IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 13, 2019, accepted March 3, 2019, date of publication March 18, 2019, date of current version April 2, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2903839

Semi-Supervised Self-Training Method Based on an Optimum-Path Forest

JUNNAN LI AND QINGSHENG ZHU
Department of Computer Science, Chongqing University, Chongqing 400044, China
Corresponding author: Qingsheng Zhu (qszhu@cqu.edu.cn)

This work was supported in part by the Natural Science Foundation of China under Grant 61272194 and Grant 61502060, and in part by the Chongqing Science and Technology Project under Grant KJZH17104 and Grant cstc2017rgzn-zdyfx0040.

ABSTRACT Semi-supervised self-training method can train an effective classifier by exploiting labeled and unlabeled samples. Recently, a self-training method based on density peaks of data (STDP) is proposed. However, it still suffers from some shortcomings to be addressed. For example, STDP is affected by cut-off distance d_c. As a result, it is tricky for STDP to select an optimal parameter d_c on each data set. Furthermore, STDP has a poor performance on data sets with some variations in density because of cut-off distance d_c. In order to solve these problems, we present a new self-training method which connects unlabeled and labeled samples as vertexes of an optimum path forest to discover the underlying structure of feature space. Furthermore, the underlying structure of the feature space is used to guide the self-training method to train a classifier. Compared with STDP, our algorithm is free of parameters and can work better on data sets with some variations in density. Moreover, we are surprised to find that our algorithm also has some advantages in dealing with overlapping data sets. The experimental results on real data sets clearly demonstrate that our algorithm has better performance than some previous works in improving the performance of base classifiers of k-nearest neighbor, support vector machine and cart.

INDEX TERMS Self-training method, semi-supervised classification, optimum-path forest, semi-supervised learning.

I. INTRODUCTION

A common task of classification requires not only a powerful classifier, but also sufficient labeled samples. In applications, experts are usually required to manually mark unlabeled samples in order to obtain enough labeled samples. But it's not realistic to get a lot of labeled samples in this way. As a result, it is often the case that a small number of labeled samples and a large number of unlabeled samples are available. Semi-supervised classification (SSC) [1], a learning paradigm aiming to find a solution to promote performance of classification by exploiting unlabeled data, can solve this problem.

There are many methods for SSC. Main methods include graph-based method [2], [3], generating model [4], transductive support vector machine [5], co-training method [6], [7] and self-training method [8], [9]. One of the relatively successful methodologies is self-training method. Compared with other methods of SSC, self-training method which doesn't need special assumptions and only teaches itself with self-predicted samples has been successfully applied to cross-lingual sentiment classification [10], text classification [11], image detection and segmentation [12], information extraction [13], automatic sentence segmentation [14], etc.

The self-training method can generally be regarded as a wrapped algorithm. Various classifiers, such as k-nearest neighbor (KNN) [15], support vector machine (SVM) [16], naive bayes (NB) [17], cart (Cart) [18], etc., are well trained by the self-training method. In addition, self-training method can also be seen as a self-taught algorithm. It iteratively predicts unlabeled samples, and then selects unlabeled samples with high confidence to expand the training set. However, the main problem of the self-training method is that it may mislabel unlabeled samples. If incorrectly marked samples are added to the training set, the predictive ability of the trained

The associate editor coordinating the review of this manuscript and approving it for publication was Hongbin Chen.

2169-3536 © 2019 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

classifier will be deteriorated. Therefore, many scholars have proposed using ensemble learning [9], [19], [20], [21] or the technology of noise filters [8], [15], [22], [23] to improve the predictive ability of classifiers.

In fact, there is an important reason for mislabeling that self-training method is limited by distribution and the number of labeled samples [18]. When labeled samples doesn't represent the structure of feature space, a result of mislabeling may occur by using the self-training method based on strategy of confidence. To solve this problem, lots of improved algorithms based on self-training method have been proposed. Help-training, a new framework for self-training method, was proposed by Adankon and Cherie [16], where a supervised classifier of NB is used to help train a supervised classifier of SVM. Although NB can help train a good SVM, NB just uses labeled samples as the training set without considering information of a large number of unlabeled samples. Therefore, Gan *et al.* [24] proposed that using semi-supervised fuzzy C means (SFCM) [25] to help self-training method (STSFCM), where the structure of feature space revealed by SFCM is used to guide self-training method. But STSFCM can't train a good supervised classifier when the distribution of data is non-spherical. To solve this problem, Wu et al. [18] propose using density peak cluster (DPC) [26] to help self-training method (STDP), where the structure of feature space discovered by DPC rather than SFCM is used to guide self-training method. Although STDP can train an effective classifier on data sets with distribution of arbitrary shapes (non spherical or spherical), it is also limited by some shortcomings of DPC, such as the parameter d_c. Therefore, it's tricky for STDP to select an optimal parameter d_c on each data set. Moreover, STDP have poor performance on data sets with some variations in density because it is difficult for DPC to discover the real structure of feature space on multi-scale data [27], [28].

