probability of RF, CRF, RSF [7], and DrCIF [9], respectively. Classification accuracy is reported by comparing the maximum value of the mean of the four class vectors of the last layer to the true class label.

3.2 Forest-Based Classifiers

Since the combination of RF, CRF, RSF, and DrCIF enables the learning of multi-granular features of time series from different perspectives, they are selected as the base classifiers for constructing our TSCF. Details are described as follows.

(1) The RF in TSCF is an integration of traditional decision trees and increases the speed of tree building through parallel techniques. For the node splitting of each decision tree, as shown in Fig. 1a, $\sqrt{n}$ point-level features of the time series are randomly selected. The split attribute of the node is the smallest *Gini* index selected from the $\sqrt{n}$ selected attributes.

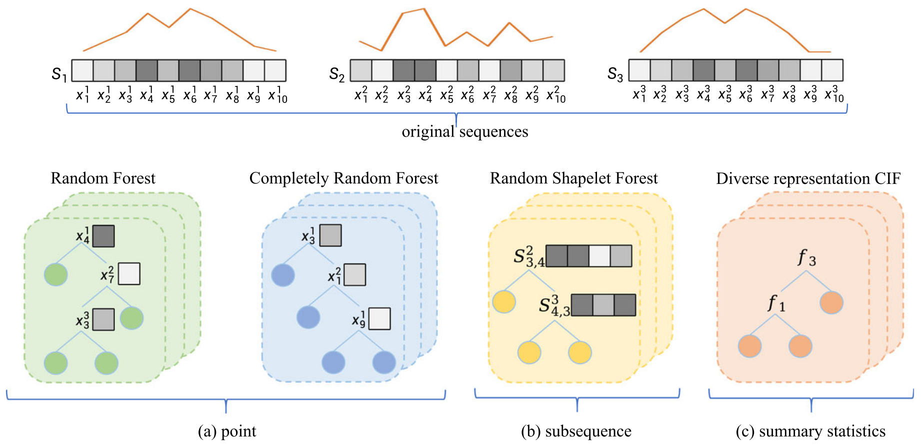

Fig. 1 Interpretation of the model's multi-granular feature learning

$$Gini(D) = 1 - \sum_{k=1}^{K} \left(\frac{|C_k|}{|D|} \right)^2, \tag{1}$$

where C_k is the subset of samples in D belonging to class k, and K is the number of classes. *Gini* index allows the RF to learn distinguishable point features present in time series.

(2) The CRF comprises totally random trees, each of which randomly selects a point-level feature for splitting at every node as shown in Fig. 1a. Compared to the RF, it has a higher level of randomness and does not consider the splitting criterion. We selected two forests at the point-level. The CRF enhances the randomness of the model and both use parallel tree building technology, which can reduce the model training time.

(3) In RSF [7], the shapelet acts as a discriminating subsequence of the time series that allows the model to learn subsequence-level features. Suppose that the shapelet is defined as $S^m_{p,q} = \left\{ x^m_p, x^m_{p+1}, \ldots, x^m_{p+q-1} \right\}$, m represents the shapelet from the m^{th} series, p is the start position and q is the length of shapelet, where $1 \leq p \leq n - q + 1$. For each node of the decision tree, the length of the shapelet q and its starting point are randomly determined. In each node, RSF randomly selects s shapelet candidates and then chooses the one with the largest information gain as the final splitting basis as shown in Fig. 1b.

(4) The last base classifier is DrCIF [9], an interval based TSC algorithm. DrCIF extracts intervals from three representations, including the primitive series, the first order difference series, and the periodograms of the entire series. The algorithm steps of DrCIF are as follows: Firstly, randomly select k intervals from three representations, and the starting point and length of each interval are randomly selected. a out of the pool of 29 feature candidates (i.e. mean, standard-deviation, slope, median, inter-quartile range, min, max, and Catch22) are randomly selected to calculate summary statistics for each interval. Then, these features are concatenated into $3 \cdot k \cdot a$ length vector f_i for each series S_i, and the new dataset $F = \{f_i\}_{i=1}^{m}$ is used to build the time series tree [5], as shown in Fig. 1c.

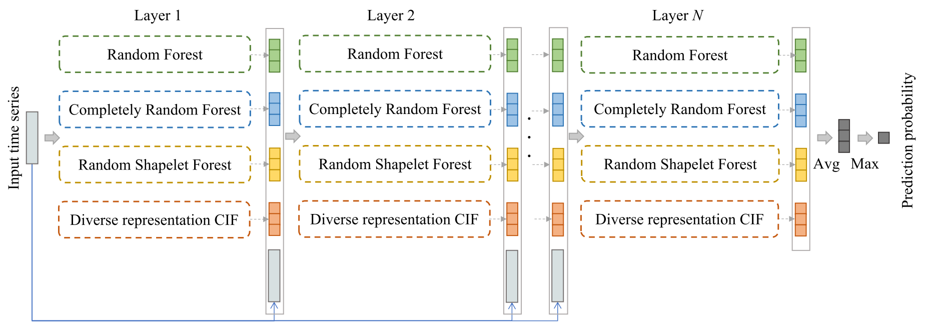

Fig. 2 Illustration of TSCF method. Each layer of the cascade consists of a RF, a CRF, a RSF, and a DrCIF. Suppose there are three classes to be classified. Each forest outputs a three-dimensional class vector, which concatenates with the original input time series as re-representation

3.3 Overall Structure

Inspired by gcForest, TSCF employs a cascade structure, as illustrated in Fig. 2. The original gcForest contains multi-grained scanning and cascade forest. In TSCF, since the four forests each perform feature learning on the data from three granularities, the multi-grained scanning module is no longer needed.

The establishment process of cascade forest is shown in Algorithm 1. First, the first cascade layer (line 1, as detailed in Algorithm 2) is built to obtain the class vectors generated by the four forests, i.e., v_1, v_2, v_3, and v_4, and then $aug = (||_{i=1}^{4} v_i)$, where $||_{i=1}^{4}$ is the concatenation operation from class vector v_1 to class vector v_4. Then, we record the classification accuracy obtained by this layer as $pivot$ (line 2). The class vector aug obtained by the previous layer is then stitched with the original time series and used as the input to the next cascade layer (lines 4–5). The new accuracy rate of $newpivot$ obtained at each layer is compared with the previous $pivot$ (lines 7–13), and if the accuracy of the two consecutive layers is no longer improved (lines 11–12), the training will be automatically terminated to obtain the final model. The final classification accuracy of the model is the ratio between the class corresponding to the maximum value after averaging the last layer of class vectors and the real labels Y.

Given an instance, each forest will generate an estimate of the class distribution by calculating the percentage of training examples of different classes of related instances on the leaf node, and then stitching together the class vectors obtained by all trees in the same forest to calculate the average. The estimated class distribution forms a class vector, which is then stitched with the original data vector to obtain a new feature representation and input to the next layer for continued training. For example, suppose there are three classes, each of the four forests will produce a three-dimensional class vector. Therefore, the next layer will receive 12 (= 3 × 4) enhanced features.

