International Journal of Machine Learning and Cybernetics (2021) 12:2213–2236
https://doi.org/10.1007/s13042-021-01302-y

ORIGINAL ARTICLE

DEP-TSP^{meta}: a multiple criteria Dynamic Ensemble Pruning technique ad-hoc for time series prediction

Jing Zhang[1] · Qun Dai[1] · Changsheng Yao[1]

Received: 22 May 2020 / Accepted: 8 March 2021 / Published online: 19 March 2021

Abstract
Time series prediction (TSP) is a process of using data collected at different times in the past for statistical analysis, so as to speculate on the trend of things, where the non-stationary and non-linear characteristics of data portray a hard setting for predictive tasks. Obviously, there will be no single model that could perform the best for all TSP issues. Dynamic Ensemble Selection (DES) technique achieves more accurate and robust performance than a single model, due to that it aims to select an ensemble of the most competent models in a dynamic fashion according to each test sample. A variety of DES approaches have been proposed to address pattern classification problems, but little work has been conducted on the research of TSP adopting the DES paradigm. Commonly, the DES approaches work by the definition of a single criterion to evaluate the capability of base classifiers. However, only one criterion is often inadequate for the comprehensive evaluation of classifier power. Thus, in this paper, a multiple criteria Dynamic Ensemble Pruning (DEP) technique exploiting *meta*-learning ad-hoc for TSP, termed DEP-TSP^{meta}, based on the inspiration from a state-of-the-art META-DES framework specifically presented for classification tasks, is developed. Within DEP-TSP^{meta}, Extreme Learning Machines (ELMs) and Hierarchical Extreme Learning Machines (H-ELMs) are integrated as the base models, and four distinct meta-attributes collections, i.e., hard prediction, local accuracy, global accuracy, and prediction confidence, are presented. Each set of meta-attributes corresponds to a specific assessment criterion, i.e., the prediction accuracy in local area of the eigenspace, the overall local accuracy, the prediction accuracy in global area of the decision space, and the confidence level of predictor. A desirable meta-predictor, obtained by training on the strength of these meta-attributes, is the key to deciding whether a base predictor is capable of predicting the unseen instance well or not. Those incapable base predictors determined by the meta-predictor will be pruned and the capable predictors will be expanded into the final dynamic ensemble system. The size of the sets of meta-attributes is specified dynamically by genetic algorithm for different time series benchmark datasets. Empirical results on eight benchmark datasets with different time granularities have verified that, the proposed DEP-TSP^{meta} algorithm possesses dramatically improved prediction performance at different granularities, when compared against three other DES approaches and four static selective ensemble learning methods.

Keywords Dynamic Ensemble Pruning (DEP) · Time series prediction (TSP) · Meta-learning · Meta-predictor · DEP exploiting *meta*-learning for TSP (DEP-TSP^{meta})

1 Introduction

A time series is an assemblage of data points acquired by sampling at equal intervals. Time series prediction (TSP) is a process to predict data future values exploiting knowledge learned from past and current values of data associated with a particular phenomenon [1]. In the digital information age, TSP algorithms have been extensively adopted in various data mining fields, including economics [2, 3], physical sciences [4, 5], and engineering [6].

TSP methods can be divided into traditional linear models, e.g., Auto Regressive (AR) and Auto Regressive Moving Average (ARMA), and nonlinear models, e.g., neural networks (NNs). Some restrictive assumptions are required by these linear models, such as linearity, smoothness and normality, which are rarely satisfied, due to the characteristics

✉ Qun Dai
daiqun@nuaa.edu.cn

[1] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China

of nonlinearity and chaos in time series. Though NNs are more effective than traditional linear models, NNs still have their own demerits, including time-consuming learning process, slow convergence, and being prone to get into local minima [7].

Huang et al. [8] proposed a learning scheme called extreme learning machine (ELM), which is a special case of Single-hidden Layer Feedforward Neural Networks (SLFNs), to improve the above mentioned disadvantages of NNs. For an ELM model, it is not necessary to set stopping criteria, learning rate, and learning epochs. ELM can offer good generalization performance with fast learning speed. Many research works [9–11] have proved that ELM is suitable for addressing various types of classification and regression issues. However, when handling natural scenes or practical applications, due to its shallow architecture, feature representation learning using ELM may not be effective. Huang et al. [12] further extended ELM and proposed a Hierarchical ELM (H-ELM) framework for Multi-Layer Perceptrons (MLPs). H-ELM reserves ELM's advantage of training efficiency, while possessing superior generalization performance to the classical ELM, simultaneously.

What's more, NNs might be somewhat unstable with randomicity. Ensemble learning paradigm is thus proposed to utilize the unique capability of each component model for better capturing different characteristics in data, which brings remarkable advancements in NNs. Numerous theoretical analysis and experimental results [13–15] have shown that the combination of different models can significantly improve the predictive performance of a single model, particularly in the cases that the base learners in an ensemble are adequately complementary and diverse [16, 17]. Diversified ensembles could be generated by utilizing diversified initial weight matrices, variable numbers of hidden nodes, or various activation functions [18]. In this work, both shallow learning models, i.e., ELMs, and deep learning models, i.e., H-ELMs, will be integrated together to form the original ensemble. The original ensembles built in this way are compounded, and possess relatively high complementarity and diversity intuitively.

However, an initial ensemble of base learners is certainly not always optimal for its prediction or classification tasks, while removing the incompetent base learners from an initial ensemble can improve its predictive or classification performance in many cases. In view of that each base learner has its own unique capabilities, it is unreasonable to always underestimate or deny one specific learner, which may have poor performance on some samples, but good performance on the other ones.

To address the above issues, the Dynamic Ensemble Selection (DES) paradigm is proposed for pattern classification tasks. With DES, only the classifiers obtaining a certain competence level for the given test sample, according to a selection criterion or several selection criteria, are dynamically selected into the ensemble currently constructed for the given test sample. Recently, many remarkable achievements and breakthroughs have been made. In [19], a novel probabilistic model for dynamic ensemble selection is proposed. In this model, an optimal subset is obtained by simultaneously measuring diversity and classifier competence. In [20], a creative approach, called dynamic multistage organization (DMO), is proposed by Cavalin et al. It is based on multistage organizations and respectively designs the optimal multistage structure for each unseen sample. In [21], to alleviate the problem of selecting classifiers which overfit the local region in typical DES, new modifications are proposed to improve the generalization performance of DES methods.

Motivated by its preferable performance on pattern recognition, we carry out research on applying the DES paradigm to time series prediction. However, the measure criteria of classifier competence in the above methods are not equally applicable to predictor. With TSP, the key point of the DES is how to evaluate the capability of a base predictor. According to our previous work [22], three novel DES algorithms have been proposed for TSP, including the DES algorithm based on Predictor Accuracy over the Local Region (DES-PALR), the DES algorithm based on the Consensus of Predictors (DES-CP), and the Dynamic validation set determination algorithm based on the similarity between the Output profile of the test sample and the Output profile of each training sample (DVS-OpOp).

All of the above algorithms define only one criterion, such as the local precision in the feature space or the global precision in the decision space, to measure the capability of a base predictor or ensembles of predictors (EoPs) to implement DES tasks. However, it is not sufficient to comprehensively evaluate the competence level of a base predictor merely by utilizing one criterion. Using a single standard to estimate the competence level of a base predictor is one-sided, and thus error prone. Therefore, multiple criteria to measure the competence of a base predictor should be considered in order to achieve a more robust DES technique.

In [23], an updated taxonomy of dynamic selection systems for classification problem is presented. A comparative study shows that the algorithms designed based on *meta*-learning can generally achieve favorable classification performance. One DES framework using *meta*-learning, called META-DES, is proposed in [24], where a meta-classifier is trained with multiple criteria to predict whether a base classifier is sufficient to classify a test sample. So even though one criterion does not work well, the framework can still get desirable capability, as the other criteria are also taken into consideration. Then, the authors of [25] assessed the influence of the meta-classifier and an extension algorithm of META-DES. The experimental results demonstrate that the performance of the meta-classifier and the classification

accuracy of the DES system are strongly correlated. Moreover, in [26], more meta-features are considered, and a meta-feature selection scheme using a Binary Particle Swarm Optimization (BPSO) is applied, in order to improve the performance of the meta-classifier. It is believed that, the whole capability of the framework will be improved, when the identification ability of the meta-classifier is improved by choosing more suitable sets of meta-features or classifier models to train the meta-classifier.

Enlightened by the researches on the META-DES framework [24–26] for classification tasks introduced above, we construct our own multiple criteria Dynamic Ensemble Pruning technique based on the *meta*-learning paradigm, specifically for solving TSP problems, called the DEP-TSP^{meta} technique. The key point of the DEP-TSP^{meta} technique is to construct a meta-predictor, which is responsible for determining whether a base predictor has the ability to predict an unseen instance well or not. The unqualified base predictors will be pruned, while the eligible base predictors will be extended to the final dynamic ensemble system.

The motivations behind the development of DEP-TSP^{meta}, and simultaneously, the contributions of this work for TSP tasks, as well as the essential differences between META-DES [24–26] and our proposed DEP-TSP^{meta} technique, are summarized as follows.

Firstly, as is known to all that, the more diverse and informative become the generated base models, the more successful the ensemble system will be. Hence, in the ensemble initialization stage of DEP-TSP^{meta}, ELMs and H-ELMs, as base predictors, which are complementary and diverse, constitute the initial collection of predictors for TSP problems. Both shallow models, i.e., ELMs, and deep models, i.e., H-ELMs, are combined together to establish the initial ensemble. In this sense, the initial ensemble is diversified, which might contribute to the predictive performance advancement acquired by the final dynamically pruned ensemble.

Secondly, in the meta-predictor construction stage of DEP-TSP^{meta}, we design four groups of meta-attributes, instead of only consider a single criterion, that have proven effective respectively in our previous research results [22]. These newly formed four groups of meta-attributes are entirely different from the meta-features in META-DES [24–26]. Besides, ELM is employed as the meta-predictor model, given its good generalization performance, the ability of fast learning and effective avoidance of local minima issues.

Thirdly, the ensemble pruning process within DEP-TSP^{meta} is implemented dynamically by a meta-predictor trained on basis of the meta-attributes, instead of probabilistic-based approaches or data handling-based approaches.

Fourthly, the parameters utilized in the meta-predictor learning procedure of DEP-TSP^{meta} are adapted dynamically by genetic algorithm, which markedly boosts its predictive performance.

Fifthly, in the prediction stage of DEP-TSP^{meta}, based on the data characteristic of time series, the average of the predicted values produced by the selected predictors, rather than the results obtained by using the majority voting rule for classification problem in META-DES, is taken as the final algorithm output.

Last but not least, the DEP-TSP^{meta} technique proposed in the work is developed specifically for TSP application scenarios, significantly differing from the META-DES framework presented in [24–26], which is focused on the problem of pattern classification. The effectiveness of DEP-TSP^{meta} in handling with TSP problems has been proven by the empirical results conducted based upon eight TSP benchmark datasets with distinct time granularities, such as year, month, quarter, and so on. In comparison with three DES approaches and four Static Ensemble Selection (SES) ones, DEP-TSP^{meta} achieves superior forecasting performance at various levels of granularities.

The remaining of this paper is arranged as follows. Section 2 presents some important principles of the ELM algorithm and H-ELM algorithm. Section 3 discusses the notion of predictor competence for DES. Section 4 describes the details of the proposed DEP-TSP^{meta} technique. The experimental investigation is carried out in Sect. 5. Finally, the conclusion and prospect of future work are presented in Sect. 6.

2 Preview of ELM and H-ELM

This section will briefly review the ELM and H-ELM algorithms, which will be utilized in the proposed DEP-TSP^{meta} technique as the base predictors.

2.1 Extreme Learning Machine (ELM)

This section describes the background knowledge of ELM (Fig. 1). The ELM model proposed by Huang et al. [8] can be built by using randomly or artificially given hidden node parameters that do not need to be further adjusted. The training dataset is denoted as $\{(\boldsymbol{x}_i, \boldsymbol{t}_i)|\boldsymbol{x}_i \in R^d, \boldsymbol{t}_i \in R^m, i = 1, \ldots, N\}$, where $\boldsymbol{x}_i$ is the feature vector of the i-th training sample, $\boldsymbol{t}_i$ denotes the target value of $\boldsymbol{x}_i$, and L is the number of hidden neurons. The principle of ELM is devoted to reaching the smallest both training error and the norm of output weights, simultaneously, namely,

$$\text{Minimize}: \quad \|\boldsymbol{\beta}\|_{\mu}^{\sigma_1} + \lambda\|\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{T}\|_{v}^{\sigma_2} \tag{1}$$

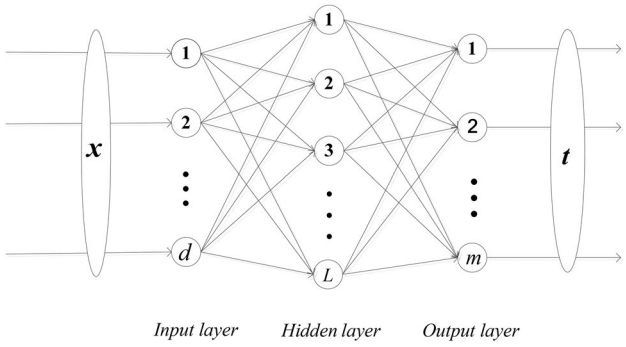

Fig. 1 The overview of ELM

where $\sigma_1 > 0, \sigma_2 > 0, u, v = 0, (1/2), \ldots, +\infty$, $\boldsymbol{H}$ represents the output matrix of the hidden layer, as shown below:

$$\boldsymbol{H} = \begin{bmatrix} h(\boldsymbol{x}_1) \\ \cdot \\ \cdot \\ \cdot \\ h(\boldsymbol{x}_N) \end{bmatrix} = \begin{bmatrix} h_1(\boldsymbol{x}_1) & \ldots & h_L(\boldsymbol{x}_1) \\ & \cdot \ \cdot \ \cdot & \\ & \cdot \ \cdot \ \cdot & \\ & \cdot \ \cdot \ \cdot & \\ h_1(\boldsymbol{x}_N) & \ldots & h_L(\boldsymbol{x}_N) \end{bmatrix} \sin^{-1}\theta \tag{2}$$

and the target matrix $\boldsymbol{T}$ of training data is defined as:

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{t}_1^T \\ \cdot \\ \cdot \\ \cdot \\ \boldsymbol{t}_N^T \end{bmatrix} = \begin{bmatrix} t_{11} & \ldots & t_{1m} \\ & \cdot \ \cdot \ \cdot & \\ & \cdot \ \cdot \ \cdot & \\ & \cdot \ \cdot \ \cdot & \\ t_{N1} & \ldots & t_{Nm} \end{bmatrix} \tag{3}$$

The ELM training process has the following steps.
(1) Set the hidden neuron parameters at random.
(2) Compute the output matrix $\boldsymbol{H}$ of the hidden layer.
(3) Attain the weight vector of output layer as below:

$$\boldsymbol{\beta} = \boldsymbol{H}^{\dagger}\boldsymbol{T} \tag{4}$$

where $\boldsymbol{H}^{\dagger}$ represents the Moore–Penrose (MP) generalized inverse of matrix $\boldsymbol{H}$.