Recently, a robust model of graph, optimum path forest (OPF) [29]–[31] is proposed and further studied, where all samples are interconnected as vertexes of an optimum path forest. Compared with DPC, OPF is free of parameters and can also discover the real structure of feature space on data sets with distribution of arbitrary shapes. Main advantages of OPF are that (a) Parameters free; (b) It can work well without considering shapes and distribution of data sets; (c) It can deal data sets with some overlapping between classes as long as the roots of forests can protect their respective classes well; (d) It can work in a natural way under multi-class circumstances.

In order to solve the problem of STDP, inspired by OPF, we propose a new self-training method based on an optimum path forest (STOPF) in this paper. Overall, STOPF consists of three main steps. The first step is that we construct an OPF over the entire data set to reveal the structure and distribution of feature space. At the second step, the structure and distribution of feature space is used to help self-training method mark unlabeled samples. Then these samples are used to expand the labeled data. At the third step, a given supervised classifier can be trained with extended labeled data. The results of contrasted experiments on real data sets clearly certify that STOPF works better than some previous algorithms and is effective in enhancing the ability of some base classifiers including KNN, SVM and Cart. Compared to previous algorithms, chief strengths of STOPF are that (a) STOPF does not require any user-defined parameters; (b) STOPF can train an effective supervised classifier without considering assumptions about distribution and shapes of data sets; (c) STOPF has a better result than previous works on data sets with some variations in density; (f) STOPF doesn't require the measurement of confidence; (e) Furthermore, we are surprised to find that STOPF also has a good performance on overlapping data sets as long as initial labeled samples can protect their respective classes. Chief contributions of this paper are as following:

(a) We propose a new self-training method named STOPF. It can use OPF to discover the spatial structure of feature space, and then use this information of spatial structure to improve the performance of the self-training method.

(b) Theoretical analysis and detailed design for STOPF.

(c) Concrete empirical conclusions conduct on real data sets.

To the author's best knowledge, such efforts have never been seen in any prior work.

The rest of the paper is organized as follows. In Section 2, we describe how OPF reveals the structure of feature space. In Section 3, we describe our algorithmic framework; In Section 4, we conduct contrasted experiments to prove the effectiveness of our algorithm on real data sets. In Section 5, we conclude this paper and make some plans for the future.

II. REVEALING STRUCTURE OF FEATURE SPACE BY THE OPTIMUM PATH FOREST

A. BACKGROUND

The $X = \{x_i\}_{i=1}^{n}$ is the training set of n d-dimensional samples. In SSC, few samples are marked, and a large number of samples are unmarked. Let $L = (x_1, x_2, \ldots, x_l)$ be labeled samples in X, while let $U = (x_{l+1}, x_{l+2}, \ldots, x_n)$ be unlabeled samples in X ($X = L + U$). At the same time, The Y is the class label set and m is the number of class labels in Y. Let $l : X \rightarrow Y$ be the function mapping each instance x_i to its class label $l(x_i)$.

B. OPTIMUM PATH FOREST

OPF is a graph structure of forests consisting of several trees, and has the ability to reveal the underlying structure of feature space. Moreover, it has been developed to the design of semi-supervised, unsupervised and supervised pattern classifiers [29], [31], [32]. First of all, we define an adjacency relation $G = X \times X$ regarded as a complete and weighted graph (X, G, d), where the d usually calculated by Euclidean distance is the weight between any two samples. Next, we turn the G into an OPF by minimizing

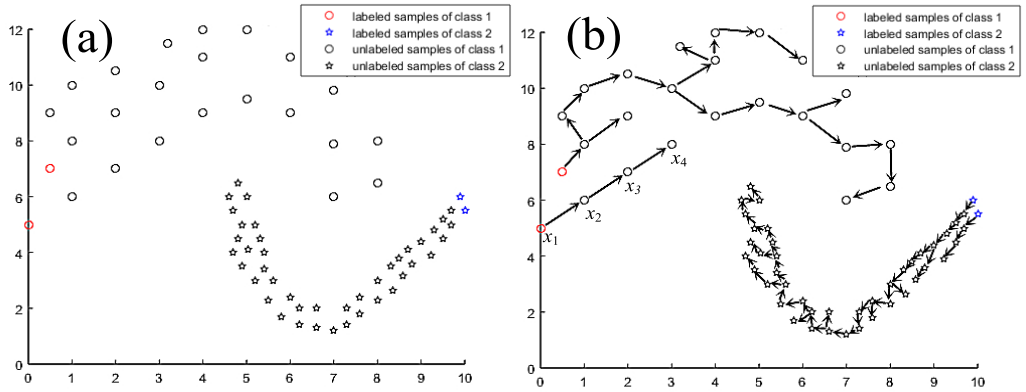

FIGURE 1. Illustration of discovering feature space: (a) distribution of data; (b) real structure of feature space revealed by an OPF.

the f_{max}.