Algorithm 1 CascadeForestClassifier(D, Y, t, s)

```
Require: the training set $D$, labels $Y$, the number of trees $t$, and the number of shapelet candidates $s$
Ensure: classification accuracy $pivot$
1: Built the first cascade layer to obtain the class vector using Algorithm 2 $aug = $ ClassificationCascadeLayer($D, Y, t, s$)
2: Set the reference performance $pivot$
3: while $NumLayers < MaxLayers$ do
4:     Concatenate $aug$ and $D$ as $X$
5:     Build a cascade layer to obtain the class vector using Algorithm 2 $aug = $ ClassificationCascadeLayer($X, Y, t, s$)
6:     Set the reference performance $newpivot$
7:     if $pivot \leq newpivot - \delta$ then // Check on early stopping
8:         Update the cascade layer
9:         $pivot = newpivot$
10:    else
11:        Activate early stopping if there are two layers' accuracy has not improved
12:        break
13:    end if
14: end while
15: return $pivot$
```

Algorithm 2 ClassificationCascadeLayer(X, Y, t, s)

```
Require: the representation $X$, labels $Y$, the number of trees $t$, and the number of shapelet candidates $s$
Ensure: class vector $aug$
1: $v_1 \leftarrow$ RandomForestClassifier($X, Y, t$)
2: $v_2 \leftarrow$ CompletelyRandomForestClassifier($X, Y, t$)
3: $v_3 \leftarrow$ ShapeletForestClassifier($X, Y, t, s$)
4: $v_4 \leftarrow$ DrCIF($X, Y, t$)
5: $aug = (||_{i=1}^{4} v_i)$
6: return $aug$
```

3.4 Complexity Analysis

For the proposed TSCF, the number of layers is N, and the number of decision trees in each forest is t. For a set of time series has m instances with length n, the time complexity of RF and CRF is $O(tmn \log(m))$ and $O(tmn)$, respectively. The computational cost for RSF is $O(tm^2sn^2 \log(msn^2))$, where s is the number of shapelet candidates. Note that the time complexity of DrCIF is $O(tmn \log(n))$. Therefore, the computational cost of each layer in TSCF depends on the highest time complexity of the four forests, that is $O(tm^2sn^2 \log(msn^2))$. Finally, multiplied by the number of layers N, the time complexity of TSCF should be $O(Ntm^2sn^2 \log(msn^2))$.

4 Experiments

In this section, we first describe the datasets and baseline methods used in our experiments, and then outline the parameter settings before evaluating the performance of TSCF in terms of accuracy and visualization. The source code of TSCF and more detailed experimental results of TSCF and other time series classification algorithms are available on our anonymous website https://anonymous.4open.science/r/Time-series-cascade-forest-D1DB.

4.1 Datasets and Baseline Methods

Experiments are conducted on 113 of the 128 public UCR datasets that are widely used in TSC studies. We have removed data with unequal length series or missing values, because most approaches cannot handle these scenarios. We experiment with a 4.10 GHz Intel Core i7-8750 H PC machine with 24 Gigabytes of memory. We use the open source software tool sktime[2] and its deep learning variant sktime-dl[3] that contain implementations of the majority of the existing algorithms we have compared.

Twenty-one different baselines are compared against, which are grouped into the following seven clusters.

- *Distance-based methods* employ similarity measures to quantify the distance between two series, such as EE [12] and shapeDTW [13].
- *Shapelet-based methods* distinguish differences between categories by extracting discriminating shapelets from time series, such as ST [17].
- *Interval-based methods* select one or more phase dependent intervals of the series and then using summary measures calculated by intervals as features, such as RISE [19].
- *Dictionary-based methods* form frequency counts of repetition of subseries, then use the histograms to build classifiers, such as BOSS [23] and WEASEL [24].
- *Hybrid methods* combine two or more of the single approaches, such as HIVE-COTE V2 [9], ROCKET [25], TS-CHIEF [26], and Catch22 [8].
- *Deep learning methods* employ nerual networks to TSC tasks, such as InceptionTime [32], FCN, MLP, and ResNet [30].
- *The classifiers use forests as the basic classification structure*, TSF [5], RSF [7], RPSF [14], Proximity Forest (PF) [34], STSF [20], CIF [6], and DrCIF [9].

We compare them separately with the TSCF. The results of the HIVE-COTE V2, TS-CHIEF, InceptionTime, and RPSF experiments were obtained directly from the official UCR website or the literature, and other methods were re-evaluated on 113 datasets, and the results are available on our supporting website. All experimental results are averaged over ten runs on the test set.

4.2 Parameter Analysis

Parameter settings have an important impact on the accuracy of the model. For the proposed TSCF, there are three factors, including the number of trees per forest t and the number of shapelet candidates for each node in RSF s. Due to the large number of datasets, we randomly selected 6 datasets from 113 UCR datasets with different numbers of time series instances, series lengths, and classes for experiments, including *BME*, *CBF*, *Chinatown*, *DiatomSizeReduction*, *GunPoint*, and *MoteStrain*, evaluated by 5-fold cross-validation on the training set. Figure 3 shows their classification accuracies across changes in $t \in \{25, 50, 75, 100, 125\}$ and $s \in \{5, 10, 15, 20, 25\}$.

[2] https://github.com/sktime/sktime.

[3] https://github.com/sktime/sktime-dl.

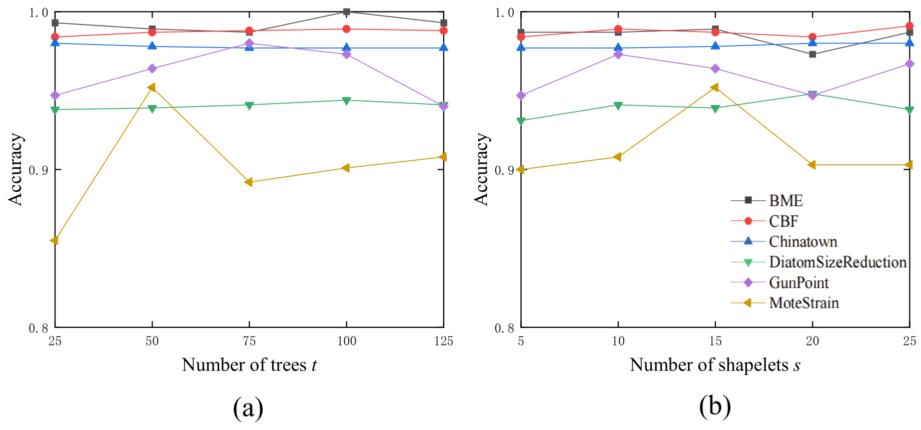

Fig. 3 Parameter analysis

As shown in Fig. 3, we can observe different trends in different datasets, and half of these datasets fluctuate moderately. Observing the dataset that is more sensitive to changes and considering the impact on training time, we set the parameters $t = 50$ and $s = 15$.