The MP generalized inverse of matrix $\boldsymbol{H}$ can be computed by utilizing the vertical project approach, namely, $\boldsymbol{H}^{\dagger} = (\boldsymbol{H}^T\boldsymbol{H})^{-1}\boldsymbol{H}^T$. In accordance with the ridge regression theory [27, 28], a positive value $(1/\lambda)$ can be added to the computation of the weight vector $\boldsymbol{\beta}$. The solution is equivalent to the ELM optimal solution with $\sigma_1 = \sigma_2 = \mu = \nu = 2$, which possesses favorable generalization capability and high stabilization. There is:

$$\boldsymbol{\beta} = \boldsymbol{H}^T\left(\frac{1}{\lambda} + \boldsymbol{H}\boldsymbol{H}^T\right)^{-1}\boldsymbol{T} \tag{5}$$

and then the output of ELM will be computed as:

$$f(\boldsymbol{x}) = h(\boldsymbol{x})\boldsymbol{\beta} = h(\boldsymbol{x})\boldsymbol{H}^T\left(\frac{1}{\lambda} + \boldsymbol{H}\boldsymbol{H}^T\right)^{-1}\boldsymbol{T} \tag{6}$$

or there is:

$$\boldsymbol{\beta} = \left(\frac{1}{\lambda} + \boldsymbol{H}\boldsymbol{H}^T\right)^{-1}\boldsymbol{H}^T\boldsymbol{T} \tag{7}$$

and then the corresponding output of ELM will be calculated as:

$$f(\boldsymbol{x}) = h(\boldsymbol{x})\boldsymbol{\beta} = h(\boldsymbol{x})\left(\frac{1}{\lambda} + \boldsymbol{H}\boldsymbol{H}^T\right)^{-1}\boldsymbol{H}^T\boldsymbol{T} \tag{8}$$

2.2 Hierarchical Extreme Learning Machine (H-ELM)

It could be observed from Fig. 2 that, differing from traditional DL frameworks in [29] and [30], the H-ELM system [12] can be partitioned into two independent subframeworks. In the first subframework, a new ELM-based autoencoder is used to obtain multilayer sparse features of the input sample. In the latter subframework, the original ELM is implemented to make final decisions.

The principles and merits of H-ELM are described detailedly below. In order to mine latent knowledge within the training instances, the raw data entered is converted into an ELM random eigenspace. High-level sparse features will be acquired through an N-layers unsupervised learning. The output of each hidden layer can be expressed in mathematical formula as follows:

$$\boldsymbol{H}_i = g(\boldsymbol{H}_{i-1} \cdot \boldsymbol{\beta}) \tag{9}$$

where $\boldsymbol{H}_i$ and $\boldsymbol{H}_{i-1}$ are the output matrices of the i-th layer and (i-1)-th layer, respectively, $g(\cdot)$ represents the hidden layers activation function, and $\boldsymbol{\beta}$ denotes the output weight vector. It is noteworthy that, when the features of the former hidden layer are extracted, the parameters of the current hidden layer will be determined and do not need to be further adjusted. The more layers, the more compact the feature are generated. Therefore, each function can be regarded as a parted feature extractor, and each hidden layer of H-ELM can be identified as a self-contained module. However, within the classical DL models [29–33], all of their hidden layers are organized as an integral. And Back-Propagation (BP) algorithm is exploited to retrain the integral model iteratively. Consequently, compared to most of the classic DL frameworks, H-ELM possesses a faster learning speed.

As stated earlier, the second subframe of the entire H-ELM framework is implemented by the original ELM, thus we will focus on the first subframe. As is well-known, an autoencoder attempts to make the reconstructed outputs resemble the input data as far as possible, so as to effectively approximate input data [34]. Owing to its universal

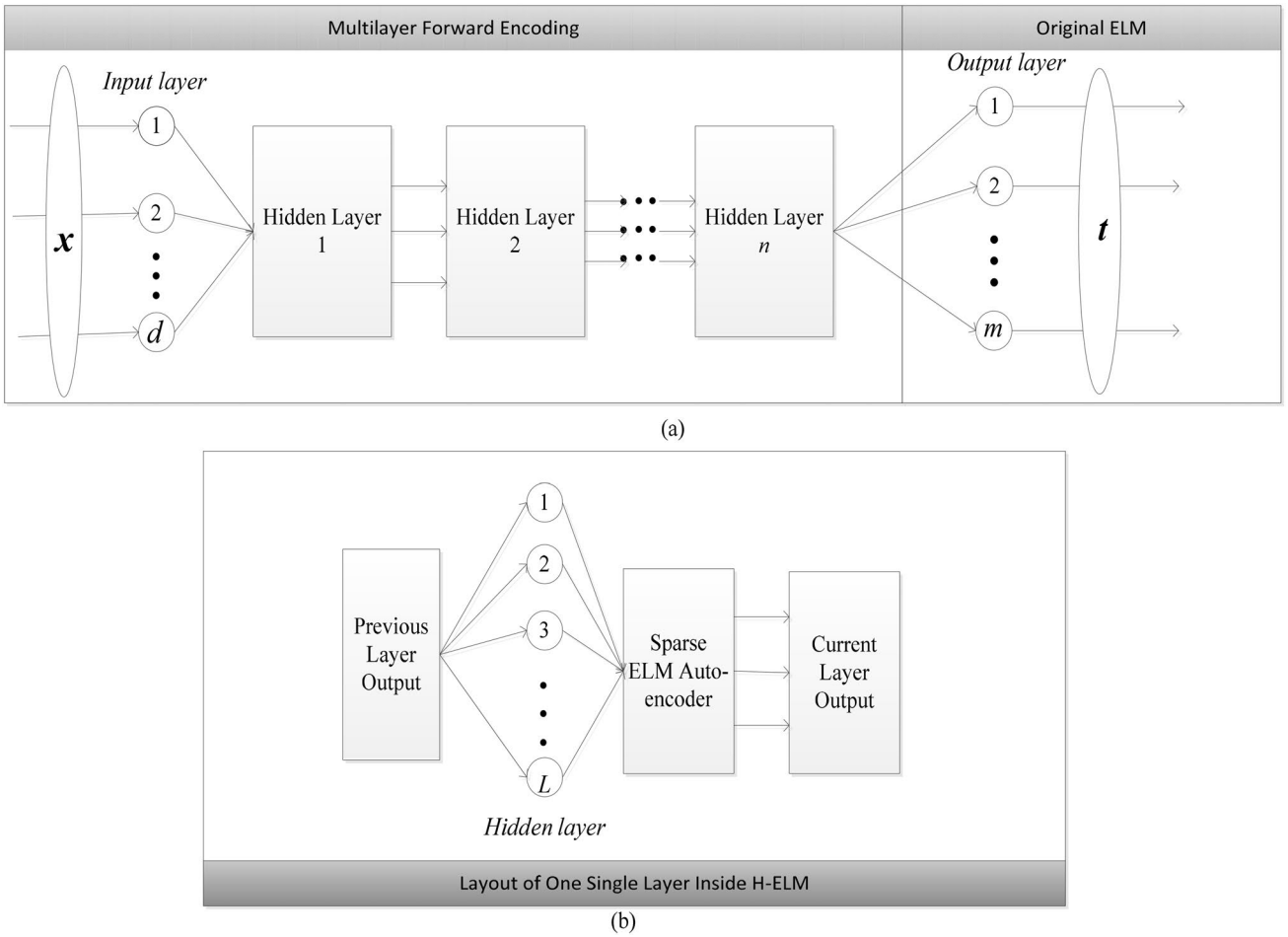

Fig. 2 The framework of H-ELM. **a** General frame of H-ELM, which is partitioned into two subframes: the frame of multilayer forward encoding and the frame of original ELM. **b** Layout of one single layer inside H-ELM

approximation ability, ELM is employed to develop the autoencoder. In the mean time, sparsity constraint is implemented on the optimization of autoencoder, forming the ELM sparse autoencoder. According to the principle of ELM, the optimization approach of this autoencoder can be formulated as Eq. (10).

$$O_\beta = \text{argmin}\left\{ \|\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{X}\|^2 + \|\boldsymbol{\beta}\|_{\ell_1} \right\} \tag{10}$$

where $\boldsymbol{X}$ represents the input samples, $\boldsymbol{H}$ denotes the output matrix of random mapping, which is not required to be optimized, and $\boldsymbol{\beta}$ is weight matrix of the hidden layer to be obtained.

Hereinafter, the ℓ_1 optimization algorithm is described. For more concise and clear expression, Eq. (10) is rewritten as:

$$O_\beta = p(\boldsymbol{\beta}) + q(\boldsymbol{\beta}) \tag{11}$$

where $p(\boldsymbol{\beta}) = \|\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{X}\|^2$, and $q(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_{\ell_1}$ is the ℓ_1 penalty term of the model.

A fast iterative shrinkage-thresholding algorithm (FISTA) is employed to tackle with the problem in Eq. (8). The implementation process of FISTA is listed out as below [35].

(1) Compute the Lipschitz constant γ of the gradient of smooth convex function ∇p.

(2) Start the iteration with $\boldsymbol{y}_1 = \boldsymbol{\beta}_0 \in \mathbf{R}^n$, $t_1 = 1$, initially. For the j-th iteration, the below holds.

(a) $\boldsymbol{\beta}_j = s_\gamma(\mathbf{y}_j)$, where s_γ is given by

$$s_\gamma = \arg\min_{\boldsymbol{\beta}} \left\{ \frac{\gamma}{2} \left\| \boldsymbol{\beta} - \left(\boldsymbol{\beta}_{j-1} - \frac{1}{\gamma} \nabla p(\boldsymbol{\beta}(j-1)) \right) \right\|^2 + q(\boldsymbol{\beta}) \right\}.$$

$$t_{j+1} = \frac{1 + \sqrt{1 + 4t_j^2}}{2} \tag{12}$$

(b)

(c) $y_{j+1} = \boldsymbol{\beta}_j + \left(\frac{t_j - 1}{t_{j+1}} \right)(\boldsymbol{\beta}_j - \boldsymbol{\beta}_{j-1})$.

By computing the iterative steps above, the data from the corrupted ones can be perfectly recovered. The ℓ_1 optimization has been proved to be a better solution for data recovery and other applications [36].

3 The measure of predictor competence for dynamic ensemble selection

The key of DES is how to measure the predictive power of a predictor for a given unseen sample. Since that the predictive complexities of different test samples dramatically differ from each other, it could be naturally thought of that, a sample will be predicted well if it is fed to the predictors which are good at forecasting it. In our previous work [22], three DES algorithms are proposed, i.e., DES-PALR, DES-CP and DVS-OpOp. They all design a measurement to estimate the predictive power of a base predictor to consider what information can obtain better predictive performance. However, in this paper, through considering multiple criteria, we present a new Dynamic Ensemble Pruning technique, i.e., the DEP-TSP^{meta} technique, and compare its forecasting performance with that of the previously proposed algorithms in Sect. 5. The three algorithms are described succinctly in the following subsections.

3.1 The DES algorithm based on Predictor Accuracy over the Local Region (DES-PALR)

Predictor precision is the most universal measurement for the implementation of DES. In DES-PALR, predictor precision is calculated based on a small area within the training dataset embracing the prescribed test sample. This area can be determined by implementing clustering algorithms, e.g., K-nearest-neighbor (KNN) algorithm [37]. Specifically, the local region would have a more similar distribute to the unseen instance than the other areas in the training set, so that the predictors which perform well on the local region are selected into the final ensemble system for the unseen sample.

The most important issue existing in DES-PALR is that, the predictive performance of this algorithm is closely related to the definition of local region. Besides, some abnormal samples around the unseen sample will have remarkable influence on the performance of this algorithm. Therefore, only using predictors that perform well on local regions is not enough to make the predicted values close to the true values. Consequently, in order to obtain better predictive performance, more information should be considered.

3.2 The DES algorithm based on the Consensus of Predictors (DES-CP)

Different from DES-PALR, DES-CP considers the extent of consensus of a pool of EoPs rather than a pool of predictors as a criterion. In this algorithm, a population of EoPs is first generated using genetic algorithm, i.e., Genetic Algorithm based Selective Ensemble (GASEN) [38]. The higher the extent of consensus of predictors, the better the predictive performance of the EoP is expected. Then, for each unseen sample, the EoP possessing the maximum consensus will act as the decision maker. Two variant algorithms based on this generated EoPs are proposed in [22]: DES-CP-Var and DES-CP-Clustering. The former assesses the consensus of EoP by calculating the variance of the predicted values of all predictors in each EoP. EoP with lower variance possesses higher consensus. The latter measures the extent of consensus of EoP by the difference between the scale of the cluster comprising the most and the second most predictive values. The bigger the difference is, the higher the consensus of EoP will be regarded.

The most difference between DES-PALR and DES-CP lies in that, DES-CP does not need to extract information from local regions. Therefore, the performance of this algorithm will not be influenced by the manner of local region definition. However, the computational costs would be increased for DES-CP, due to its requirement of generating a group of predictor ensembles.