$$f_{\max}(\langle s\rangle) = \begin{cases} 0 & \text{if } s \in L, \\ +\infty & otherwise \end{cases} \tag{1}$$

$$f_{\max}(\pi_s \cdot \langle s, t\rangle) = \max\{f_{\max}(\pi_s), d(s, t)\}.$$

$$C(t) = \min_{\forall \pi_t \in (X,G)} \{f_{\max}(\pi_t)\} \tag{2}$$

The whole process of path's transition is similar to Dijskra's algorithm, where the path relationship is defined as $\pi_t = \langle s_1, s_2, \ldots, s_{k-1}, t\rangle$ with terminus $t = s_k$ and $(s_i, s_{i+1}) \in G$ for $1 \leq i \leq k-1$. A path is said trivial if $\pi_t = \langle t\rangle$. Note that a path π_t is considered to optimum if $f(\pi_t) \leq f(\tau_t)$ for any other path τ_t with the same terminus t. In addition, we write $\pi_s \cdot \langle s, t\rangle$ to indicate the concatenation of a path π_s and arc (s, t). Usually, roots of OPF can be revealed by exploiting the neighbor relationship between Minimum Spanning Tree (MST) [33]. In this paper, we set all labeled samples as roots of OPF because of the environment of SSC [29]. Algorithm 1 shows the pseudo code of the OPF. The OPF's algorithm assigns a map P of optimum path from every sample $x_i \in L$ to some samples in U, forming an optimum path forest. The map P is a matrix with n rows and 1 column. If sample x_i is the predecessor node of x_j, $P(x_j) \leftarrow x_i$.

The time complexity of OPF's algorithm is $O(n^2)$, where n is the number of samples in X. Fig. 1 show an illustration of discovering the structure and distribution of feature space.

Fig. 1 (a) shows non spherical data with some variations in densities, where 4 labeled samples come from two classes. Then, an OPF is constructed on entire data, as shown in Fig. 1 (b). Note that each sample x_i points to its corresponding successor nodes x_j. For example, the sequenced relationships of samples x_1, x_2, x_3, x_4 are interconnected. By constructing an OPF on X, the real structure of feature space can be discovered, regardless of spherical distribution or non-spherical distribution of data. Even the structure and distribution of data sets with some variations in density can also be revealed, as shown in Fig. 1 (b). Although STDP can also use DPC to reveal the real structure of feature space with distribution of arbitrary shapes, DPC is limited by parameter d_c. Compared to DPC, OPF is free of parameters. In addition, it's difficult for DPC to discover real spatial structure on data sets with some variations in density because DPC calculates density by cut-off distance d_c [27], [28], [34]. However, OPF doesn't have this limitation. Therefore, our algorithm combined with OPF has obvious theoretical advantages. In the next section, we will explain these advantages with toy examples and introduce how STOPF uses this the structure and distribution of feature space revealed by the OPF to train a satisfactory classifier.

Algorithm 1 OPF

Input: L, U
Output: a map P of optimum path forest
Auxiliary: a priority queue Q
Method:
1. ***For each*** sample $x_i \in X$
2. set $C(x_i) \leftarrow +\infty$.
3. ***If*** $x_i \in L$, ***then***
4. $C(x_i) \leftarrow 0$, $P(x_i) \leftarrow nil$ and Insert t in Q.
5. ***While*** Q is not empty, ***do***
6. Remove from Q a sample s such that $C(s)$ is minimum.
7. ***For each*** $t \in X$ ***such that*** $C(t) > C(s)$, ***do***
8. Compute $cst \leftarrow \max\{C(s), d(s, t)\}$.
9. ***If*** $cst < C(t)$, ***then***
10. ***If*** $C(t) \neq +\infty$, ***then*** remove t from Q.
11. $P(t) \leftarrow s$, $C(t) \leftarrow cst$. Insert t in Q.
12. ***Return*** P.

III. PROPOSED ALGORITHM

In this section, we propose a framework for our algorithm. Like STDP, our proposed algorithms uses OPF to discover real distribution and structure of feature space to guide self-training method. However, the differences are that (a) The process of revealing the structure of feature space is free of parameters in our algorithm; (b) Our algorithm can use OPF to discover more real structure and distribution of feature space to help train an effective supervised classifier.