4.3 Classification Performance

In this section, we demonstrate that TSCF is competitive in term of classification accuracy contrast to state-of-the-art algorithms.

4.3.1 Compared with Forest-Based Benchmark Methods

We first compare benchmark forest-based methods, such as TSF [5], RSF [7], RPSF [14], PF [34], CIF [6], DrCIF [9], STSF [20], and gcForest [10] with our TSCF. Figure 4 shows the accuracy comparison between TSCF and the other 8 forest-based classifiers, and areas below the diagonal line indicate that TSCF is better.

It is clear that TSCF shows outstanding performance. Compared with TSF, RSF, RPSF, PF, CIF, DrCIF and STSF, TSCF performs better on 90, 94, 51, 97, 55, 71, 64 out of 113 (or 112/77) datasets, respectively. In addition, TSCF outperforms the original gcForest on 95 datasets. A more detailed version of the number of Wins/Draws/Losses is indicated by the W/D/L in the bottom-right corner of Fig. 4.

Figure 5 shows the critical difference (CD) diagram [35, 36] over the average ranks of the tested forest-based classification methods. The classifier with the lowest (best) rank lies in the upper right corner. The group of classifiers that are not significantly different is connected by a bar. The average ranking of TSCF is 2.7257 which is the lowest and there are no significant difference between the TSCF, CIF, and STSF. TSCF performs significantly better than DrCIF, RPSF, TSF, RSF, and PF.

From the experimental results, it can be proved that the direct use of gcForest for TSC tasks does not perform as well as existing forest-based methods, and the classifier that is more suitable for TSC based on its cascade structure combination can significantly improve the classification accuracy and perform better than the existing forest-based methods.

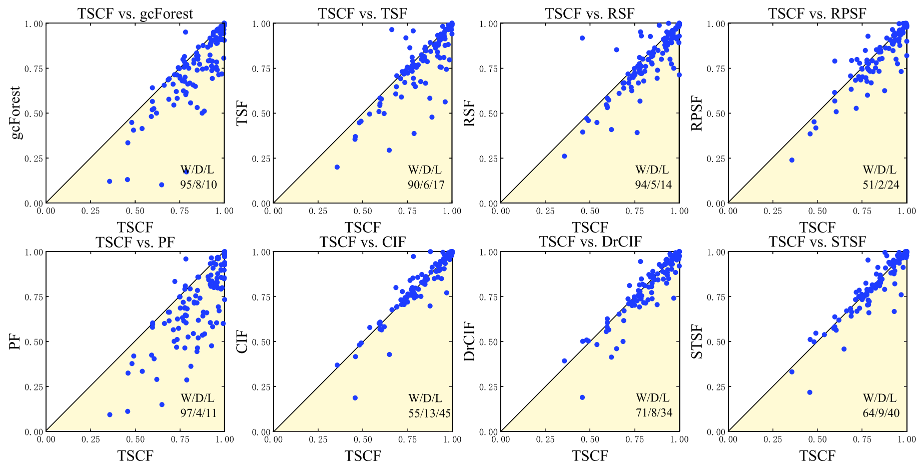

Fig. 4 Accuracy comparison between TSCF and other forest-based classifiers

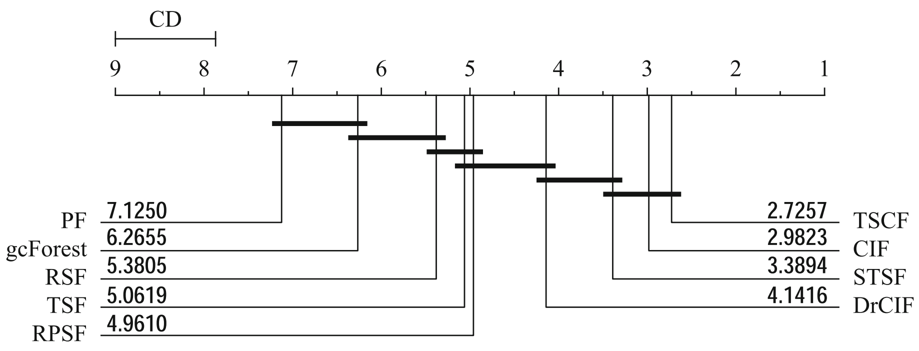

Fig. 5 CD diagram for comparisons among TSCF and other forest-based classifiers

Specific experimental results for TSCF and the 8 methods are shown in Table 1. The average accuracy of TSCF on 113 datasets is 0.839, followed by CIF with 0.832. For the 1-to-1 comparison on a single dataset, the accuracy is improved by more than 10% on 6 datasets compared with CIF, such as the accuracy of *PigAirwayP* and *PigCVP* is increased by 144.39% and 51.64%, respectively. The experimental results of RPSF came from their paper [14], which was not re-evaluated because it was written in Java and took a long time to train. The other methods are the result of re-runs, because each classifier in TSCF contains 50 trees, so the number of decision trees in several other forest-based methods is set to 200 (=50×4) for direct comparison with TSCF.

4.3.2 Compared with Distance/Shapelet/Interval/Dictionary-Based Methods

In addition to the forest-based methods, there are many traditional methods based on distance/shapelet/interval/dictionary to solve TSC problems. We compare TSCF with six related benchmarks. EE [12] and shapeDTW [13] are distance-based methods. Shapelet-based method we select ST [17]. RISE [19] is interval-based approaches. The dictionary-based methods we consider are BOSS [23] and WEASEL [24].

Table 1 Detailed UCR results for forest-based methods and TSCF

Datasets	TSF	RSF	RPSF	PF	STSF	CIF	DrCIF	gcForest	TSCF
ACSF1	0.667	0.820	–	0.520	0.869	0.844	**0.890**	0.800	**0.852**
Adiac	0.738	0.668	0.697	0.614	**0.824**	0.790	**0.783**	0.624	0.771
ArrowHead	0.731	0.691	0.799	**0.834**	0.668	0.778	0.811	0.714	0.721
Beef	0.790	0.567	0.600	0.647	0.703	**0.853**	0.778	0.700	0.780
BeetleFly	0.755	**1.000**	0.800	0.800	0.945	0.850	0.850	**0.950**	0.940
BirdChicken	0.880	0.800	0.860	0.660	**0.940**	0.850	0.917	0.750	0.900
BME	0.941	0.713	0.820	0.927	**1.000**	**1.000**	**1.000**	0.993	**1.000**
Car	0.765	0.733	0.792	0.793	0.815	**0.857**	**0.811**	0.667	0.807
CBF	0.990	0.974	0.949	0.896	0.984	**0.994**	**0.994**	0.909	0.980
Chinatown	0.973	0.971	–	0.927	0.979	0.980	**0.982**	**0.983**	0.981
ChlorineC	0.732	0.625	0.694	0.511	**0.785**	0.725	0.704	0.713	0.734
CinCECGT	0.983	0.800	0.775	0.825	0.979	**0.996**	0.988	0.750	0.948
Coffee	0.996	0.929	0.989	**0.993**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
Computers	0.718	0.644	0.703	0.468	**0.750**	0.706	0.719	0.620	0.731
CricketX	0.671	0.708	**0.778**	**0.625**	0.685	0.744	0.700	0.582	0.736
CricketY	0.694	0.721	0.744	**0.604**	0.730	0.772	0.744	0.603	**0.784**
CricketZ	0.662	0.762	0.776	**0.643**	0.705	**0.796**	0.737	0.615	0.786
Crop	0.754	0.740	–	0.686	0.768	**0.778**	0.754	0.755	**0.775**
DiatomSR	0.892	0.889	0.933	**0.965**	**0.952**	0.894	0.954	0.892	0.926
DistalPOAG	0.745	0.763	**0.849**	0.718	0.758	0.758	0.755	0.799	0.768
DistalPOC	0.765	0.775	**0.795**	0.464	0.789	0.778	0.779	0.786	0.777
DistalPTW	0.708	0.669	**0.793**	0.663	0.664	0.694	0.671	0.691	0.691
Earthquakes	**0.748**	0.748	0.748	0.748	**0.762**	0.748	0.748	0.748	0.748