3.3 The dynamic validation set determination algorithm based on the similarity between the Output profile of the test sample and the output profile of each training sample (DVS-OpOp)

DVS-OpOp is somewhat similar to DES-PALR, where the goal of both of them is to select samples which are close to the unseen sample to form the validation set. However, with DVS-OpOp, the similarity is calculated based on space of decisions rather than eigenspace. What's more, the similarity is measured by combining the output profile of the unseen sample and the output profiles of training set. The output profile of one sample is a vector that consists of the predicted values obtained by the base predictors for that sample.

The principle merit of this algorithm lies in that, it is not restricted by the definition of the local region in feature space. However, DVS-OpOp only considers the global knowledge of the unseen sample, while ignores its local information. Hence, we can simultaneously consider the local and the global knowledge and other features to measure competence of the base predictor in this work.

4 The proposed technique: DEP-TSP^{meta}

Aiming at addressing TSP problems, we specifically construct a dynamic ensemble pruning technique, i.e., DEP-TSP^{meta}, as shown in Fig. 3, which is partitioned into the following three stages. Moreover, to clearly describe our proposed technique, symbols frequently used in this paper is summarized in Table 1.

(a) Ensemble initialization stage, in which a preliminary collection of predictors using the training sample set T_λ is generated, denoted by $P = \{p_1, p_2, \ldots, p_M\}$.

(b) Meta-predictor construction stage, given each instance $(\boldsymbol{x}_j, y_j)$ from the training dataset T_ℓ and each predictor p_i, the meta-attribute vector $(\theta_{i,j}, \boldsymbol{m}_{i,j})$ is extracted and put into T^* that is later used to build several candidates and select the most competent one as the meta-predictor ℓ. A different training dataset is used at this stage to prevent overfitting.

(c) Prediction stage, for an unseen instance $(\boldsymbol{x}_{test\_j}, y_{test\_j})$, the meta-information $\boldsymbol{m}_{i,test\_j}$ is extracted based on dynamic pruning dataset D_{pru} is fed to the meta-predictor ℓ, which determines several excellent predictors to constitute the final dynamic ensemble P', so as to make the final predictive decision for this instance.

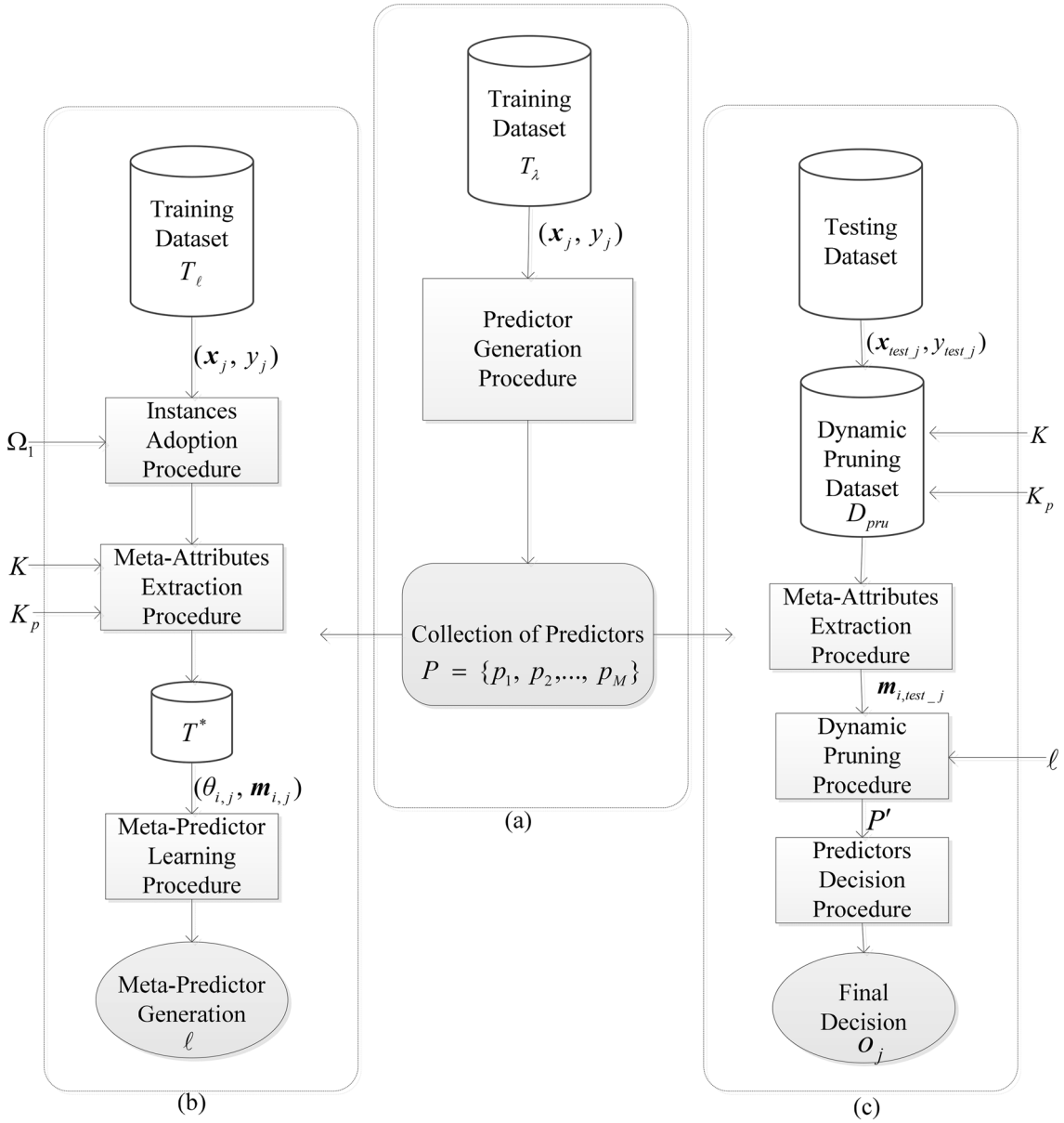

Fig. 3 Overview of the proposed DEP-TSP^{meta} technique

Table 1 Definitions of the major mathematical symbols used in our paper

Symbols	Descriptions
T_λ	Training dataset for ensemble initialization
T_ℓ	Training dataset for meta-predictor construction
D_{pru}	Dynamic pruning dataset
$(\boldsymbol{x}_j, y_j)$	The j-th training instance
$(\boldsymbol{x}_{test\_j}, y_{test\_j})$	The *test_j*-th testing instance
p_i	The i-th base predictor
P	Collection of predictors
$\boldsymbol{m}_{i,j}$	Meta-attribute vector for the j-th training instance extracted by the i-th predictor
$\boldsymbol{m}_{i,test\_j}$	Meta-attribute vector for the *test_j*-th testing instance extracted by the i-th predictor
$\theta_{i,j}$	The i-th predictor's competence for the j-th training instance
$\theta_{i,test\_j}$	The i-th predictor's competence for the *test_j*-th testing instance
T^*	Collection of meta-attributes
ℓ	Meta-predictor
P'	Final dynamic ensemble

4.1 Ensemble initialization

During the ensemble initialization stage, the goal is to produce a diversified and informative collection of predictors, as it makes no sense to integrate predictors that always render duplicate outputs. The method of integrating varying predictor types and different learning parameters is utilized in this stage. Specifically, two types of base learners, i.e., ELMs and H-ELMs, with different numbers of hidden neurons layers, disparate hidden neurons quantities, and diversified activation functions are employed to generate the collection of base predictors, using the training samples from the dataset T_λ. The detailed settings of these generated base predictors $P = \{p_1, p_2, \ldots, p_M\}$ are shown in Table 3 of Sect. 5.2.

4.2 Meta-predictor construction

As displayed in Fig. 3, this stage mainly includes three procedures: the instances adoption procedure, the meta-attributes extraction procedure, and the meta-predictor learning procedure. The crucial parameters existing in the meta-predictor training procedure are specifically introduced in the dynamic parameters adjustment procedure at last. The details of each procedure are described as follows.

4.2.1 Instances adoption

In order to solve the problem of low degree of consensus, we determine to focus the training of meta-predictor ℓ to specifically handle the case where the extent of consensus among the collection of predictors is low. Each instance $(\boldsymbol{x}_j, y_j)$ is first estimated by the whole ensemble of predictors to obtain $P(\boldsymbol{x}_j) = (p_1(\boldsymbol{x}_j), p_2(\boldsymbol{x}_j), \ldots, p_M(\boldsymbol{x}_j))$, which denotes the predicted values of all predictors in the ensemble P. Then, the consensus of the ensemble P is evaluated by calculating the variance of the predicted values made by its constituent predictors. The prediction variance of instance $(\boldsymbol{x}_j, y_j)$ among the collection of predictors is calculated as below:

$$var(P(\boldsymbol{x}_j)) = \frac{\sum_{i=1}^{M}\left(p_i(\boldsymbol{x}_j) - \frac{\sum_{a=1}^{M} p_a(\boldsymbol{x}_j)}{M}\right)}{M} \tag{13}$$

The smaller the variance is, the higher the consensus will be. Thus, the degree of consensus is computed as:

$$con((\boldsymbol{x}_j, y_j), P) = \frac{1}{var(P(\boldsymbol{x}_j))} \tag{14}$$

To judge whether the extent of consensus is low, a minimum acceptable consensus needs to be defined, i.e., the consensus threshold Ω_1. If the consensus $con((\boldsymbol{x}_j, y_j), P)$ falls below the threshold Ω_1, the instance $(\boldsymbol{x}_j, y_j)$ will be selected to extract meta-attributes for training meta-predictor ℓ.

Before meta-attributes are extracted, the local area of the instance $(\boldsymbol{x}_j, y_j)$ is calculated by using the KNN algorithm, which is composed of its K most similar instances, and is represented by $L_{(\boldsymbol{x}_j, y_j)} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_K, y_K)\}$. Next, the instance $(\boldsymbol{x}_j, y_j)$ and all the instances in the training set T_ℓ are transformed into their corresponding output property files. The output property file of the instance $(\boldsymbol{x}_j, y_j)$ is denoted as $\tilde{y}_j = (\tilde{y}_{j,1}, \tilde{y}_{j,2}, \ldots, \tilde{y}_{j,M})$, where $\tilde{y}_{j,i}$ is the predictive result generated by the base predictor p_i for the instance $(\boldsymbol{x}_j, y_j)$. Furthermore, the similarity is computed between the output property file of the instance $(\boldsymbol{x}_j, y_j)$ and the output property files of all the instances in training set T_ℓ by utilizing the KNN algorithm. The outcome of this procedure will guide us to select instances, that are most similar to $(\boldsymbol{x}_j, y_j)$, to constitute $G_{(\boldsymbol{x}_j, y_j)} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_{Kp}, y_{Kp})\}$.

Finally, by employing each p_i in the collection of predictors P, together with the instance $(\boldsymbol{x}_j,y_j)$, the local area $L_{(\boldsymbol{x}_j,y_j)}$ and the global area $G_{(\boldsymbol{x}_j,y_j)}$, one meta-attribute vector $\boldsymbol{m}_{i,j}$ can be extracted.

4.2.2 Meta-attributes extraction

In one of our previous research works [22], we use one single criterion to estimate the capability of a predictor. While in this work, we take into consideration the data characteristics of TSP problems, further proposing four different sets of meta-attributes. Each attribute set, i.e., $\boldsymbol{f}_1$, f_2, $\boldsymbol{f}_3$ and f_4, reflects a characteristic about the behavior of a base predictor, and can be regarded as a criterion, such as the prediction performance estimated in the local area, and the predictor confidence for the prediction of an unseen instance. Utilizing four different sets of meta-attributes, even if one criterion does not work owing to the inaccuracy in the local areas or the results with low confidence, the system can still accomplish favorable predictive performance, because other criteria are taken into account in the algorithm implementation.

The two attribute sets $\boldsymbol{f}_1$ and f_2 are calculated employing the information drawn from the local area of capacity $L_{(\boldsymbol{x}_j,y_j)}$. The attribute set $\boldsymbol{f}_3$ uses information obtained from the global area of capacity $G_{(\boldsymbol{x}_j,y_j)}$. And f_4 is computed directly from the instance $(\boldsymbol{x}_j,y_j)$, which shows the level of confidence of p_i for the accurate prediction for $(\boldsymbol{x}_j,y_j)$. The details of the four sets of meta- attributes are described in the following.

The criterion of the neighbor's hard prediction is denoted by $\boldsymbol{f}_1$. Firstly, a vector with K elements is set up, where K denotes the size of the local area. For each element $(\boldsymbol{x}_\sigma^{c_1},y_\sigma^{c_1})$, $\sigma \in [1, K]$, belonging to the local area of the input instance $(\boldsymbol{x}_j,y_j)$, where c_1 represents the first criterion, if the Root Mean Square Error (RMSE) of the prediction made by p_i is less than the preset threshold, the σ-th element of the vector is assigned to 1, otherwise it is assigned to 0.

$$D_\sigma^{c_1} = \begin{cases} 1, & RMSE(p_i,(\boldsymbol{x}_\sigma^{c_1},y_\sigma^{c_1})) < threshold \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

$$\boldsymbol{f}_1 = (D_1^{c_1},\mathrm{D}_2^{c_1},\ldots,D_K^{c_1}) \tag{16}$$

where $\boldsymbol{f}_1$ is a vector of $D_\sigma^{c_1}$, $\sigma \in [1, \mathrm{K}]$, and $D_\sigma^{c_1}$ represents the performance of p_i when implemented on $(\boldsymbol{x}_\sigma^{c_1},y_\sigma^{c_1})$.