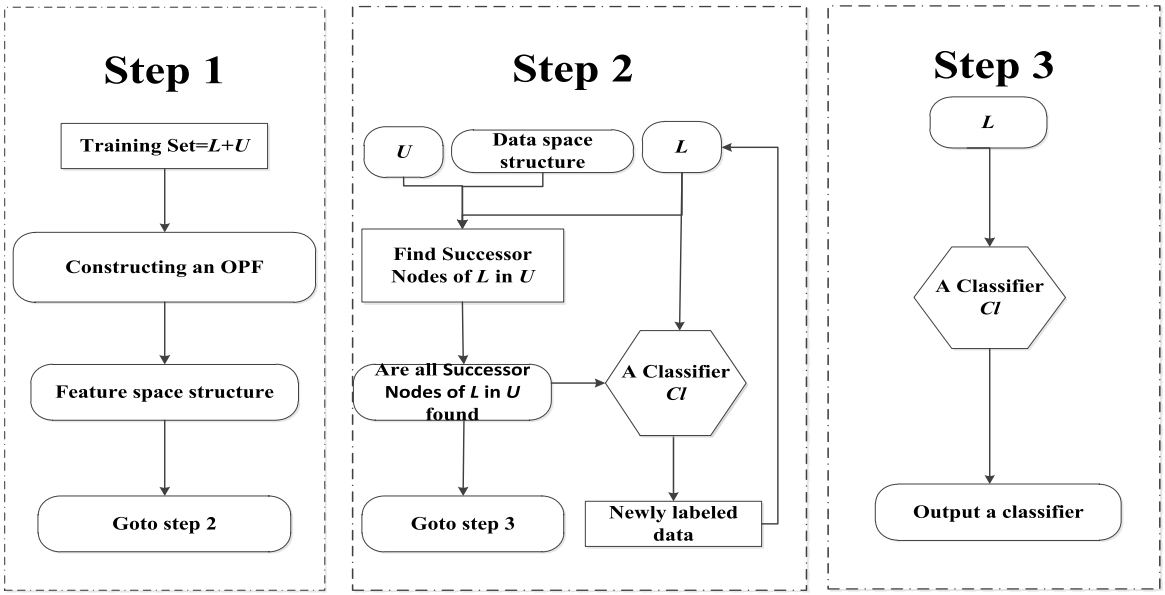

FIGURE 2. A flowchart of our algorithm.

In practice, we have also found that the given classifier trained by STOPF is more accurate than that trained by STDP. Fig. 2 shows the flowchart of our algorithm, which consists of three parts. At the first step, we build an OPF on the X to discover this structure of feature space. At the second step, we use the structure of feature space to guide self-training method to mark unlabeled samples, where successor nodes of each samples $x_i \in L$ in U are chose to be iteratively marked by a classifier Cl. Then, we extend L with the new labeled samples and update U. At the final step, we retrain a classifier Cl with the extended L. After that, a good classifier Cl can be trained. Note that (a) Any classifier, such as KNN, SVM and Cart, can be trained well; (b) Although the measurement of confidence is not clearly given, successor nodes of L are actually samples with high confidence because they are relatively close to L.

The pseudo-code of STOPF is described as follows:

Algorithm 2 Self-Training Method Based on an Optimum Path Forest (STOPF)

Input: L, U
Output: a classifier Cl
Method:
1. P=OPF (L, U).
2. Train a classifier Cl using L.
3. **Repeat** until all the successor nodes of L are selected from U.
4. Select a dataset T from U, where each sample x_i in T is the successor node of L.
5. Label the samples of T using the trained classifier Cl.
6. Update the current labeled dataset $L = L \cup T$.
7. Update the current unlabeled dataset $U = U - T$.
8. Retrain classifier Cl using L.
9. A classifier Cl is trained using L.
10. **Output** Cl.

Fig. 3 shows two toy examples of label propagation of STOPF and STDP, where their associated base classifier is KNN with $k = 3$. Note that label propagation refers to that using self-training method iteratively mark unlabeled samples. In SSC, only if unlabeled samples are marked correctly, a given classifier will be trained satisfactorily.

Fig. 3 (a) shows an artificial data set with some variations in density, and the corresponding result of label propagation in Fig 3 (b)-(c) shows that STOPF is better than STDP. This result isn't so surprising because STOPF can use OPF to find a better spatial structure on data sets with some variations in density. According to previous knowledge, OPF can handle with data with some overlapping between classes as long as roots of forests can protect their respective classes well. We are also surprised to find STOPF can also work well on overlapping data sets as long as initial labeled samples can protect their respective classes because STOPF inherits some characteristics of OPF. Fig. 3 (d) shows an artificial data set with some overlapping, where labeled samples of two classes are in boundary area and protect their respective classes well. As Wu *et al.* [19] mentioned, it's difficult for STDP to deal with overlapping data sets. So, In Fig. 3 (e)-(f), label propagation of STOPF is better than STDP on the toy data set with some overlapping.

In the next section, we will use specific experiments to prove that our proposed algorithm performs better than previous works.