Table 1 continued

Datasets	TSF	RSF	RPSF	PF	STSF	CIF	DrCIF	gcForest	TSCF
ECG200	0.867	0.840	0.866	**0.858**	0.852	0.860	**0.870**	0.810	0.850
ECG5000	0.940	0.940	0.940	0.938	**0.942**	**0.942**	0.937	0.937	0.938
ECGFiveD	0.965	0.994	**1.000**	0.897	0.981	0.978	0.959	0.806	0.997
ElectricD	0.692	0.718	–	0.565	0.733	0.732	**0.737**	0.661	0.713
EOGHS	0.509	0.547	–	0.424	0.573	**0.602**	**0.555**	0.481	0.591
EOGVS	0.495	0.448	–	**0.334**	0.538	**0.578**	0.483	0.414	0.539
EthanolL	0.498	0.409	–	0.289	0.619	0.582	0.413	0.500	**0.620**
FaceAll	0.754	0.747	0.752	**0.809**	0.779	0.755	0.763	0.778	0.780
FaceFour	0.951	0.955	**0.998**	0.852	0.995	0.982	0.902	0.784	0.957
FacesUCR	0.885	**0.911**	0.905	**0.843**	0.881	0.890	0.847	0.791	**0.911**
FiftyWords	0.733	0.727	0.677	**0.688**	0.771	0.787	0.692	0.677	**0.777**
Fish	0.808	0.891	0.931	0.808	0.893	**0.945**	0.869	0.789	**0.933**
FordA	0.810	0.909	–	0.604	**0.969**	0.905	0.928	0.756	0.955
FordB	0.676	0.798	–	0.544	0.797	0.751	0.732	0.657	**0.825**
FreezerRT	0.995	0.892	–	0.854	**0.999**	**0.999**	0.976	0.965	0.998
FreezerSlT	0.893	0.739	–	0.642	0.995	**1.000**	0.876	0.745	0.878
Fungi	0.926	0.952	–	–	0.667	0.906	**0.957**	0.737	**0.930**
GunPoint	0.965	0.967	**0.999**	0.792	0.912	0.972	0.984	0.940	0.980
GunPointAS	0.984	0.981	–	0.849	0.967	0.985	0.992	0.981	**0.997**
GunPointMVF	0.994	0.997	–	0.939	0.993	**1.000**	0.999	0.994	**1.000**

Table 1 continued

Datasets	TSF	RSF	RPSF	PF	STSF	CIF	DrCIF	gcForest	TSCF
GunPointOVY	**1.000**	**1.000**	–	0.983	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
Ham	**0.756**	0.724	0.745	0.749	0.733	0.720	0.702	0.705	**0.750**
HandOutlines	0.923	0.916	–	0.476	0.916	**0.927**	0.882	0.900	**0.927**
Haptics	0.447	0.471	0.452	0.377	**0.498**	0.481	**0.508**	0.448	0.481
Herring	0.584	0.531	0.568	0.603	0.637	0.594	0.604	**0.641**	0.597
HouseTwenty	0.877	0.960	–	0.798	0.960	0.934	0.930	0.740	**0.961**
InlineSkate	0.370	0.395	0.385	0.324	**0.511**	0.416	0.501	0.335	0.459
IEPGRT	**1.000**	**1.000**	–	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
IEPGST	**1.000**	**1.000**	–	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
IWingS	**0.965**	0.610	–	0.600	0.670	0.677	0.652	0.655	**0.663**
ItalyPD	0.563	0.947	0.959	0.964	**0.971**	0.963	0.955	0.961	0.962
LargeKA	0.769	0.760	0.731	0.362	0.803	0.791	0.714	0.557	**0.811**
L2	0.767	0.721	0.738	**0.682**	0.723	0.764	0.754	0.770	**0.790**
L7	**0.919**	0.699	0.685	**0.573**	0.745	0.737	0.712	0.753	0.742
Mallat	0.918	0.941	0.942	0.929	**0.972**	0.959	**0.979**	0.914	0.951
Meat	0.933	0.917	**0.965**	0.933	**0.943**	0.920	0.928	0.917	0.920
MedicalI	0.755	0.679	0.669	0.736	0.782	0.752	0.716	0.750	**0.801**
MPOAG	0.553	0.584	**0.790**	0.595	0.582	0.608	0.626	0.565	0.597
MPOC	0.814	0.780	0.733	0.715	0.830	0.836	0.805	0.835	**0.845**
MPTW	0.543	0.539	**0.615**	0.579	0.577	0.569	0.580	0.519	0.596
MixedSRT	0.926	0.931	–	**0.828**	0.936	0.953	0.949	0.898	**0.958**

Table 1 continued

Datasets	TSF	RSF	RPSF	PF	STSF	CIF	DrCIF	gcForest	TSCF
MixedSST	0.840	0.879	–	0.818	0.884	0.924	**0.930**	0.793	**0.919**
MoteStrain	0.878	0.876	0.848	0.807	0.923	0.919	**0.930**	0.887	**0.923**
NIFECGT1	0.881	0.866	–	0.869	**0.930**	0.927	0.881	0.878	0.923
NIFECGT2	0.907	0.879	–	**0.908**	0.938	**0.939**	0.899	0.905	0.935
OliveOil	0.863	0.833	0.908	0.860	**0.930**	**0.933**	0.889	0.867	0.907
OSULeaf	0.583	0.694	0.625	0.535	0.790	0.698	0.705	0.500	**0.876**
PhaOC	0.815	0.815	0.840	0.444	0.818	0.832	0.772	0.819	**0.845**
Phoneme	0.200	0.261	–	0.093	0.331	0.369	**0.392**	0.120	0.357
PigAirwayP	0.355	**0.918**	–	0.111	0.217	0.187	0.189	0.130	0.457
PigArtP	0.387	**0.930**	–	0.286	0.660	0.743	0.848	0.173	0.788
PigCVP	0.294	**0.853**	–	0.149	0.458	0.428	0.460	0.101	0.649
Plane	**1.000**	0.990	0.984	**0.971**	**1.000**	**1.000**	**1.000**	0.990	**1.000**
PowerCons	**1.000**	0.989	0.900	0.996	**1.000**	0.998	0.972	**1.000**	**1.000**
PPOAG	0.848	0.834	0.841	0.858	0.839	0.841	0.859	**0.868**	0.855
PPOC	0.844	0.849	0.863	0.663	**0.909**	0.886	0.843	0.876	0.881
PPTW	0.811	0.810	0.782	0.791	0.760	0.800	**0.815**	**0.815**	0.777
RefrD	0.579	0.573	–	0.404	0.563	0.565	0.565	0.525	**0.605**
Rock	**0.864**	0.392	–	0.632	0.752	0.840	0.727	0.680	0.764
ScreenType	0.455	0.459	–	0.419	**0.546**	0.492	0.503	0.405	0.491
SemgHGCh2	0.940	0.774	–	0.807	**0.972**	0.948	0.937	0.920	0.956
SemgHMCh2	**0.858**	0.774	–	0.784	0.810	0.846	0.738	0.531	0.848