The criterion of the overall local accuracy is denoted by f_2. The *RMSE* of the prediction made by p_i on the entire area of capacity $L_{(\boldsymbol{x}_j,y_j)}$ is calculated, denoted by $RMSE(p_i,L_{(\boldsymbol{x}_j,y_j)})$ and the reciprocal of $RMSE(p_i,L_{(\boldsymbol{x}_j,y_j)})$ is encoded as f_2.

$$f_2 = \frac{1}{RMSE(p_i,L_{(\boldsymbol{x}_j,y_j)})} \tag{17}$$

The criterion of global area accuracy is denoted by $\boldsymbol{f}_3$. Similar to $\boldsymbol{f}_1$, first, a vector with Kp elements is set up, where Kp denotes the size of the global area. Then, for each element $(\boldsymbol{x}_\sigma^{c_3},y_\sigma^{c_3})$ in $G_{(\boldsymbol{x}_j,y_j)}$, $\sigma \in [1, Kp]$, where c_3 represents the third criterion, belonging to the global area decided by output property file, if difference between the *RMSE* obtained by p_i on $(\boldsymbol{x}_\sigma^{c_3},y_\sigma^{c_3})$ and the *RMSE* obtained by p_i on $(\boldsymbol{x}_j,y_j)$ is less than the threshold, the σ-th element of the vector is assigned to 1, otherwise it is assigned to 0.

$$D_\sigma^{c_3} = \begin{cases} 1, & \left|RMSE(p_i,(\boldsymbol{x}_j,y_j)) - RMSE(p_i,(\boldsymbol{x}_\sigma^{c_3},y_\sigma^{c_3}))\right| < threshold \\ 0, & \text{otherwise} \end{cases} \tag{18}$$

$$\boldsymbol{f}_3 = (D_1^{c_3},D_2^{c_3},\ldots,D_{Kp}^{c_3}) \tag{19}$$

where f_3 is a vector of $D_\sigma^{c_3}$, $\sigma \in [1, Kp]$, and $D_\sigma^{c_3}$ represents the consensus of the decisions of p_i when implemented on $(\boldsymbol{x}_\sigma^{c_3},y_\sigma^{c_3})$ and $(\boldsymbol{x}_j,y_j)$.

The criterion of predictor's confidence is denoted by f_4. The predicted value of p_i on $(\boldsymbol{x}_j,y_j)$ is added to the set of predicted values, with the scale of the set being equal to the scale of the initial collection of predictors. Then, all the deviations between the predicted values and the true value are calculated. Finally, the min–max normalization approach is utilized to normalize all the deviations to the interval [0, 1]. Each of deviation value d_i is normalized by the min–max normalization formula, which is shown as below:

$$d_i^{new} = \frac{d_i - d^{min}}{d^{max} - d^{min}} \tag{20}$$

where d_i^{new} is the normalized value, $d_i^{\min}$ and $d_i^{\max}$ represent the minimum and maximum value of all deviations, respectively.

$$f_4 = \frac{1}{d_i^{new}} \tag{21}$$

A vector $\boldsymbol{m}_{i,j} = (\boldsymbol{f}_1,f_2,\boldsymbol{f}_3,f_4)$ can be constructed at the end of meta-attributes extraction procedure (Fig. 4), where $\boldsymbol{m}_{i,j}$ represents the meta-knowledge that extracted by p_i from $(\boldsymbol{x}_j,y_j)$, with the size of $\boldsymbol{m}_{i,j}$ being $K + Kp + 2$. If the psredicted value of p_i on $(\boldsymbol{x}_j,y_j)$ is very close to the true value, the class label of $\boldsymbol{m}_{i,j}$, i.e., $\theta_{i,j}$, is set to 1; otherwise, it is set to 0.This class label indicates whether p_i possesses good enough performance on $(\boldsymbol{x}_j,y_j)$ or not. $(\theta_{i,j},\boldsymbol{m}_{i,j})$ is stored into the meta-attributes dataset T^*.

Each predictor in the collection of predictors can extract a meta-attribute from each instance. There are N instances belonging to T_ℓ, whose consensus $E\left((\boldsymbol{x}_j,y_j),P\right)$ is less than Ω_1. In this way, the size of the meta-attributes dataset T^* equals $M \times N$ (M is the size of the collection of predictors). Hence, we can address the small sample size prediction

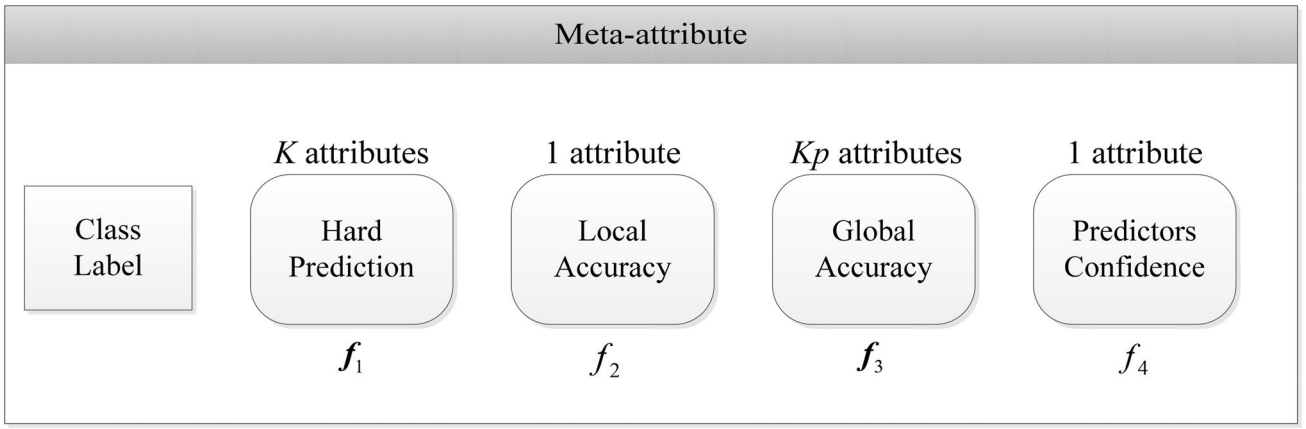

Fig. 4 Attribute vector contains the meta-knowledge about the behavior of the base predictor. The class label indicates whether p_i possesses good enough performance on $(\mathbf{x}_j, y_j)$ or not

issue, due to the lack of training data, by the increasement to the scale of the collection of predictors.

4.2.3 Learning of meta-predictor

The purpose of this procedure is to train the meta-predictor ℓ. In this work, we use 75 percent of the dataset T^* for learning and the remaining 25 percent for validation. ELM is employed as the base model for the meta-predictor. Ten ELMs are trained, using {10, 20, …, 100} as the set of the numbers of hidden neurons, and the sigmoid transfer function as the activation function. The criterion utilized to evaluate the performance of the ten meta-predictors is RMSE. That is to say, the meta-predictor achieving the minimum RMSE over the validation set is chosen to be the decisive meta-predictor ℓ.

The whole meta-predictor construction stage is formalized in Algorithm 1.

4.2.4 The dynamic adjustment of parameters

There exist three crucial parameters for the successful implementation of the proposed technique: the consensus threshold Ω_1, the local area size K, and the global area size Kp. For different datasets, the most appropriate parameters should be dynamically adapted to apply to the DEP-TSP$^{\text{meta}}$ technique. Here, genetic algorithm (GA) [39–41] is applied to find the optimal parameters for each dataset.

GA is a searching algorithm designed on basis of biological evolutionary principle. It is a mathematical model for simulating Darwin's genetic selection and natural elimination. As an effective global parallel optimization search tool, GA is simple, universal, robust and fit for concurrent processing.

In our paper, the three parameters are optimized by GA, including the following several basic steps:

Step 1: The three parameters are encoded into binary format and an initialized population of size 20 is generated. And the maximal generation epoch of GA is set to 20.

Step 2: The initialized population of size 20 is used in the proposed technique for evaluating the performance of DEP-TSP$^{\text{meta}}$. RMSE is employed as the measurement for the fitness value.

Step 3: The "roulette" is set to be the selection function. Roulette is the conventional selection function with the survival probability equal to the fitness of i/sum of all the individuals.

Step 4: The "simpleXover" is set to be the crossover function. According to the results obtained by using trial and error method, the probability of crossover operators is assigned to be 0.6. According to the crossover probability, the individuals in the population are randomly matched and the parameters of different positions are changed.

Step 5: The "binaryMutation" is set to be the mutation function. The probability of mutation operators is set to be 0.005. The "binaryMutation" function varies each bit of the parent on basis of the mutation probability.

According to the literature [42], it is proved by means of homogeneous finite Markov chain analysis that GA does not converge to the global optimum. However, as the number of iterations increases, GA can guarantee to find an approximate optimal solution. Moreover, a more practical question regards the time complexity of the algorithm to achieve the optimal solution. Thus, by using the trial-and-error method, a sufficient and acceptable initialized population size and generation epoch are set to ensure that the appropriate values of the three parameters are acquired by searching. Specifically, when the maximal generation epoch is reached, the individual with the maximum fitness function value is the output of the final solution and the algorithm is terminated.

Algorithm 1. The meta-predictor construction stage

Input:

Training dataset: T_ℓ;

The collection of predictors: $P = \{p_1, p_2, ..., p_M\}$;

Output:

The meta-predictor: ℓ.

For each instance $(\boldsymbol{x}_j, y_j) \in T_\ell$ do

  Compute the consensus of P according to Eqs. (13)-(14), denoted by $con((\boldsymbol{x}_j, y_j), P)$.

  If $con((\boldsymbol{x}_j, y_j), P) < \Omega_1$

    Find the most similar K instances to $(\boldsymbol{x}_j, y_j)$ using the KNN algorithm to constitute the local area, denoted by $L_{(\boldsymbol{x}_j, y_j)}$.

    Compute the output property files of $(\boldsymbol{x}_j, y_j)$ and all instances in T_ℓ.

    Find the most similar Kp instances to $(\boldsymbol{x}_j, y_j)$ based on the output property files using the KNN algorithm to constitute the global area, denoted by $G_{(\boldsymbol{x}_j, y_j)}$.

    For each predictor $p_i \in P$ do

      $\boldsymbol{m}_{i,j} = \text{meta} - \text{attributes} - \text{extraction}\left(L_{(\boldsymbol{x}_j, y_j)}, G_{(\boldsymbol{x}_j, y_j)}, p_i, (\boldsymbol{x}_j, y_j)\right)$ according to Eqs. (15)-(21).

      Compute the $RMSE_{i,j}$ between $p_i(\boldsymbol{x}_j)$ and y_j.

      If $RMSE_{i,j} < \Omega_2$

        $\theta_{i,j} = 1$ denotes that p_i is capable for $(\boldsymbol{x}_j, y_j)$.

      Else

        $\theta_{i,j} = 0$ denotes that p_i is incapable for $(\boldsymbol{x}_j, y_j)$.

      End if

      Put $(\theta_{i,j}, \boldsymbol{m}_{i,j})$ into the collection of meta-attributes T^*.

    End for

  End if

End for

Partition T^* into 75% for training and 25% for validation.

Build some ELMs based on the 75% training subset of T^*, and choose the optimal one, on basis of the 25% validation subset of T^*, as the meta-predictor ℓ.

Return ℓ.

Just as the "No Free Lunch" theorem declares [43], there exists no algorithm that is superior to any other ones among all the probable problems. Therefore, it is necessary that GA is utilized to adjust the three crucial parameters dynamically, i.e., Ω_1, K, Kp. It is found through our experiments that, dynamic adjustments to the values of the crucial parameters significantly boost the predictive performance of the proposed technique.

4.3 Prediction

The prediction stage is described in Algorithm 2. Given the test instance $(\boldsymbol{x}_{test\_j},y_{test\_j})$, in this stage, the local area of capacity $L_{(\boldsymbol{x}_{test\_j},y_{test\_j})}$ having K most similar instances is defined with the instances from the dynamic pruning dataset D_{pru}. Next, the output property files of all the instances in D_{pru} and $(\boldsymbol{x}_{test\_j},y_{test\_j})$ are computed, and the most Kp similasr instances are determined as the global area of capacity $G_{(\boldsymbol{x}_{test\_j},y_{test\_j})}$, accosrding to the similarity between the output property files of all the instances in the dynamic pruning dataset and the output property file of the test instance.

Then, for each predictor p_i in the initial collection of predictors P, the meta-attributes extraction procedure is the same as that described in Sect. 4.2.2, and the vector $\boldsymbol{m}_{i,test\_j}$ is extracted. Next, $\boldsymbol{m}_{i,test\_j}$ will be fed into the meta-predictor ℓ. If the output $\theta_{i,test\_j}$ of ℓ equals 0 (i.e., p_i is incapable for the test instance), p_i will be pruned; otherwise if the output $\theta_{i,test\_j}$ of ℓ equals 1 (i.e., p_i is capable for the test instance), p_i will be added into the ensemble P'. When each predictor in the collection of predictors is estimated, the final ensemble P' is obtained. The averaged predicted values made by the predictors in the ensemble P' is taken as the final decision for $(\boldsymbol{x}_{test\_j},y_{test\_j})$.

Detailed pseudocode of DEP-TSP$^{\text{meta}}$ is described in the following Algorithm 2.

Algorithm 2. The proposed DEP-TSP$^{\text{meta}}$ algorithm

Input:

The training dataset: $T_\lambda \bigcup T_\ell$;

The dynamic pruning dataset: D_{pru};

The test instance: $(\boldsymbol{x}_{test\_j}, y_{test\_j})$;

Output:

The final decision: o_{test_j}.

1: Generate the collection of predictors $P=\{p_1,p_2,...,p_M\}$ using T_λ.

2: Construct the meta-predictor ℓ using T_ℓ.

3: Find the most similar K instances to $(\boldsymbol{x}_{test\_j}, y_{test\_j})$ in D_{pru} using the KNN algorithm to constitute the local area, denoted by $L_{(\boldsymbol{x}_{test\_j}, y_{test\_j})}$.

4: Compute the output property files of $(\boldsymbol{x}_{test\_j}, y_{test\_j})$ and all instances in D_{pru}.

5: Find the most similar Kp instances to $(\boldsymbol{x}_{test\_j}, y_{test\_j})$ based on the output property files using the KNN algorithm to constitute the global area, denoted by $G_{(\boldsymbol{x}_{test\_j}, y_{test\_j})}$.

6: Initialize dynamic ensemble of predictors as $P'=\{\}$.

7: **For** each predictor $p_i \in P$ do

8: $\boldsymbol{m}_{i,test\_j} = meta-attributes-extraction\left(L_{(\boldsymbol{x}_{test\_j}, y_{test\_j})}, G_{(\boldsymbol{x}_{test\_j}, y_{test\_j})}, p_i, (\boldsymbol{x}_{test\_j}, y_{test\_j})\right)$ according to Eqs. (15)-(21).