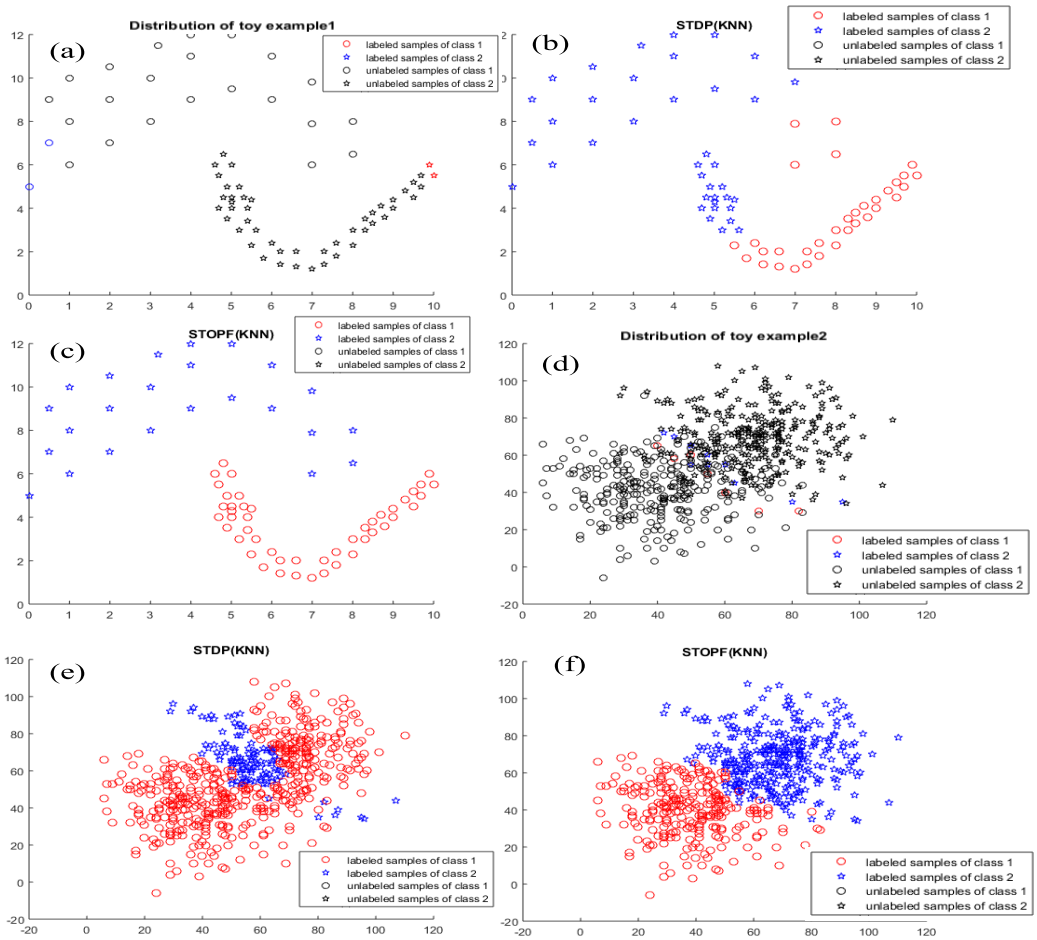

FIGURE 3. Two toy examples of label propagation.

IV. EXPERIMENTAL RESULTS AND DISCUSS

A. DATA SETS

All experiments are run on a PC with a 64-bit operating system and 8G memory. The code for all algorithms is written with MATLAB2015.

In order to illustrate the practical superiority of our algorithm, we select 12 benchmark data sets from the University of California Irvine (UCI) repositories (http://archive.ics.uci.edu/ml/datasets.html), as shown in Table 1.

B. COMPARISONS BETWEEN OUR ALGORITHM AND SOME PREVIOUS WORKS

In order to prove the effectiveness of our algorithm, we run an experiment to compare our algorithm with previous works. Some representative algorithms we select are shown in Table 2. Moreover, specific descriptions can be described as follows:

(1) The tri-training (TT) proposed by Zhou and Li [35] is an extension of self-training method, which combines three classifiers to accomplish the self-taught process of self-training method. Three classifiers play a synergistic role.

(2) The self-training with SFCM (STSFCM) is proposed by Gan *et al.* [24], where SFCM, a clustering algorithm, is used to guide self-training method to train an effective classifier.

(3) The self-training with DPC (STDP) is proposed by Wu *et al.* [18]. A desirable classifier can be trained by the STDP, where the structure of feature space revealed by DPC is integrated into the self-training iterative process to help train an effective classifier.

(4) Base classifiers of contrasted algorithms are 3NN, SVM and Cart. In other words, contrasted algorithms are used to train supervised classifiers of 3NN, SVM and Cart. In addition, we also use supervised algorithms (3NN, SVM and Cart) as contrasted algorithms in our experiments, where supervised algorithms use only original labeled samples as the training set.

At the experimental stage, the 10-fold cross-validation strategy is adopted. Firstly, each data set is randomly divided into 10 folds. Secondly, The training set contains 9 folds while the testing set contains one fold. Thirdly, we use

TABLE 1. Description of experimental data sets.

No.	Data sets	Examples	Attributes	Classes	Abbreviation
1	Website Phishing	1353	9	3	WEP
2	User Knowledge Modeling	403	5	5	UKM
3	Qualitative Bankruptcy	250	6	2	QUB
4	IRIS	150	4	3	IRIS
5	Ionosphere	351	34	2	ION
6	Australian Credit Approval	690	14	2	ACA
7	spambase	4601	57	2	SPA
8	Vehicle	848	18	4	VEH
9	Mammographic masses	961	5	2	MAM
10	Tic Tac Toe Endgame	958	9	2	TTE
11	Segmentation	2310	19	7	SEG
12	Audio	776	26	2	AUD

TABLE 2. Contrasted algorithms and parameters.