Table 1 continued

Datasets	TSF	RSF	RPSF	PF	STSF	CIF	DrCIF	gcForest	TSCF
SemgHSCh2	0.904	0.851	–	0.836	0.877	**0.926**	0.824	0.762	**0.929**
ShapeletSim	0.478	**0.989**	0.918	0.542	0.977	0.913	**0.928**	0.511	0.888
ShapesAll	0.787	**0.863**	0.804	0.741	0.847	0.847	0.844	0.772	**0.862**
SmallKA	0.803	0.821	0.800	0.464	0.830	0.818	0.824	0.739	**0.848**
SmoothS	0.988	0.997	–	0.980	0.979	0.991	0.978	0.987	**1.000**
SAIBORS1	0.792	**0.927**	0.852	0.754	0.912	0.824	0.859	0.649	0.838
SAIBORS2	0.825	0.836	0.853	0.714	**0.875**	0.873	**0.872**	0.796	0.830
StarLightC	0.972	0.968	0.972	0.856	0.980	0.980	0.974	0.957	**0.981**
Strawberry	0.966	0.920	0.934	0.693	0.961	**0.970**	0.960	0.968	0.965
SwedishLeaf	0.917	0.892	0.913	0.831	0.945	**0.957**	0.955	0.888	**0.957**
Symbols	0.914	0.956	0.877	0.877	0.961	0.956	**0.978**	0.890	0.961
SyntheticC	0.989	0.996	0.981	**0.998**	0.987	0.991	0.978	0.943	**0.997**
ToeS1	0.757	0.917	**0.946**	0.582	0.821	0.900	0.883	0.693	0.933
ToeS2	0.817	0.900	0.886	0.605	0.882	0.858	0.882	0.823	**0.920**
Trace	0.991	0.990	**1.000**	0.600	**1.000**	**1.000**	**0.993**	0.910	0.992
TwoLeadECG	0.797	0.912	**0.994**	0.799	0.983	0.956	**0.954**	0.718	0.986
TwoPatterns	0.991	0.997	0.979	**0.733**	0.995	0.999	0.920	0.905	**1.000**
UMD	0.882	0.924	0.946	0.899	0.966	**0.982**	0.884	0.917	**0.979**
UWGLAll	0.958	0.952	0.928	**0.959**	0.953	**0.973**	0.945	0.951	0.783

Table 1 continued

Datasets	TSF	RSF	RPSF	PF	STSF	CIF	DrCIF	gcForest	TSCF
UWGLX	0.800	0.799	0.780	0.719	0.817	**0.829**	0.803	0.778	**0.829**
UWGLY	0.723	0.714	0.702	**0.686**	0.739	**0.764**	0.731	0.694	**0.752**
UWGLZ	0.744	0.750	–	0.671	0.760	0.771	0.741	0.714	**0.969**
Wafer	0.997	0.988	0.992	0.989	**1.000**	0.998	0.998	0.994	0.999
Wine	0.607	0.667	0.707	0.663	0.683	**0.800**	**0.500**	0.685	0.685
WordS	0.647	0.639	0.633	**0.615**	0.641	**0.692**	0.616	0.577	0.689
Worms	0.590	0.668	0.527	0.501	**0.774**	0.706	0.710	0.545	0.717
WormsTC	0.648	0.801	0.733	0.525	0.791	0.797	**0.818**	0.662	**0.795**
Yoga	0.848	0.841	0.853	**0.618**	0.826	**0.861**	0.853	0.819	0.853
Avg acc	0.791	0.804	0.819	0.701	0.827	0.832	0.815	0.750	**0.839**
Avg rank	5.0619	5.3805	4.9610	7.1250	3.3894	2.9823	4.1416	6.2655	**2.7257**
1-to-1 Wins	90	94	51	97	64	55	71	95	–
1-to-1 Draws	6	5	2	4	9	13	8	8	–
1-to-1 Losses	17	14	24	11	40	45	34	10	–

The best accuracy for each dataset is marked with bold

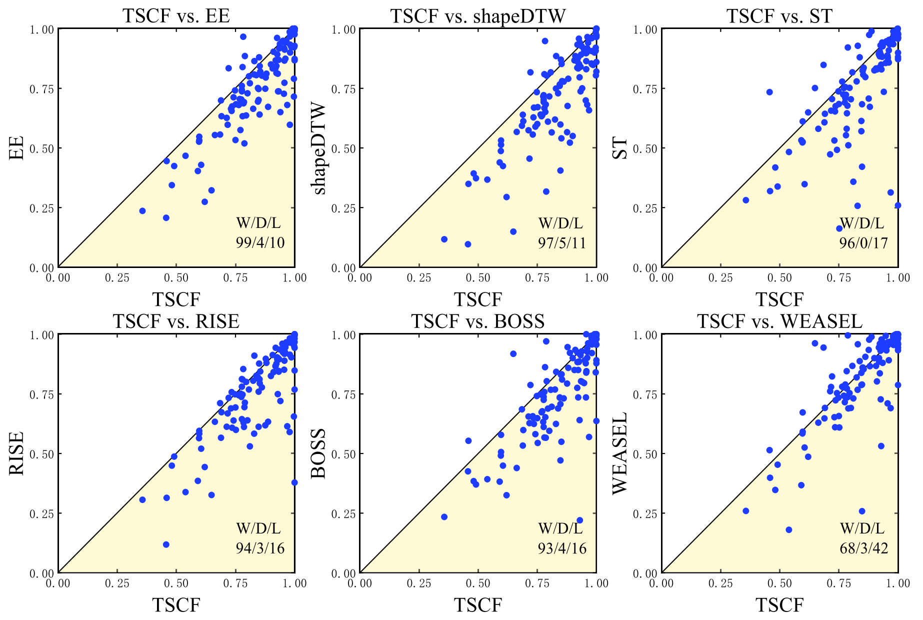

Fig. 6 Accuracy comparison between TSCF and distance/shapelet/interval/dictionary-based benchmark classifiers