9: Input $\boldsymbol{m}_{i,test\_j}$ into the meta-predictor ℓ and output $\theta_{i,test\_j}$.

10: **If** $\theta_{i,test\_j}=1$

11: $P'=P'\bigcup p_i$.

12: **End if**

13: **End for**

14: $o_{test\_j}=\frac{1}{\|P'\|}\sum_{t=1}^{\|P'\|} p_{a_t}(\boldsymbol{x}_{test\_j})$, $p_{a_t}\in P'$.

15: **Return** o_{test_j}.

4.4 A summary of the proposed DEP-TSP^{meta} technique

The contents of this section could be divided into three parts: (1) reorganizing and summarizing the core idea of the DEP-TSP^{meta} technique; (2) discussing advantages and disadvantages of the proposed technique; (3) analyzing the computational complexity of the proposed technique and making comparison to previous algorithms.

As stated in the "No Free Lunch" (NFL) theorem, without a prior assumption about specific problems, no algorithm could be expected to perform more superior than any other. There does not exist a universal optimal algorithm [44]. However, we believe that using one single criterion to estimate an algorithm performance is biased. It would be comprehensive and reasonable to consider more measurement criteria. In this work, a multiple criteria Dynamic Ensemble Pruning technique exploiting *meta*-learning specialized for TSP, i.e., the DEP-TSP^{meta} technique, is presented. The meta-attributes employed by *meta*-learning are the distinct criteria applied to evaluate the competence of base predictors from different angles.

In our proposed DEP-TSP^{meta} technique, both shallow models, i.e., ELMs, and deep models, i.e., H-ELMs, with varying predictor types and different learning parameters are combined together to constitute the initial ensemble. Four meta-attributes representing different criteria for evaluating base predictors capacities are developed, including the prediction performance of a predictor in a local area, the prediction confidence of a predictor on every instance in a local area, the predictive performance of a predictor in the global area, and the prediction confidence of a predictor on the current test instance. The meta-attributes extracted from the training dataset are utilized to build one meta-predictor, and this meta-predictor will be responsible for selecting the most appropriate base predictors to construct the final ensemble for making the predictive decision. The crucial parameters of the meta-attributes are adjusted by genetic algorithm for matching the current dataset dynamically.

In summary, DEP-TSP^{meta} possesses the following three advantages:

- It is developed based upon four different meta-attributes, therefore, it is robust even though some base predictors do not perform well on one or two of the meta-attributes.
- The ensemble within DEP-TSP^{meta} is constructed employing ELM and H-ELM as its base learners. Consequently, the technique not only inherits the merit of the former in its fast running speed, but also inherits the advantage of the latter in its adequate extraction of data features.
- The dynamic ensemble pruning paradigm employed within DEP-TSP^{meta} can improve its generalization performance, while reducing its scale, simultaneously.

In contrast with the above summarized advantages, DEP-TSP^{meta} possesses two obvious disadvantages:

- Genetic algorithm employed within DEP-TSP^{meta} results in its relatively high computational complexity;
- The structure of DEP-TSP^{meta} is relatively complex.

The computational complexities of the DEP-TSP^{meta} training and testing phases are, respectively, $O\left(N\left(M + M\left(N\log K + 2M + N\log K_p\right)\right)gM^2(1.5r + rf)\right)$ and $O\left(M + M\left(N\log K + 2M + N\log K_p\right)\right)$, where N represents the scale of training dataset, M denotes the number of predictors in the initial ensemble collection, K and K_p represent the scales of global area and local area, respectively, g denotes the amount of generations, r denotes the scale of population, and f represents the chromosome length. Because the proposed DEP-TSP^{meta} technique is

Table 2 Crucial information of the eight experimental datasets

Dataset	Metrics	Time granularity	Time range	Description	Domain	Repository
IAP	144 fact values	Month	Jan 1949–Dec 1960	International airline passengers	Transport	TSDL
QIS	127 fact values	Quarter	1959 Q4–1991 Q2	Quarterly increase in stocks non-farm total of Australia	Finance	TSDL
DHA	91 fact values	Year	1919–2004	Deaths from homicides in Australia	Crime	TSDL
DSA	91 fact values	Year	1919–2004	Deaths from suicides in Australia	Crime	TSDL
ITD	69 fact values	Date	19 Nov 2004–26 Jan 2005	Internet traffic data in bits	Computing	TSDL
DJI	126 fact values	Date	1 Jan 2020–30 Jun 2020	Dow Jones Industrial Average	Finance	Yahoo
Odonovan	70 fact values	Time	1983	Consecutive yields for batch chemical process	Chemistry	TSDL
Montgome	100 fact values	Time	1976	Chemical process readings every two minutes	Chemistry	TSDL

Table 3 The exhaustive information of base models used in experiments

ID	Basis model	The number of hidden layers	The number of hidden neurons	Activation function
1–10	ELM	1	11–20	Sigmoid (22)
11–20	ELM	1	11–20	Sine (23)
21–30	ELM	1	11–20	Hard limit transfer (24)
31–40	ELM	1	11–20	Triangular basis transfer (25)
41–50	ELM	1	11–20	Radial basis transfer (26)
51–100	H-ELM	3	51–100	Tangent sigmoid (27)

Table 4 Genetic algorithm parameters

Population size	20
Number of generations	20
Probability of crossover	0.6
Probability of mutation	0.005

constructed based upon four meta-attributes extracted from DES-PALR, DES-CP, and DES-OpOp, the computational complexities of the proposed technique in training and testing phases become larger compared to previous algorithms. However, this construction scheme brings higher prediction accuracy and stronger robustness to the DEP-TSP^{meta} technique, accordingly.

5 Experiments

5.1 Datasets

We carry out empirical study based upon eight one-dimensional time series datasets selected from Time Series Data Library (TSDL) [45] and Yahoo Finance [46]. The key information of the eight benchmark datasets is presented, respectively, in Table 2. The eight datasets are drawn from diverse real-world domains, with different numbers of data points, different time granularities, different time ranges, different domains, and different types of data, which guarantee the diversity of experimental samples.

It is required to regulate the domain value of each attribute into the interval [0, 1], due to the different scopes of dataset attributes. This operation of normalization guarantees that the data attributes with greater values do not overwhelm the smaller ones, so that predictive performance could be enhanced. Each value in the whole dataset is normalized by min–max normalization.

5.2 Experimental setup

MAE [47] and RMSE [48] are employed as the evaluation measurements of prediction errors in this work, with the definitions being presented as below:

Table 5 The parameters values determined by genetic algorithm

Dataset	Local area size	Global area size	Consensus threshold
IAP	5	5	0.2
QIS	6	9	0.6
DHA	3	10	0.6
DSA	7	8	0.6
ITD	7	5	0.6
DJI	5	8	0.7
Odonovan	6	5	0.6
Montgome	7	4	0.3

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|y_i - \tilde{y}_i\right| \tag{22}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \tilde{y}_i)^2} \tag{23}$$

here y_i is the true value, $\tilde{y}_i$ denotes the predicted value, and N is the number of samples.

Experiments are carried out with the system configuration as follows: 2.6 GHz PC with 1 GB of RAM using Windows XP operating system, MATLAB language source code.

Then, as base models, 50 well-trained ELMs and 50 well-trained H-ELMs constitute the initial collection of predictors. And the settings of these base models are shown in the following Table 3.

The definitions of the six activation functions are shown as below:

$$f(x) = \frac{1}{1+e^{-x}} \tag{24}$$

$$f(x) = sin(x) \tag{25}$$

$$f(x) = \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases} \tag{26}$$

Table 6 The detailed RMSE performance of corresponding algorithms on the eight benchmark time series datasets

RMSE	Dataset							
Model	IAP	QIS	DHA	DSA	ITD	DJI	Odonovan	Montgome
DEP-TSP^{meta}	0.1036	**0.2360**	0.2228	0.1842	**0.0037**	**0.0204**	**0.2088**	0.2100
DES-PALR	**0.1017**	0.2727	0.2038	0.2685	0.0951	0.0218	0.2527	0.2160
DES-CP-Clustering	0.1589	0.2367	0.1888	0.1922	0.1075	0.0209	0.2642	**0.2025**
DVS-OpOp	0.1257	0.7154	0.2055	0.2096	0.0933	0.0222	0.2647	0.2150
GASEN	0.1066	0.2761	**0.1845**	**0.1484**	0.0963	0.1307	0.2787	0.2251
AA	0.4147	1.2235	0.3642	2.3006	0.1153	0.0530	0.3793	0.2378
BS-ELM	0.2518	0.2552	0.2163	0.1970	0.1432	0.0761	0.3410	0.2446
BS-H-ELM	1.3512	1.6098	0.7640	2.9352	0.1801	0.1050	0.4266	0.2955

The boldface indicates the best RMSE performance obtained by the corresponding algorithm on each time series dataset

Table 7 The detailed MAE performance of corresponding algorithms on the eight benchmark time series datasets

MAE	Dataset							
Model	IAP	QIS	DHA	DSA	ITD	DJI	Odonovan	Montgome
DEP-TSP^{meta}	**0.0681**	**0.1832**	**0.1568**	0.1464	**0.0028**	**0.0163**	**0.1665**	0.1608
DES-PALR	0.0797	0.2144	0.1725	0.1897	0.0754	0.0188	0.1968	0.1780
DES-CP-Clustering	0.1267	0.1988	0.1581	0.1660	0.0781	0.0167	0.2239	**0.1600**
DVS-OpOp	0.0938	0.3609	0.1674	0.1610	0.0726	0.0192	0.2091	0.1702
GASEN	0.0763	0.2354	0.1611	**0.1296**	0.0756	0.1127	0.2301	0.1796
AA	0.2672	0.7727	0.2810	0.9642	0.0862	0.0450	0.2967	0.1824
BS-ELM	0.2078	0.2189	0.1796	0.1629	0.1158	0.0644	0.2771	0.1907
BS-H-ELM	0.6650	0.9012	0.5770	1.2451	0.1431	0.0920	0.3404	0.2251

The boldface indicates the best MAE performance obtained by the corresponding algorithm on each time series dataset

$$f(x) = \begin{cases} 1 - |x|, & \text{if } -1 \le x \le 1 \\ 0, & \text{otherwise} \end{cases} \tag{27}$$

$$f(x) = e^{-x^2} \tag{28}$$

$$f(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{29}$$

Each experiment is repeated for 10 times, and all the reported experimental results are the average of these 10 repetitive runs. By using the trial-and-error method, and taking into account the dataset scale, simultaneously, the time window size (TWS) is set to five. For each repetition, the datasets are divided into: 50% for the training dataset, 25% for the dynamic pruning dataset, and 25% for the test dataset. For the proposed DEP-TSP^{meta} technique, 50% of the training dataset is used to generate the initial collection of predictors, and the other 50% is used for meta-predictor construction. According to the literature [49], this procedure used in our experiments is actually one kind of cross validation procedures, i.e., hold-out cross validation, which is a technique that relies on a single split of data. The data is divided into two non-overlapping parts and these two parts are used for training and testing respectively. Therefore, in our experiment, the samples that test the performance of the proposed algorithm do not appear in the training dataset, there will not yield an overoptimistic result.

For each dataset, the specific values of GA parameters are listed out in Table 4.

GA determines the most appropriate values of the three parameters for each dataset, as shown in Table 5.

We compare the predictive accuracy of the proposed DEP-TSP^{meta} technique, against seven current techniques. The seven comparative algorithms used in this study are: DES-PALR, DES-CP-Clustering, DVS-OpOp, Genetic Algorithm based on Selective Ensemble (GASEN), Averaging All (AA), Best Single ELM (BS-ELM), and Best Single H-ELM (BS-H-ELM), respectively. The first three algorithms belong to the category of DES techniques, while the others are part of the static ensemble selection techniques.

The objective of the comparative study is mainly to test and verify three research questions: (1) Does the ensemble of predictors outperform the best single model? (2) Does the DEP paradigm outperform the state-of-the-art static ensemble selection one, especially GASEN? (3) Whether the

implementation of multiple DEP criteria as meta-attributes leads to a more robust performance or not, even when it is confronted with ill-defined problems?

5.3 Experimental results

We spilt the experimental results into two tables: Table 6 shows the detailed RMSE performance on the eight benchmark time series datasets compared with the seven techniques, including three DES algorithms and four static ensemble selection rules. And Table 7 shows the corresponding comparisons based on the detailed MAE performance.