Mark	Algorithms	Parameters
3NN	KNN	k=3
SVM	SVM	LibSVM: default parameters
Cart	cart	Default parameters
TT	Tri-training [35]	
STSFCM	Self-training with SFCM [24]	Threshold ε_1 =1/ (the number of classes)
STDP	Self-training with DPC [18]	P_a=2. Please refer to the original literature
STOPF	Our algorithm	

TABLE 3. Experimental results (MAR, Wilcoxon test and STD) of contrasted algorithms with a base classifier 3NN.

Data sets	3NN	TT	STSFCM	STDP	STOPF
WEP	79.82 = (2.94)	78.42 = (3.40)	80.12 = (2.86)	79.46 = (2.53)	**80.19** (2.04)
UKM	63.29 - (7.80)	60.32 = (6.28)	64.04 - (9.12)	**64.29** - (6.78)	58.30 (4.99)
QUB	78.00 + (10.20)	66.80 + (13.20)	80.40 + (7.17)	81.20 + (6.55)	**86.80** (4.64)
IRIS	95.33 = (7.73)	95.33 = (7.70)	95.33 = (834)	94.67 = (5.26)	**96.00** (5.62)
ION	70.66 + (4.83)	70.39 + (8.66)	71.52 + (5.51)	70.94 + (3.95)	**76.67** (7.27)
ACA	64.64 = (6.56)	65.07 = (7.71)	64.78 = (6.70)	**65.36** = (4.40)	63.48 (7.09)
SPA	**67.49** = (2.61)	66.40 = (2.66)	66.25 = (2.30)	65.90 = (2.28)	66.79 (2.73)
VEH	54.24 = (5.59)	52.40 = (5.50)	54.48 = (4.55)	53.90 = (4.49)	**56.61** (6.63)
MAM	76.70 + (4.83)	75.14 + (4.49)	76.90 = (4.72)	78.67 = (5.48)	**79.41** (6.39)
TTE	57.42 + (4.47)	56.69 + (4.38)	56.06 + (5.30)	63.78 + (5.15)	**69.62** (5.43)
SEG	84.72 + (1.39)	80.78 + (1.57)	85.54 + (1.89)	85.28 + (1.32)	**86.80** (1.45)
AUD	**87.63** = (2.99)	86.98 = (3.24)	87.11 = (3.65)	86.47 = (4.21)	86.60 (3.34)
Average MAR	73.33	71.23	73.54	69.16	**75.61**
Wine/lose/tie	4/1/7	5/0/7	4/1/7	4/1/7	

random stratified selection to divide training set into labeled part L and unlabeled part U, which means that the selected number of samples for each class is proportional to the number of them in the training set. Especially, in the training set, the percentage of labeled samples is 10%, and the rest is unlabeled samples. Fourthly, the experiment is repeated 10 times on each data set. The mean accuracy rate (MAR) and standard deviation (STD) on test set are used to evaluate the performance of contrasted algorithms on each data set. Tables 3-5 shows some experimental results to demonstrate that our proposed algorithm, compared with representative algorithms, can better training some classifiers including 3NN, SVM and Cart.

$$AR = \frac{\text{The number of samples correctly classified in test set}}{\text{The number of samples in test set}} \times 100\% \quad (3)$$

$$MAR = \frac{1}{n}\sum_{i=1}^{n} AR_i \quad (n = 10) \quad (4)$$

$$STD = \sqrt{\frac{1}{n}\sum_{i}^{n}(AR_i - MAR)^2} \quad (n = 10) \quad (5)$$

In order to further analyze the results in the Tables 3-5, we also carry out the Wilcoxon test with the significance level $\alpha = 0.05$ to conduct the statistical analysis. Symbol "+" denotes that STOPF outperforms the given algorithm, while "−" denotes that the given algorithm is significantly better than STOPF, and finally "=" denotes that there is no statistically significant difference between the STOPF and given algorithm. Those results are also concluded as "win/lose/tie" at the bottom of Tables 3-5.

TABLE 4. Experimental results (MAR, Wilcoxon test and STD) of contrasted algorithms with a base classifier SVM.