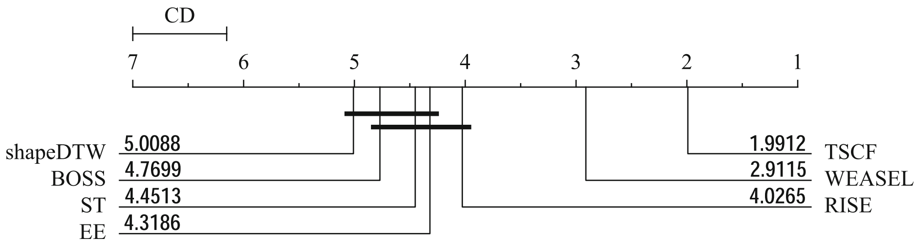

Fig. 7 CD diagram for the average ranking comparison among TSCF and distance/shapelet/interval/dictionary-based classifiers

The analysis in Figs. 6 and 7 reveals that TSCF performs significantly better than these six methods. The experimental accuracy results for TSCF and other methods are available on our anonymous website.

4.3.3 Compared with Hybrid Benchmark Methods

In this section, we try to analyze TSCF with state-of-the-art hybrid methods. Hybrid methods include HIVE-COTE V2 [9], ROCKET [25], TS-CHIEF [26], and Catch22 [8]. As shown in Figs. 8 and 9, although our method is worse than HIVE-COTE V2 and TS-CHIEF, it is competitive with ROCKET and better than Catch22. In addition, TSCF consists of only four forest-based classifiers and each forest contains 50 trees. But, TS-CHIEF uses PF, BOSS, and RISE as node splitting functions for 500 decision trees, which is more complex than TSCF, taking into account distance, dictionary, and interval-based spectral features. HIVE-COTE V2 contains four component classifiers: the dictionary based temporal dictionary

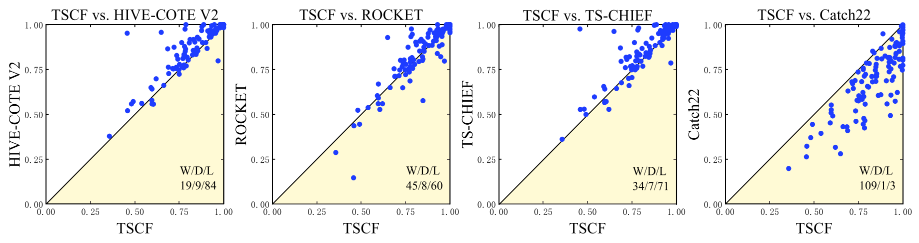

Fig. 8 Accuracy comparison between TSCF and hybrid classifiers

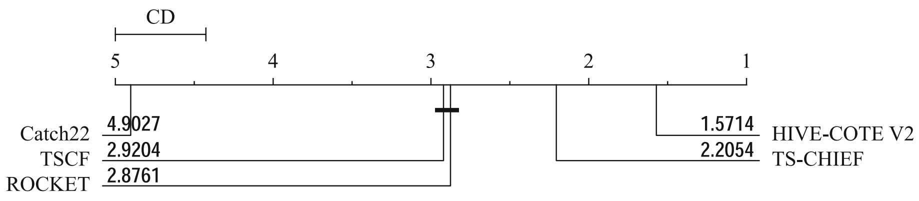

Fig. 9 CD diagram for the comparison among TSCF and hybrid classifiers

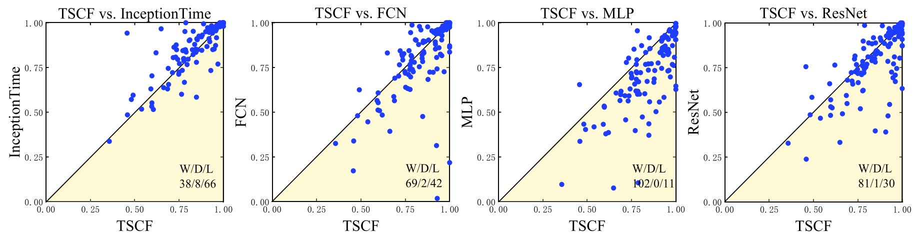

Fig. 10 Accuracy comparison between TSCF and deep learning classifiers

ensemble (TDE); the interval based DrCIF; an adaptation of ROCKET and the latest version of ST. Therefore, TSCF's performance not exceeding the two of them is to be expected. Since TSCF uses DrCIF as a base classifier, and the 29 summary statistics in DrCIF include Catch22, TSCF's classification ability is better than Catch22.

4.3.4 Compared with Deep Learning Benchmark Methods

We compared four deep learning models, which are InceptionTime [32], FCN [37], MLP [38], and ResNet [30] with TSCF. As shown in Figs. 10 and 11, TSCF is competitive with InceptionTime, and do better performance than FCN, MLP and ResNet. This proves that the construction of deep forests based on the cascade structure of gcForest can obtain performance comparable to DNNs.

4.4 Variants of TSCF

We include several variants of the proposed TSCF. For the four forest-based classifiers used in TSCF, we conducted different combinations, including: (1) replacing DrCIF with PF; (2) one each for RF and CRF, two for RSF; (3) one each for RF and CRF, two for DrCIF; (4) two RSFs and two DrCIFs.

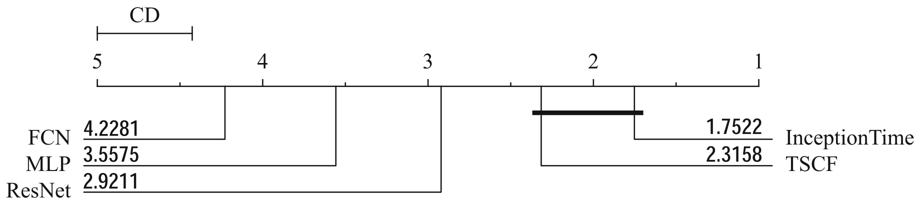

Fig. 11 CD diagram for the comparison among TSCF and deep learning classifiers

Table 2 Experimental comparison of the average accuracy of TSCF and its variants in 25 datasets

Methods	Avg acc
RF, CRF, RSF, PF	0.893
RF, CRF, RSF×2	0.876
RF, CRF, DrCIF×2	0.897
RSF×2, DrCIF×2	0.894
TSCF (RF, CRF, RSF, DrCIF)	0.899

As shown in Table 2, the experiment was conducted on 25 datasets. The results showed that TSCF has better classification performance (0.899 in average). Compared with other combinations of the four forest-based classifiers, the accuracy of TSCF is the highest, which validates the effectiveness of multi-granular representation learning of time series.

4.5 Ablation Study

We implement ablation studies to understand the contribution of four different base classifiers. The settings are summarized as follows: (1) w/o RF means abandoning the RF classifier, only using the other three models; (2) w/o CRF represents not using the CRF; (3) w/o RSF indicates deleting the RSF module; (4) w/o DrCIF denotes only using RF, CRF, and RSF to constitute a cascade layer; (5) w/o C means that the output of each layer in the model does not concatenate the original time series, and is directly used as the input to the next layer.