It is clearly shown from the results reported in Table 6 that, the proposed technique has the best RMSE performance on the QIS, ITD, DJI, and Odonovan datasets, and obtains the second best RMSE performance on the IAP, DSA, and Montgome datasets. That is to say, for 7 out of the 8 RMSE results, the proposed algorithm performs significantly better

Table 8 T-test results on RMSE between DEP-TSP^{meta} and other comparative algorithms on the eight benchmark time series datasets

Model A	Model B	RMSE							
		IAP	QIS	DHA	DSA	ITD	DJI	Odonovan	Montgome
The Proposed Algorithm	DES-PALR	0.5300	0.1602	0.9993	**0.0220**	**1.6552e−09**	0.2108	**0.0457**	0.2719
		H = 0	H = 0	H = 0	**H = 1**	**H = 1**	H = 0	**H = 1**	H = 0
	DES-CP-Clustering	**0.0055**	0.2590	1.000	0.6252	**1.1315e−10**	0.2964	**0.0209**	0.1650
		H = 1	H = 0	H = 0	H = 0	**H = 1**	H = 0	**H = 1**	H = 0
	DVS-OpOp	0.9399	0.1227	0.9975	0.1744	**7.4196e−13**	**0.0414**	**0.0282**	0.3167
		H = 0	H = 0	H = 0	H = 0	**H = 1**	**H = 1**	**H = 1**	H = 0
	GASEN	0.0660	**0.0023**	0.5673	0.1259	**5.5684e−09**	**2.4249e−6**	**0.0202**	0.0831
		H = 0	**H = 1**	H = 0	H = 0	**H = 1**	**H = 1**	**H = 1**	H = 0
	AA	0.6255	**2.7114e−08**	**0.0028**	**5.9136e−12**	**9.7159e−16**	**4.6045e−8**	**0.0051**	0.1880
		H = 0	**H = 1**	**H = 1**	**H = 1**	**H = 1**	**H = 1**	**H = 1**	H = 0
	BS-ELM	**1.4151e−04**	0.3987	0.9703	**1.0807e−04**	**2.1171e−08**	**7.7365e−6**	**0.0204**	**0.0272**
		H = 1	H = 0	H = 0	**H = 1**	**H = 1**	**H = 1**	**H = 1**	**H = 1**
	BS-H-ELM	**0.0019**	**2.7728e−05**	**1.2856e−05**	**1.3776e−04**	**2.0131e−6**	**8.8911e−8**	**0.0002**	**0.0191**
		H = 1	**H = 1**	**H = 1**	**H = 1**	**H = 1**	**H = 1**	**H = 1**	**H = 1**

Entries displayed in bold and with H = 1 indicate that the proposed algorithm significantly outperforms a specific rival algorithm at a 5% significance level based on the RMSE measurement

Table 9 T-test results on MAE between DEP-TSP^{meta} and other comparative algorithms on the eight benchmark time series datasets

Model A	Model B	MAE							
		IAP	QIS	DHA	DSA	ITD	DJI	Odonovan	Montgome
The proposed algorithm	DES-PALR	**0.0335**	0.1362	0.2501	**0.0172**	**8.5264e−09**	0.1068	0.0840	0.0581
		H = 1	H = 0	H = 0	**H = 1**	**H = 1**	H = 0	H = 0	H = 0
	DES-CP-Clustering	**0.0012**	0.2264	0.8258	0.8644	**9.0605e−10**	0.2542	**0.0067**	0.4666
		H = 1	H = 0	H = 0	H = 0	**H = 1**	H = 0	**H = 1**	H = 0
	DVS-OpOp	0.0602	0.1179	0.1363	0.3436	**7.3287e−12**	**0.0073**	**0.0449**	0.1703
		H = 0	H = 0	H = 0	H = 0	**H = 1**	**H = 1**	**H = 1**	H = 0
	GASEN	**0.0039**	**0.0011**	**0.0107**	0.1578	**8.0353e−09**	**3.6349e−6**	**0.0145**	0.0605
		H = 1	**H = 1**	**H = 1**	H = 0	**H = 1**	**H = 1**	**H = 1**	H = 0
	AA	**0.0097**	**1.8776e−9**	**0.0013**	**8.2967e−12**	**1.9245e−15**	**2.9850e−7**	**0.0064**	0.1619
		H = 1	**H = 1**	**H = 1**	**H = 1**	**H = 1**	**H = 1**	**H = 1**	H = 0
	BS-ELM	**2.6485e−04**	0.1646	**0.0017**	**3.2954e−05**	**2.8638e−07**	**1.7975e−5**	**0.0145**	**0.0160**
		H = 1	H = 0	**H = 1**	**H = 1**	**H = 1**	**H = 1**	**H = 1**	**H = 1**
	BS-H-ELM	**0.0010**	**4.9229e−06**	**4.2404e−07**	**8.1344e−05**	**7.8341e−07**	**2.7308e−7**	**0.0002**	**0.0211**
		H = 1	**H = 1**	**H = 1**	**H = 1**	**H = 1**	**H = 1**	**H = 1**	**H = 1**

Entries displayed in bold and with H = 1 indicate that the proposed algorithm significantly outperforms a specific rival algorithm at a 5% significance level based on the MAE measurement

than the comparative algorithms. In contrast, BS-H-ELM yields the worst results on all the datasets, which could be explained by the lack of training samples. However, the proposed algorithm could overcome this difficulty well. For example, if there are 30 training instances in the meta-predictor construction stage and the collection of predictors consists of 100 base models, then the number of training instances will reach 3,000. Thus, enough learning instances are gotten for training the meta-predictor. In this manner, the performance of the proposed technique will not be limited by the lack of training instances.

It could be observed from the results shown in Table 7 that, the proposed DEP-TSP^{meta} technique achieves results that are superior to other algorithms on almost all the datasets in terms of MAE, except for the DSA and Montgome datasets. While on the DSA and Montgome datasets, DEP-TSP^{meta} acquires the second best MAE performance. Since, in DEP-TSP^{meta}, four different criteria, designed to measure the capacity of base predictors, are employed as the meta-attributes, even though one or two of them do not work well, the integral framework can still achieve excellent performance. In this sense, our framework is a more robust prediction technique than the comparative algorithms considering only one single criterion.

In addition, the algorithms achieving best performances on the datasets are all some ensemble selection methods

Table 10 Friedman test and post hoc Finner test based on RMSE values (significance level of 0.1)

Friedman test								
Statistic			p-value			Result		
13.0597			0.0000			H0 is rejected		
Algorithm	DEP-TSP^{meta}	DES-PALR	DES-CP-Clustering	DVS-OpOp	GASEN	AA	BS-ELM	BS-H-ELM
Ranking	2.0000	3.2500	2.8750	4.0000	4.0000	6.5000	5.5000	7.8750

Post-hoc Finner tests			
Comparison	Statistic	Adjusted p-value	Result
DEP-TSP^{meta} vs DES-PALR	1.0206	0.3486	H0 is accepted
DEP-TSP^{meta} vs DES-CP-Clustering	0.7144	0.4750	H0 is accepted
DEP-TSP^{meta} vs DVS-OpOp	1.6330	0.1724	H0 is accepted
DEP-TSP^{meta} vs GASEN	1.6330	0.1724	H0 is accepted
DEP-TSP^{meta} vs AA	3.6742	0.0008	H0 is rejected
DEP-TSP^{meta} vs BS-ELM	2.8577	0.0099	H0 is rejected
DEP-TSP^{meta} vs BS-H-ELM	4.7969	0.0000	H0 is rejected

Table 11 Friedman test and post hoc Finner test based on MAE values (significance level of 0.1)

Friedman test								
Statistic			p-value			Result		
19.1333			0.0000			H0 is rejected		
Algorithm	DEP-TSP^{meta}	DES-PALR	DES-CP-Clustering	DVS-OpOp	GASEN	AA	BS-ELM	BS-H-ELM
Ranking	1.2500	3.6250	3.2500	3.6250	4.1250	6.5000	5.7500	7.8750

Post-hoc Finner tests			
Comparison	Statistic	Adjusted p-value	Result
DEP-TSP^{meta} vs DES-PALR	1.9392	0.0727	H0 is rejected
DEP-TSP^{meta} vs DES-CP-Clustering	1.6330	0.1025	H0 is accepted
DEP-TSP^{meta} vs DVS-OpOp	1.9392	0.0727	H0 is rejected
DEP-TSP^{meta} vs GASEN	2.3474	0.0329	H0 is rejected
DEP-TSP^{meta} vs AA	4.2866	0.0001	H0 is rejected
DEP-TSP^{meta} vs BS-ELM	3.6742	0.0006	H0 is rejected
DEP-TSP^{meta} vs BS-H-ELM	5.4093	0.0000	H0 is rejected

rather than the method of selecting the best single model, which demonstrates the superiority of ensemble selection paradigm in prediction performance. At the same time, although the DES methods can achieve the best performances on most datasets, static ensemble selection methods, especially GASEN, sometimes have better performance than the DES methods, including the proposed technique, which shows that dynamic ensemble selection algorithms are not always better than static ensemble selection algorithms in any situation. This observation also proves the "No Free Lunch" (NFL) theorem [44].

Next, to make sure whether the proposed DEP-TSP^{meta} technique is superior to the three DES algorithms and other comparative algorithms in a statistic sense, it is necessary to perform *t*-tests on the RMSE and MAE results obtained by all the algorithms on the eight time series benchmark datasets.

Then, when the *t*-tests are performed, the significance level ALPHA is set to 0.05 and TAIL is set to left. The results are shown in Tables 8 and 9. The items displayed in bold and with H = 1 indicate that hypothesis H0 is rejected, i.e., Model A significantly improves the predictive performance of Model B, at 5% significance level (t-value ≤ -1.8331). Conversely, the items in normal font and with H = 0 manifest that hypothesis H0 cannot be rejected, i.e., there is no significant difference between the predictive performance of Model A and Model B at 5% significant level.

As shown in Tables 8 and 9, for 70 out of the 112 *t*-tests (62.5%), the proposed DEP-TSP^{meta} technique achieves significant improvements over comparative algorithms at

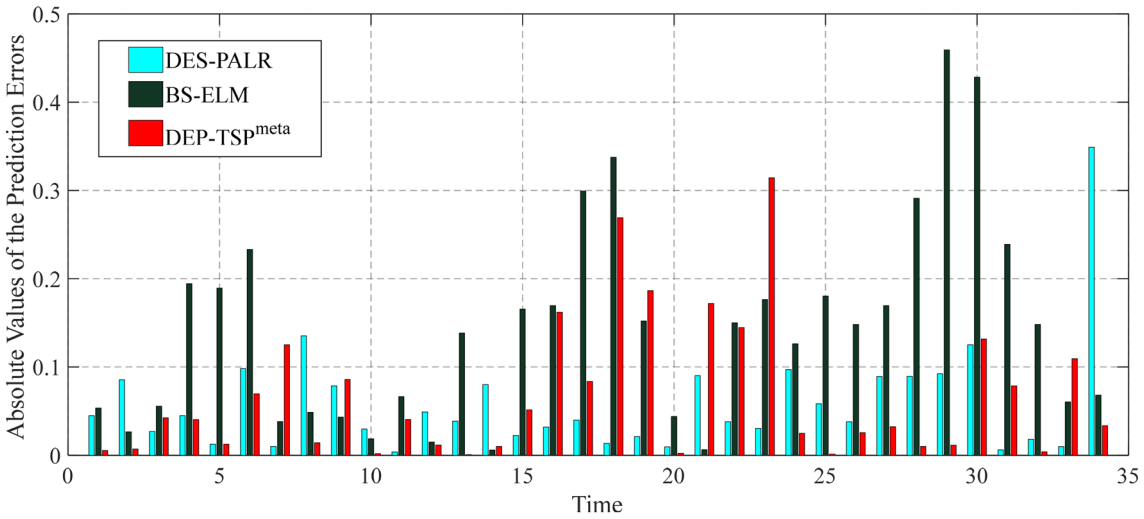

Fig. 5 The absolute values of the prediction errors obtained by DEP-TSP^{meta}, DES-PALR and BS-ELM on the IAP time series

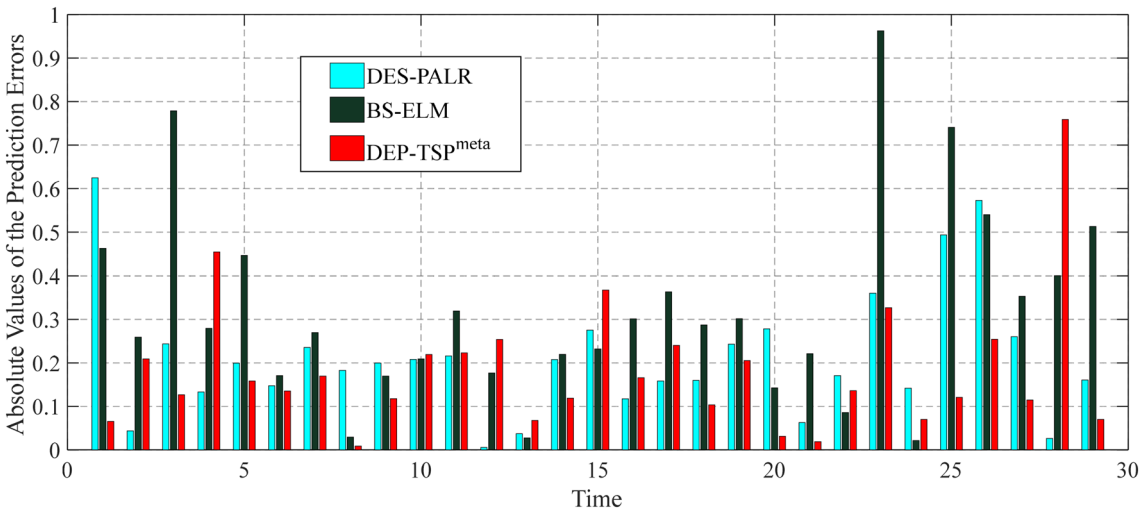

Fig. 6 The absolute values of the prediction errors obtained by DEP-TSP^{meta}, DES-PALR and BS-ELM on the QIS time series

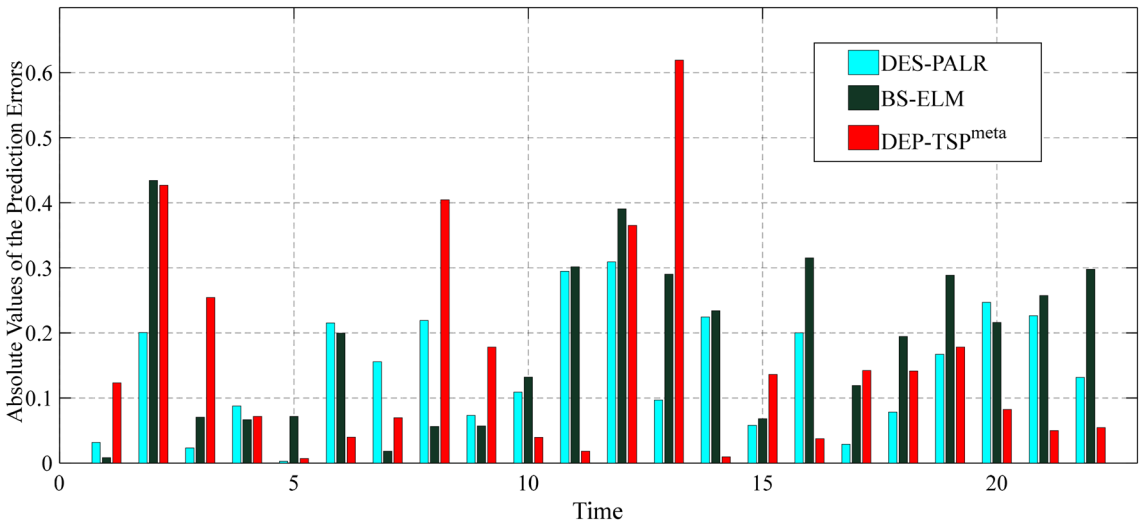

Fig. 7 The absolute values of the prediction errors obtained by DEP-TSP^{meta}, DES-PALR and BS-ELM on the DHA time series