Data sets	SVM	TT	STSFCM	STDP	STOPF
WEP	81.90 = (4.11)	81.85 = (4.78)	81.75 = (3.92)	81.75 = (4.08)	**82.10** (4.07)
UKM	46.20 + (7.41)	47.68 + (7.02)	47.92 + (7.42)	54.87 = (6.46)	**55.62** (6.69)
QUB	90.40 = (5.72)	90.00 = (7.11)	92.00 = (4.22)	81.60 + (6.65)	**92.40** (7.41)
IRIS	94.67 = (7.57)	90.00 = (11.00)	94.67 = (6.13)	**96.67** = (4.71)	**96.67** (4.71)
ION	74.35 + (7.65)	73.77 + (7.41)	74.07 + (7.56)	73.79 + (7.84)	**84.32** (7.18)
ACA	59.86 = (7.64)	59.28 = (7.68)	59.86 = (7.64)	59.42 = (8.17)	**60.00** (8.18)
SPA	**67.49** = (2.61)	66.40 = (2.26)	66.25 = (2.30)	65.90 = (2.28)	66.79 (2.73)
VEH	**36.40** = (4.72)	31.45 = (3.92)	32.74 = (2.65)	33.56 = (2.51)	33.68 (3.43)
MAM	75.03 + (4.39)	72.12 + (5.36)	76.38 = (4.54)	72.85 + (3.96)	**78.25** (4.90)
TTE	52.40 = (3.85)	**52.61** = (3.81)	**52.61** = (3.81)	**52.61** = (3.81)	**52.61** (3.81)
SEG	**40.78** = (3.00)	32.39 + (2.25)	40.56 = (2.85)	34.16 + (2.45)	38.23 (3.05)
AUD	63.67 = (5.77)	52.59 + (3.38)	39.31 + (5.30)	39.31 + (5.30)	**64.44** (5.33)
Average MAR	65.26	62.51	63.18	62.21	**67.09**
Wine/lose/tie	3/0/9	5/0/7	3/0/9	6/0/7	

TABLE 5. Experimental results (MAR, Wilcoxon test and STD) of contrasted algorithms with a base classifier Cart.

Data sets	Cart	TT	STSFCM	STDP	STOPF
WEP	82.18 = (3.99)	**83.88** = (3.45)	82.18 = (3.99)	82.03 = (4.55)	82.11 (4.05)
UKM	**79.16** = (7.35)	77.16 = (7.70)	79.41 = (6.84)	78.16 = (7.50)	77.46 (6.54)
QUB	97.20 = (3.29)	88.00 + (7.31)	97.20 = (3.29)	88.00 + (8.00)	**98.40** (2.07)
IRIS	84.00 = (7.83)	77.33 = (8.92)	84.00 = (7.83)	**84.67** = (6.32)	**84.67** (6.32)
ION	**76.93** = (8.22)	74.03 + (9.00)	**76.93** = (8.22)	76.06 = (8.22)	**76.93** (8.22)
ACA	78.99 = (7.52)	78.84 = (7.01)	79.42 = (7.13)	**80.14** = (7.04)	78.26 (7.64)
SPA	88.13 = (1.59)	**88.89** = (1.67)	88.24 = (1.62)	87.85 = (1.73)	87.57 (1.27)
VEH	65.25 = (5.04)	61.70 = (5.13)	65.32 = (5.14)	65.02 = (6.52)	**65.77** (5.49)
MAM	83.67 = (4.94)	82.53 = (5.35)	**83.88** = (5.06)	81.49 = (5.29)	80.76 (4.66)
TTE	**52.60** = (3.63)	**52.60** = (3.63)	**52.60** = (3.63)	**52.60** = (3.63)	**52.60** (3.63)
SEG	89.31 = (1.85)	89.56 = (1.92)	89.35 = (1.76)	90.17 = (1.50)	**90.48** (1.49)
AUD	93.43 = (1.86)	92.78 = (2.13)	93.43 = (1.86)	92.27 = (2.19)	**93.82** (1.69)
Average MAR	80.90	78.94	**81.00**	79.87	80.73
Wine/lose/tie	0/0/12	2/0/10	0/0/12	1/0/11	

Results shown in Tables 3-5 demonstrate that STOPF, in general, can train a better classifier of 3NN, SVM and Cart, compared with contrasted algorithms. By taking a closer look, our proposed algorithm achieves the highest MAR for 8 of 12 data sets in Table 3 and also achieves the highest MAR for 9 of 12 data sets in Table 4. Moreover, the average MAR of STOPF on all data sets is also the highest in Tables 3-4. Although the average MAR of STOPF on all data sets is not the highest in Table 5, STOPF achieves the highest MAR for 7 of 12 data sets in Table 5. All these prove that STOPF has some superiority.

In addition, the Wilcoxon test in Tables 3-5 shows that STOPF statistical outperforms other algorithms on some data sets. It's another clear evidence that the STOPF works better than others, when the trained classifier is 3NN or SVM or Cart. Furthermore, we have found that TT, STDP and STSFCM, in some cases, have the worse MAR than the supervised classifier 3NN or SVM or Cart. We think that some unlabeled samples are marked incorrectly during the iterative self-labeling process. As a result, a trained classifier (3NN or SVM or Cart) will be deteriorated. On the contrary, this situation less occurs in our algorithm. We believe this reason is that our proposed algorithm can use the structure of feature space revealed by OPF to improve self-training method.

C. DISCUSSING THE EFFECT OF RATIO OF LABELED SAMPLES

In this section, we will further discuss the percentage of labeled samples in our experiments, and compare some representative algorithms listed in Table 2 on 6 data sets, in which base classifiers are 3NN, SVM and Cart. In experiments, we increase the percentage of labeled samples from 10% to 90%. Similarly, we use a 10-fold cross-validation to evaluate our results. Figs. 4-6 shows the mean accuracy rate (MAR) of different algorithms with respect to different percentages of labeled samples on 6 data sets (WEP, LON, QUB, IRIS, VEH and TTE).