Table 3 shows the average accuracy of TSCF and its ablation variants on 25 datasets, and it turns out that all four base classifiers combined with TSCF contribute to the whole, allowing TSCF to achieve better performance than individuals. Among them, the existence of DrCIF improved model performance by 2.860%, and RSF by 1.011%. RF and CRF are also indispensable roles, which can increase the diversity of models and sample randomness. From w/o C, we can conclude that after the training of each layer, it is necessary to concatenate the output of this layer with the original time series as the input to the next layer. In this way, the layer-by-layer classifier can further learn from the original data after obtaining the learning results of the previous layer. Overall, all variants of the model perform slightly worse than the full TSCF, which also proves the effectiveness of each base classifier.

4.6 Visualization

We visualized the learned representation to show that TSCF effectively captures the underlying structure among different classes of time series. In this section, we randomly selected 5 of the 113 UCR datasets, including *Meat*, *UMD*, *Mallat*, *MoteStrain*, and *Symbols*. First,

Table 3 Ablation results of TSCF on 25 datasets

Datasets	w/o RF	w/o CRF	w/o RSF	w/o DrCIF	w/o C	TSCF
ArrowHead	0.731	0.695	0.726	0.693	0.742	0.721
Beef	0.780	0.780	0.767	0.720	0.747	0.780
BeetleFly	0.920	0.950	0.880	0.880	0.800	0.940
BirdChicken	0.830	0.900	0.900	0.760	0.920	0.900
BME	0.997	1.000	1.000	0.967	0.997	1.000
CBF	0.981	0.978	0.978	0.957	0.975	0.980
Chinatown	0.975	0.977	0.983	0.980	0.980	0.981
Coffee	0.993	1.000	1.000	1.000	1.000	1.000
DiatomSR	0.918	0.925	0.932	0.933	0.929	0.926
ECG200	0.830	0.838	0.870	0.818	0.848	0.850
ECGFiveD	0.995	0.994	0.994	0.971	0.958	0.997
GunPoint	0.983	0.973	0.981	0.957	0.979	0.980
Ham	0.724	0.758	0.743	0.756	0.760	0.750
ItalyPD	0.958	0.962	0.960	0.959	0.955	0.962
Lightning2	0.738	0.780	0.748	0.793	0.790	0.790
MedicalI	0.764	0.787	0.795	0.747	0.772	0.801
MoteStrain	0.924	0.924	0.912	0.898	0.866	0.923
PPOAG	0.851	0.851	0.854	0.862	0.845	0.855
PPOC	0.882	0.880	0.878	0.869	0.880	0.881
Symbols	0.964	0.961	0.952	0.937	0.853	0.961
ToeS1	0.941	0.932	0.876	0.927	0.924	0.933
ToeS2	0.926	0.915	0.883	0.912	0.906	0.920
TwoLECG	0.944	0.987	0.985	0.900	0.929	0.986
UMD	0.982	0.988	0.993	0.976	0.989	0.979
Wine	0.759	0.678	0.667	0.689	0.748	0.685
Avg acc	0.892	0.897	0.890	0.874	0.875	**0.899**
Avg difference	−0.785%	−0.223%	−1.011%	−2.860%	−2.743%	–

we use the t-SNE visual algorithm [39] to compare the learned representations. As shown in Fig. 12, each column represents the test result of a dataset. The first row is a visualization of the original data. The second is the representation of gcForest. The last is our method. For gcForest and TSCF, we used the splicing class vectors of the four forests output by the last layer as the representations learned by the model and then used them as the input of t-SNE visualization. Figure 12 shows that TSCF can learn class embeddings with larger inter-class distances and compact intra-classes distributions. These results also illustrate the effectiveness of TSCF in learning data characteristics from different granularities. Then, we analyse the learned representation of TSCF with a heatmap. Figure 13 indicates that TSCF can learn distinguishable features among different classes.

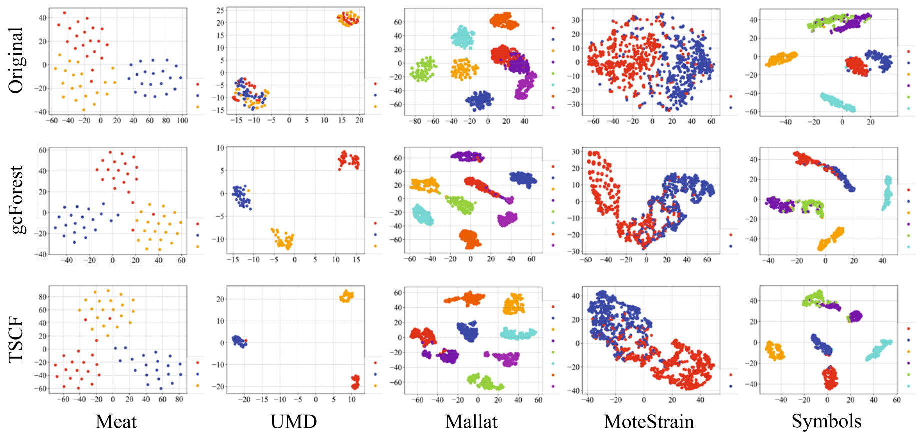

Fig. 12 The t-SNE visualization of original, gcForest, and TSCF (from top to bottom) representation space on 5 datasets. Different colors represent different classes in each subgraph, and each dot represents one instance

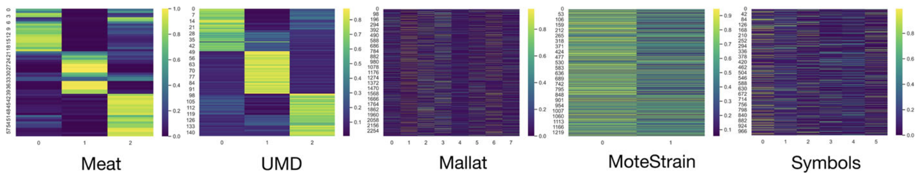

Fig. 13 The heatmap visualization of representations learned by TSCF on 5 datasets. Each column in the subgraph represents a class in the dataset

5 Conclusion

This paper proposes TSCF, an ensemble model based on the deep forest for TSC tasks. TSCF performs feature learning on the original time series at point-level, subsequence-level, and the summary statistics of intervals, using four different forests to capture more sequence information. The extensive evaluation of TSCF demonstrates its effectiveness, and visualization of learned representations validates its capability to capture different granularities of time series. Based on the framework of TSCF, other base classifiers for point, subsequence and interval-level can be considered for multi-granular representation learning in practical applications, such as TSF [5], RPSF [14], etc. Our future work will focus on including more efficient base classifiers to enhance the performance of cascade forests.