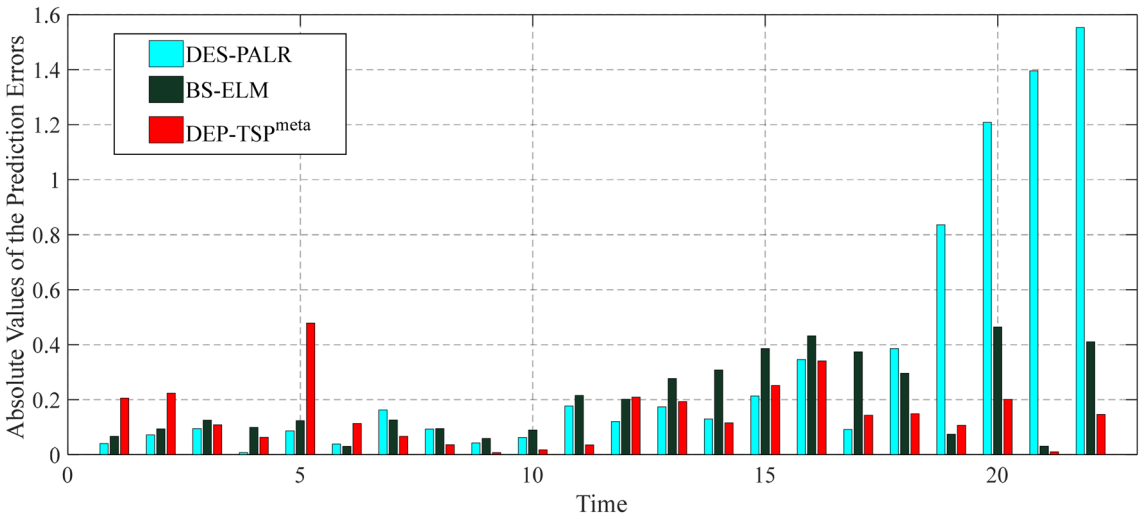

Fig. 8 The absolute values of the prediction errors obtained by DEP-TSP^{meta}, DES-PALR and BS-ELM on the DSA time series

5% significance level in terms of RMSE and MAE. When applied to the ITD and Odonovan time series datasets, DEP-TSP^{meta} is far better than all other compared algorithms, including three DES algorithms and four static ensemble selection techniques. At the same time, DEP-TSP^{meta} achieves significant improvements over BS-H-ELM in terms of RMSE and MAE on all the datasets. These results clearly show that, the proposed technique based on *meta*-learning is applicable to tackle with the TSP problems with small sample datasets.

STAC [50], which is a web platform that provides a more appropriate statistical test, is used, in this work, to determine whether the superiority of our algorithm is accidental. The statistical test results are shown in Tables 10 and 11. Specifically, the Friedman test [51], a nonparametric statistical test, is used to test differences across multiple algorithms based on the rankings of the algorithms on multiple datasets in terms of RMSE and MAE values. For each dataset, the Friedman test ranks each algorithm, where the best performing algorithm is ranked the first, the next best one is ranked the second, and so on. And then, the average ranking of each algorithm on all the datasets is computed. The best algorithm is the one with the lowest average ranking. The null hypothesis for the Friedman test is that the predictive performances of all the algorithms are equivalent or similar. We set the level of significance $\alpha = 0.1$, i.e., 90% confidence. The Friedman test results show that the null hypotheses are rejected with extremely low p-values, and it can be

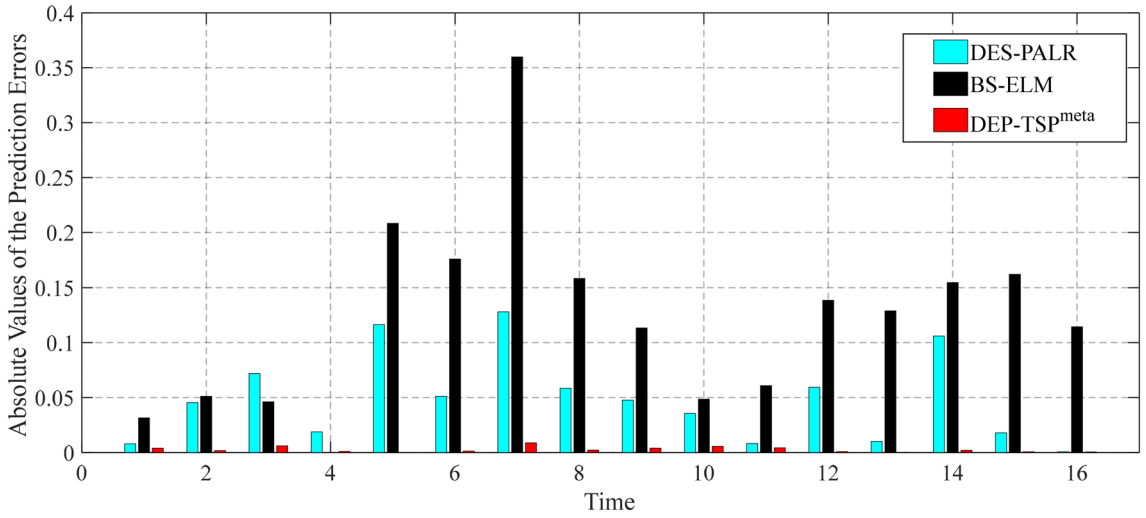

Fig. 9 The absolute values of the prediction errors obtained by DEP-TSP^{meta}, DES-PALR and BS-ELM on the ITD time series

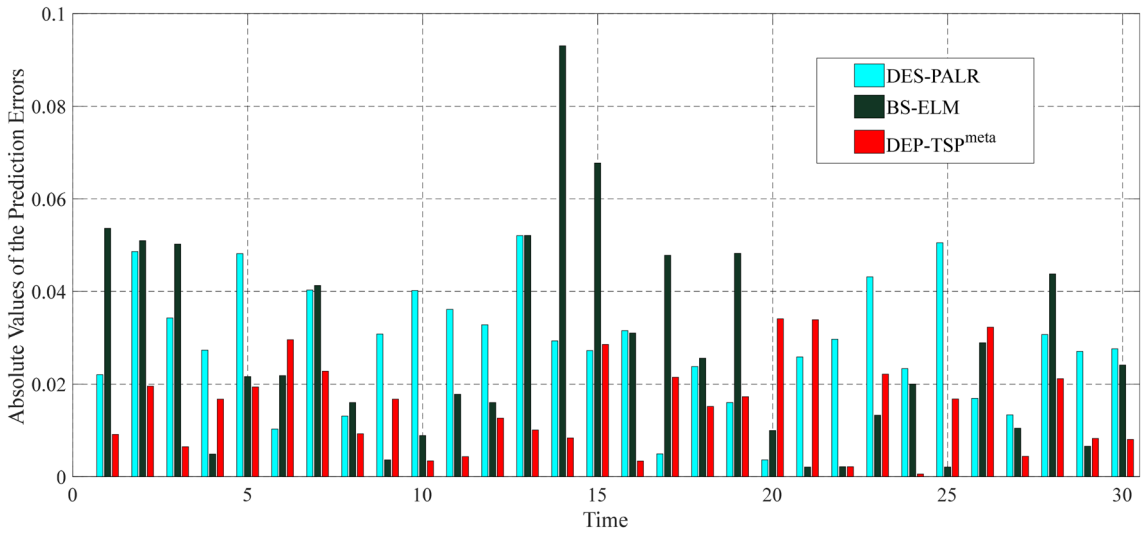

Fig. 10 The absolute values of the prediction errors obtained by DEP-TSP^{meta}, DES-PALR and BS-ELM on the DJI time series

concluded that the predictive performances of at least two of the algorithms are significantly different from each other.

Then, a post hoc Finner test [52] with the level of significance $\alpha = 0.1$ is conducted for a pairwise comparison between the rankings achieved by each algorithm, so as to check whether the performance differences between the proposed DEP-TSP^{meta} algorithm and those comparative algorithms on multiple datasets are statistically significant. In terms of MAE values, DEP-TSP^{meta} obtains the lowest average ranking of 1.25, followed by the DES-CP-Clustering technique, presenting an average ranking of 3.25. From the Finner test results, the performance of DEP-TSP^{meta} is significantly better when compared to the majority of the DES techniques and four static ensembles selection techniques. Only DES-CP-Clustering obtains a statistically equivalent performance. Moreover, as shown in Tables 10 and 11, for 9 out of the 14 Finner tests (64.3%), the predictive performance of the proposed DEP-TSP^{meta} technique is significantly superior with a 90% confidence, when compared to other comparative algorithms, on the eight benchmark datasets, in terms of RMSE and MAE values.

Finally, Figs. 5, 6, 7, 8, 9, 10, 11, 12 display, respectively, the absolute values of the prediction errors obtained by the proposed technique, DES-PALR and BS-ELM on the eight benchmark time series datasets.

From the above comparisons, it can be concluded that the proposed technique has better generalization performance and smaller prediction errors than DES-PALR and BS-ELM

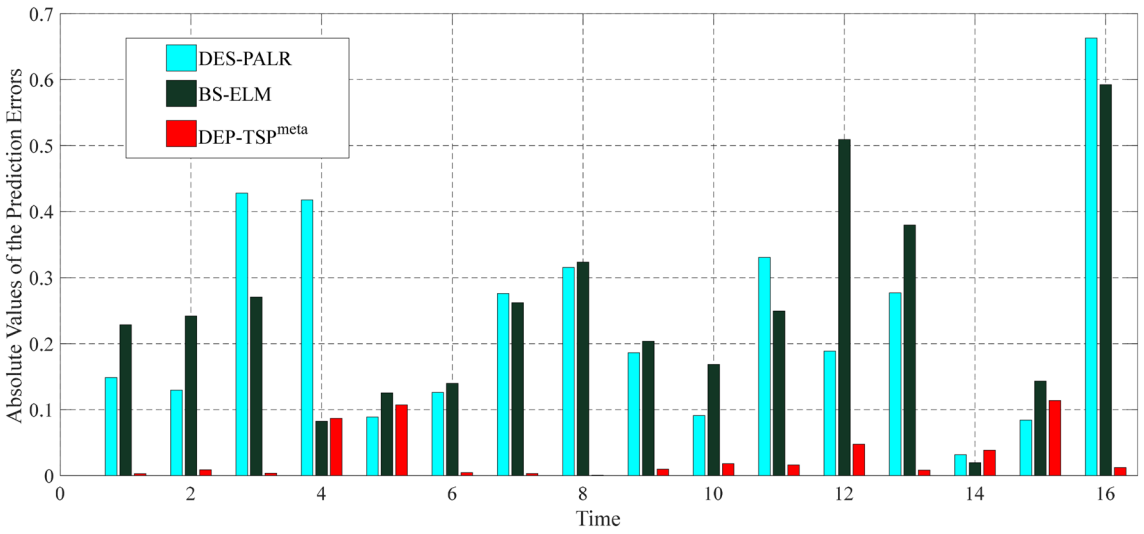

Fig. 11 The absolute values of the prediction errors obtained by DEP-TSP^{meta}, DES-PALR and BS-ELM on the Odonovan time series

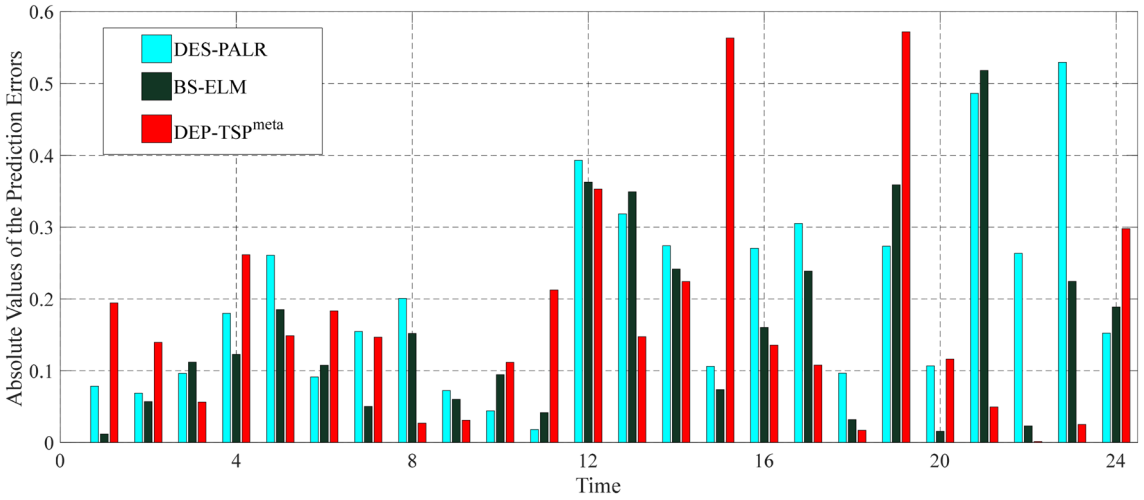

Fig. 12 The absolute values of the prediction errors obtained by DEP-TSP^{meta}, DES-PALR and BS-ELM on the Montgome time series

on the six benchmark TSP problems, except for the DSA and the Montgome datasets. In addition, the proposed DEP-TSP^{meta} technique obtains the prediction values approximating the real values. It is worth mentioning that, the proposed technique achieves results close to the Oracle [53] performance on the ITD dataset. The Oracle expresses an almost perfect pruning scheme and its error rate is approximately zero.

The above reported experimental results are based upon the results of 10 repetitive runs. The primary reason for implementing repeated experiments is to reduce the impact of random factors, such as the random initialization of weights in neural networks, on the performance of the algorithms, and to decrease the influence of accidental errors on performance evaluation. In order to further explore whether the number of repetitions have an effect on the experimental results, we have added a series of controlled experiments with different repetitions, i.e., 5, 10, 15, 20, on the Odonovan dataset. The detailed RMSE and MAE performances with different repetitions of experiments on the Odonovan dataset are, respectively, presented in Tables 12 and 13.