While TSCF exhibits the capability to learn from time series data at various granularities, it is constrained by the practice of concatenating the learning outcomes of the forest with the original sequence, serving as the input for the subsequent layer. Furthermore, each layer necessitates a reiteration of learning on the raw data, incurring a time overhead. In future endeavors, it is imperative to explore strategies for efficiently processing data within the cascading structure and establishing effective data transmission mechanisms between layers. This is essential to enable the classifier to consistently and comprehensively grasp the distinctive features of the data.

Acknowledgements This work was supported by the National Key R&D Program of China (No. 2022YFE0200400) and the Fundamental Research Funds for the Central Universities (No. 2022JBMC011).

Author Contributions MD: Conceptualization, methodology, software, validation, formal analysis, investigation, data curation, writing-original draft, writing-review and editing, visualization. JY: formal analysis, resources, writing-reviewing and editing, supervision, project administration, funding acquisition. HL: writing-reviewing and editing, supervision, project administration. JW: investigation, data curation.

Declarations

Conflicts of interest The authors declare no competing interests.

References

1. Pazzani MJ, et al (2001) Derivative dynamic time warping. In: Proceedings of the 2001 SIAM international conference on data mining. Society for Industrial and Applied Mathematics
2. Yuan J, Lin Q, Zhang W, et al (2019) Locally slope-based dynamic time warping for time series classification. In: Proceedings of the 28th ACM international conference on information and knowledge management, pp 1713–1722
3. Jeong YS, Jeong MK, Omitaomu OA (2011) Weighted dynamic time warping for time series classification. Pattern Recogn 44(9):2231–2240
4. Ye L, Keogh E (2009) Time series shapelets: a new primitive for data mining. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining, pp 947–956
5. Deng H, Runger G, Tuv E et al (2013) A time series forest for classification and feature extraction. Inf Sci 239:142–153
6. Middlehurst M, Large J, Bagnall A (2020) The canonical interval forest (CIF) classifier for time series classification. In: 2020 IEEE international conference on big data (Big Data), IEEE, pp 188–195
7. Karlsson I, Papapetrou P, Boström H (2016) Generalized random shapelet forests. Data Min Knowl Disc 30:1053–1085
8. Lubba CH, Sethi SS, Knaute P et al (2019) catch22: canonical time-series characteristics: Selected through highly comparative time-series analysis. Data Min Knowl Disc 33(6):1821–1852
9. Middlehurst M, Large J, Flynn M et al (2021) Hive-cote 2.0: a new meta ensemble for time series classification. Mach Learn 110(11–12):3211–3243
10. Zhou ZH, Feng J (2019) Deep forest. Natl Sci Rev 6(1):74–86
11. Stefan A, Athitsos V, Das G (2012) The move-split-merge metric for time series. IEEE Trans Knowl Data Eng 25(6):1425–1438
12. Lines J, Bagnall A (2015) Time series classification with ensembles of elastic distance measures. Data Min Knowl Disc 29:565–592
13. Zhao J, Itti L (2018) shapedtw: shape dynamic time warping. Pattern Recogn 74:171–184
14. Yuan J, Shi M, Wang Z et al (2022) Random pairwise shapelets forest: an effective classifier for time series. Knowl Inf Syst 64:1–32
15. Grabocka J, Schilling N, Wistuba M, et al (2014) Learning time-series shapelets. In: Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining, pp 392–401
16. Zhang Z, Zhang H, Wen Y et al (2018) Discriminative extraction of features from time series. Neurocomputing 275:2317–2328
17. Lines J, Davis LM, Hills J, et al (2012) A shapelet transform for time series classification. In: Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining, pp 289–297
18. Lines J, Taylor S, Bagnall A (2016) Hive-cote: The hierarchical vote collective of transformation-based ensembles for time series classification. In: 2016 IEEE 16th international conference on data mining (ICDM), IEEE, pp 1041–1046

19. Lines J, Taylor S, Bagnall A (2018) Time series classification with hive-cote: the hierarchical vote collective of transformation-based ensembles. ACM Trans Knowl Discov Data 12(5):1–35
20. Cabello N, Naghizade E, Qi J, et al (2020) Fast and accurate time series classification through supervised interval search. In: 2020 IEEE international conference on data mining (ICDM), IEEE, pp 948–953
21. Baydogan MG, Runger G, Tuv E (2013) A bag-of-features framework to classify time series. IEEE Trans Pattern Anal Mach Intell 35(11):2796–2802
22. Baydogan MG, Runger G (2016) Time series representation and similarity based on local autopatterns. Data Min Knowl Disc 30:476–509
23. Schäfer P (2015) The boss is concerned with time series classification in the presence of noise. Data Min Knowl Disc 29:1505–1530
24. Schäfer P, Leser U (2017) Fast and accurate time series classification with weasel. In: Proceedings of the 2017 ACM on conference on information and knowledge management, pp 637–646
25. Dempster A, Petitjean F, Webb GI (2020) Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. Data Min Knowl Disc 34(5):1454–1495
26. Shifaz A, Pelletier C, Petitjean F et al (2020) Ts-chief: a scalable and accurate forest algorithm for time series classification. Data Min Knowl Disc 34(3):742–775
27. Middlehurst M, Schäfer P, Bagnall A (2023) Bake off redux: a review and experimental evaluation of recent time series classification algorithms. arXiv preprint arXiv:2304.13029
28. Le Guennec A, Malinowski S, Tavenard R (2016) Data augmentation for time series classification using convolutional neural networks. In: ECML/PKDD workshop on advanced analytics and learning on temporal data
29. Cui Z, Chen W, Chen Y (2016) Multi-scale convolutional neural networks for time series classification. arXiv preprint arXiv:1603.06995
30. He K, Zhang X, Ren S, et al (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
31. Tanisaro P, Heidemann G (2016) Time series classification using time warping invariant echo state networks. In: 2016 15th IEEE international conference on machine learning and applications (ICMLA), IEEE, pp 831–836
32. Ismail Fawaz H, Lucas B, Forestier G et al (2020) Inceptiontime: finding alexnet for time series classification. Data Min Knowl Disc 34(6):1936–1962
33. Ismail Fawaz H, Forestier G, Weber J et al (2019) Deep learning for time series classification: a review. Data Min Knowl Disc 33(4):917–963
34. Lucas B, Shifaz A, Pelletier C et al (2019) Proximity forest: an effective and scalable distance-based classifier for time series. Data Min Knowl Disc 33(3):607–635
35. Benavoli A, Corani G, Demšar J et al (2017) Time for a change: a tutorial for comparing multiple classifiers through Bayesian analysis. J Mach Learn Res 18(1):2653–2688
36. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
37. Zhao B, Lu H, Chen S et al (2017) Convolutional neural networks for time series classification. J Syst Eng Electron 28(1):162–169
38. Wang Z, Yan W, Oates T (2017) Time series classification from scratch with deep neural networks: A strong baseline. In: 2017 international joint conference on neural networks (IJCNN), IEEE, pp 1578–1585
39. Van der Maaten L, Hinton G (2008) Visualizing data using t-SNE. J Mach Learn Res 9(11):2579–2605