From Tables 12 and 13, it is not hard to see that, whether the number of repetitions is set as 5, 10, 15, or 20, the proposed DEP-TSP^{meta} technique consistently obtains the best performances on the Odonovan dataset, in terms of RMSE and MAE, compared with the other seven techniques, including the three DES algorithms and the four static ensemble selection rules. Therefore, it could be concluded that the

Table 12 The detailed RMSE performance of corresponding algorithms on the Odonovan dataset

RMSE	Repetitions			
Model	5	10	15	20
DEP-TSP^{meta}	**0.2276**	**0.2088**	**0.2047**	**0.2078**
DES-PALR	0.2500	0.2527	0.2563	0.2591
DES-CP-Clustering	0.2501	0.2642	0.2557	0.2546
DVS-OpOp	0.2609	0.2647	0.2633	0.2630
GASEN	0.2550	0.2787	0.2590	0.2632
AA	0.4167	0.3793	0.4131	0.3836
BS-ELM	0.3168	0.3410	0.3365	0.3448
BS-H-ELM	0.5454	0.4266	0.4838	0.4809

The boldface indicates the best RMSE performance obtained by the corresponding algorithm with different repetitions of experiments

Table 13 The detailed MAE performance of corresponding algorithms on the Odonovan dataset

MAE	Repetitions			
Model	5	10	15	20
DEP-TSP^{meta}	**0.1776**	**0.1665**	**0.1624**	**0.1640**
DES-PALR	0.1920	0.1968	0.1923	0.1988
DES-CP-Clustering	0.2054	0.2239	0.2122	0.2129
DVS-OpOp	0.2051	0.2091	0.2071	0.2072
GASEN	0.2093	0.2301	0.2124	0.2161
AA	0.3264	0.2967	0.3316	0.3070
BS-ELM	0.2590	0.2771	0.2690	0.2753
BS-H-ELM	0.4307	0.3404	0.3870	0.3854

The boldface indicates the best MAE performance obtained by the corresponding algorithm with different repetitions of experiments

number of repetitions is changed or dynamic, the comparative experiment results will not change, and the proposed technique still can achieve optimal performance on the Odonovan dataset. When the number of repetitions increases, the performance of our algorithm tends to be more stable. Thus, a rough conclusion could be drawn that, the more numbers of repetitive experiments are conducted, the more reliable the experimental results will be. However, considering the efficiency, ten repetitive experiments are enough to obtain relatively reliable results for analysis.

6 Conclusion and future work

In this paper, a multiple criteria Dynamic Ensemble Pruning technique dedicated to Time Series Prediction applying the *meta*-learning paradigm, namely the DEP-TSP^{meta} technique, is proposed. Four sets of meta-attributes are designed, with each set of meta-attribute corresponding to a specific DEP criterion for evaluating the capacity of each predictor in the initial collection of predictors, such as the predictor accuracy in the local area computed over the feature space, the predictor accuracy in the global area computed over the decision space, and the predictor's confidence. These meta-attributes are utilized to train a meta-predictor. The meta-predictor will be responsible to evaluate whether or not one predictor is capable for predicting the unseen sample. Those incapable predictors determined by the meta-predictor will be pruned, while, in contrast, the capable predictors will be selected to constitute the final dynamic ensemble system.

For different TSP datasets, the size of the local area and global area, and the extent of consensus are totally different, which entails these crucial parameters to be adapted dynamically. After exploiting genetic algorithm to dynamically adjust these key parameters, significant improvement to the prediction accuracy of DEP-TSP^{meta} is obtained.

Experiments are conducted on eight benchmark TSP datasets coming from different fields, including transport, tourism, finance, crime, and labor market, etc. And DEP-TSP^{meta} is compared against three DES techniques (each technique measures the competence level of a base predictor based on a single criterion), as well as four static ensemble selection techniques. Empirical results demonstrate that DEP-TSP-^{meta} has better predictive precision than other techniques on most of the eight datasets. These results may benefit from its designing scheme, the advantages of which are summarized as follows: (1) It dynamically provides the most qualified ensemble system for distinct test instances; (2) It devises four distinct DEP criteria to evaluate the competence of base predictors from different angles. Even if one or two criteria might lose efficacy, it can still keep its validity due to the consideration of other criteria; (3) It automatically generates more meta-knowledge to train the meta-predictor and consequently achieves significant performance gains even though the size of the training dataset is small. These characteristics yield the proposed DEP-TSP^{meta} technique with relatively high effectiveness and robustness.

Furthermore, numerous meta-attribute vectors generated by training instances can be used to train the meta-predictor, consequently, the problem of lack of training instances for meta-predictor could be overcome.

Also, some limitations exist in the proposed DEP-TSP^{meta} algorithm. Firstly, a desirable meta-predictor, obtained by training on the strength of four distinct meta-attributes, is the key to determine whether a base predictor is capable of predicting an unseen instance well or not. However, there are more criteria than just four sets of meta-attributes, which are worth considering for training a meta-predictor. Secondly, multiple DEP criteria are embedded in the proposed DEP-TSP^{meta} algorithm encoded as different sets of meta-attributes. However, some DEP criteria are not suitable for every TSP problem. Different prediction problems may require distinct sets of meta-attributes and the meta-predictor training

process should be optimized for each specific prediction problem. As such, in our future work, we will work on the following aspects for time series prediction: (1) designing novel criteria to evaluate the capacity of the base predictors more effectively, such as the ranking information based on the number of consecutive correct predictions made by the base predictors; (2) developing a new meta-attributes selection scheme based on optimization algorithms, and selecting adaptively an appropriate set of meta-attributes for specific prediction problems, so as to improve the performance of the meta-predictor, and consequently, the predictive accuracy of the algorithm.

Acknowledgements This work is supported by the National Key R&D Program of China (Grant Nos. 2018YFC2001600, 2018YFC2001602), and the National Natural Science Foundation of China under Grant No. 61473150.

References

1. Kayacan E, Ulutas B, Kaynak O (2010) Grey system theory-based models in time series prediction. Expert Syst Appl 37:1784–1789
2. Chen CF, Lai MC, Yeh CC (2012) Forecasting tourism demand based on empirical mode decomposition and neural network. Knowl-Based Syst 26:281–287
3. Xia M, Wong WK (2014) A seasonal discrete grey forecasting model for fashion retailing. Knowl-Based Syst 57:119–126
4. Langella G, Basile A, Bonfante A, Terribile F (2010) High-resolution space-time rainfall analysis using integrated ANN inference systems. J Hydrol 387:328–342
5. Ailliot P, Monbet V (2012) Markov-switching autoregressive models for wind time series. Environ Model Softw 30:92–101
6. Moura MD, Zio E, Lins ID, Droguett E (2011) Failure and reliability prediction by support vector machines regression of time series data. Reliab Eng Syst Saf 96:1527–1534
7. Yu L, Dai W, Tang L (2015) A novel decomposition ensemble model with extended extreme learning machine for crude oil price forecasting. Eng Appl Artif Intell 47:110–121
8. Huang G-B, Zhu Q-Y, Siew C-K (2006) Extreme learning machine: theory and applications. Neurocomputing 70:489–501
9. Wong WK, Guo ZX (2010) A hybrid intelligent model for medium-term sales forecasting in fashion retail supply chains using extreme learning machine and harmony search algorithm. Int J Prod Econ 128:614–624
10. Shrivastava NA, Panigrahi BK (2014) A hybrid wavelet-ELM based short term price forecasting for electricity markets. Int J Electr Power Energy Syst 55:41–50
11. Huixin T, Bo M A new modeling method based on bagging ELM for day-ahead electricity price prediction, In: 2010 IEEE 5th International Conference on Bio-Inspired Computing: Theories and Applications, Changsha, 2010, pp 1076–1079
12. Tang JX, Deng CW, Huang GB (2016) Extreme Learning Machine for Multilayer Perceptron. IEEE Transactions on Neural Networks and Learning Systems 27:809–821
13. Baxt WG (1992) Improving the accuracy of an artificial neural network using multiple differently trained networks. Neural Comput 4:772–780
14. Wedding DK II, Cios KJ (1996) Time series forecasting by combining RBF networks, certainty factors, and the Box-Jenkins model. Neurocomputing 10:149–168
15. Krogh A, Vedelsby J (1995) Neural network ensembles, cross validation, and active learning. In Advances in Neural Information Processing Systems, Denver, pp 231–238
16. dos Santos EM, Sabourin R, Maupin P (2007) Single and multi-objective genetic algorithms for the selection of ensemble of classifiers, In 2006 International Joint Conference on Neural Networks, Vancouver, pp 3070–3077
17. Brown G, Wyatt J, Harris R, Yao X (2005) Diversity creation methods: a survey and categorisation. InfFusion 6:5–20
18. Kraipeerapun P, Fung C, Nakkrasae S (2009) Porosity prediction using bagging of complementary neural networks. In 6th International Symposium on Neural Networks, Berlin, pp 175–184
19. Lysiak R, Kurzynski M, Woloszynski T (2014) Optimal selection of ensemble classifiers using measures of competence and diversity of base classifiers. Neurocomputing 126:29–35
20. Cavalin PR, Sabourin R, Suen CY (2013) Dynamic selection approaches for multiple classifier systems. Neural Comput Appl 22:673–688
21. Cruz RMO, Sabourin R, Cavalcanti GDC (2018) Prototype selection for dynamic classifier and ensemble selection. Neural Comput Appl 29:447–457
22. Yao CS, Dai Q, Song G (2019) Several novel dynamic ensemble selection algorithms for time series prediction. Neural Process Lett 50:1789–1829
23. Cruz RMO, Sabourin R, Cavalcanti GDC (2018) Dynamic classifier selection: recent advances and perspectives. Inf Fusion 41:195–216
24. Cruz RMO, Sabourin R, Cavalcanti GDC, Ren TI (2015) META-DES: a dynamic ensemble selection framework using meta-learning. Pattern Recogn 48:1925–1935
25. Cruz RMO, Sabourin R, and Cavalcanti GDC, META-DES.H: a dynamic ensemble selection technique using meta-learning and a dynamic weighting approach, In 2015 International Joint Conference on Neural Networks, New York, 2015, pp 1–8
26. Cruz RMO, Sabourin R, Cavalcanti GDC (2017) META-DES. Oracle: meta-learning and feature selection for dynamic ensemble selection. Inf Fusion 38:84–103
27. Huang GB (2014) An insight into extreme learning machines: random neurons, random features and kernels. Cogn Comput 6:376–390
28. Huang GB, Zhou HM, Ding XJ, Zhang R (2012) Extreme learning machine for regression and multiclass classification. IEEE Trans Syst Man Cybern Part B 42:513–529
29. Bengio Y (2009) Learning deep architectures for Al. Found Trends Mach Learn 2:1–127
30. Vincent P, Larochelle H, Bengio Y, Manzagol PA (2008) Extracting and composing robust features with denoising autoencoders, In International Conference on Machine Learning, Helsinki, Finland, p. 1096–1103
31. Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science 313:504–507
32. Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. Neural Comput 18:1527–1554
33. Bengio Y, Courville A, Vincent P (2013) Representation Learning: a Review and New Perspectives. IEEE Trans Pattern Anal Mach Intell 35:1798–1828
34. Huang GB, Li MB, Chen L, Siew CK (2008) Incremental extreme learning machine with fully complex hidden nodes. Neurocomputing 71:576–583
35. Beck A, Teboulle M (2009) A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. Siam Journal on Imaging Sciences 2:183–202
36. Beck A, Teboulle M (2009) Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. IEEE Trans Image Process 18:2419–2434

37. Cruz RMO, Cavalcanti GDC, Ren TI, IEEE (2011) A method for dynamic ensemble selection based on a filter and an adaptive distance to improve the quality of the regions of competence, In 2011 International Joint Conference on Neural Networks, San Jose, CA, pp 1126–1133
38. Zhou ZH, Wu J, Jiang Y, Chen S (2001) Genetic algorithm based selective neural network ensemble, In International Joint Conference on Artificial Intelligence, Seattle, Washington, USA, pp 797–802
39. Whitley D (1994) A genetic algorithm tutorial. Stat Comput 4:65–85
40. Horn J, Nafpliotis N, Goldberg DE (1994) A niched Pareto genetic algorithm for multiobjective optimization. In Proceedings of 1th IEEE Conference on Evolutionary Computation, Orlando, FL, USA, pp 82–87
41. Deb K, Agrawal S, PratapA, Meyarivan T (2000) A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In Proceedings of 6th International Conference on Parallel Problem Solving from Nature, Paris, France, pp 849–858
42. Rudolph G (1994) Convergence analysis of canonical genetic algorithms. IEEE Trans Neural Networks 5:96–101
43. Corne DW, Knowles JD (2003) No free lunch and free leftovers theorems for multiobjective optimisation problems. In International Conference on Evolutionary Multi-Criterion Optimization, pp 327–341
44. Ho YC, Pepyne DL (2002) Simple explanation of the no-free-lunch theorem and its implications. J Optim Theory Appl 115:549–570
45. Hyndman R, Yang Y. *tsdl:* Time Series Data Library. v0.1.0. https://pkg.yangzhuoranyang.com/tsdl/
46. Yahoo Finance[EB/OL]. https://finance.yahoo.com/
47. Mean absolute .https://en.wikipedia.org/wiki/Mean_absolute_error
48. Root-mean-square deviation. https://en.wikipedia.org/wiki/Root-mean-square_deviation
49. Arlot S, Celisse A (2010) A survey of cross-validation procedures for model selection. Stat Surv 4:40–79
50. Rodriguez-Fdez I, Canosa A, Mucientes M, Bugarin A (2015) STAC: A web platform for the comparison of algorithms using statistical tests. In IEEE International Conference on Fuzzy Systems, Istanbul, Turkey, pp 1–8
51. Demiar J, Schuurmans D (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
52. Finner H (1993) On a monotonicity problem in step-down multiple test procedures. J Am Stat Assoc 88:920–923
53. Kuncheva LI (2002) A theoretical study on six classifier fusion strategies. IEEE Trans Pattern Anal Mach Intell 24:281–286