As shown in Figs. 4-6, MAR of 5 algorithms increases with the increase of percentage of labeled data. Furthermore, we can also conclude that STOPF can generally train better classifiers including 3NN, SVM and Cart than other algorithms. In detail, we have found from Figs. 4-6 that when the percentage of labeled samples is relatively low, our algorithm works better than others in most cases. In addition, when the percentage of labeled samples is relatively high, our algorithm achieves similar results with contrasted algorithms. The reason may be that when the proportion of labeled samples is relatively low, the real structure of feature space discovered by OPF can help our algorithm work better than contrasted algorithms. As the proportion of labeled samples increased,

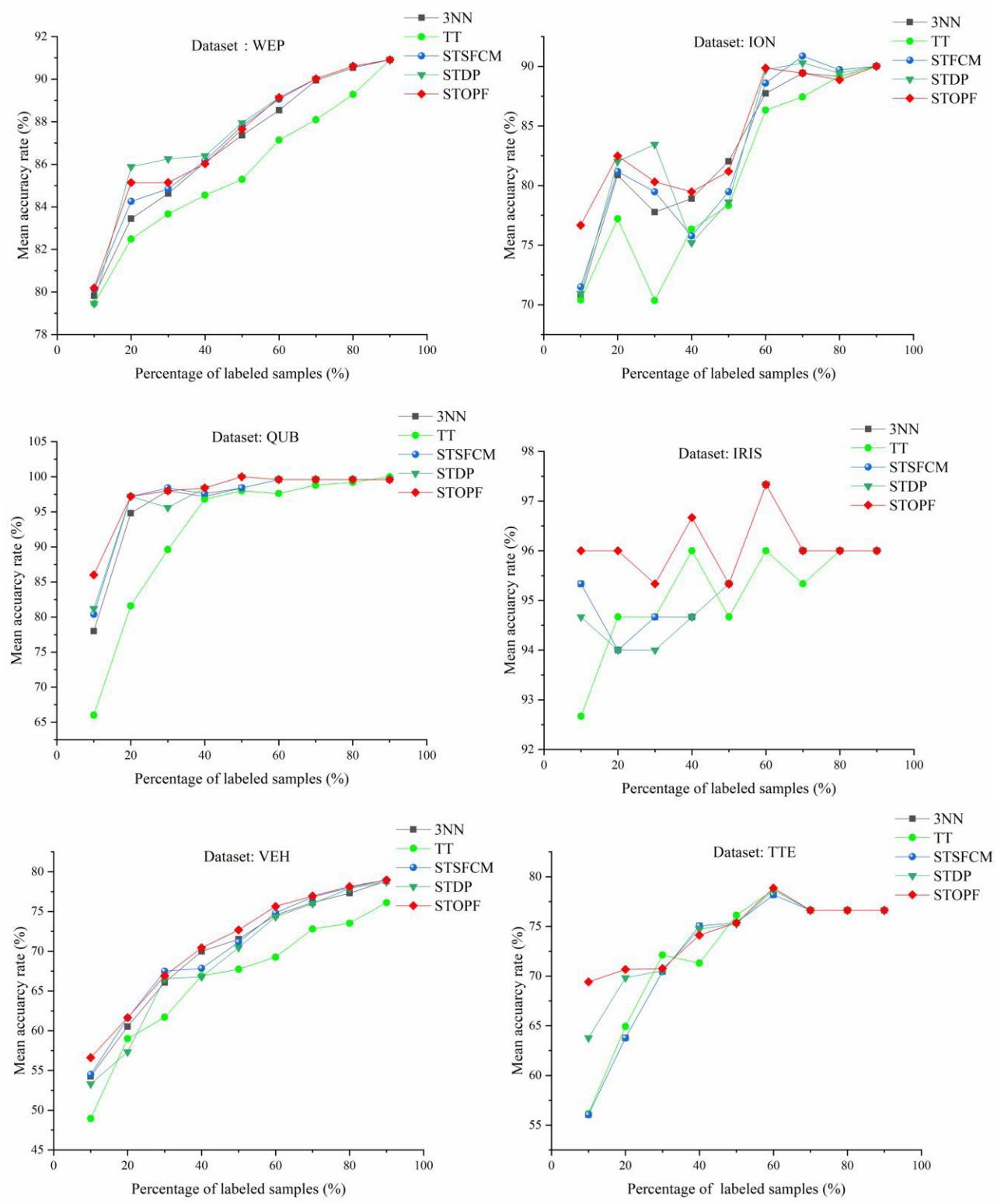

FIGURE 4. Contrasted algorithms with a base classifier 3NN with respect to different percentages of labeled samples on 6 data sets.

labeled samples can gradually represent the distribution of the entire data set. So, advantages of the structure revealed by OPF in our algorithm gradually narrow with the increase of labeled samples.

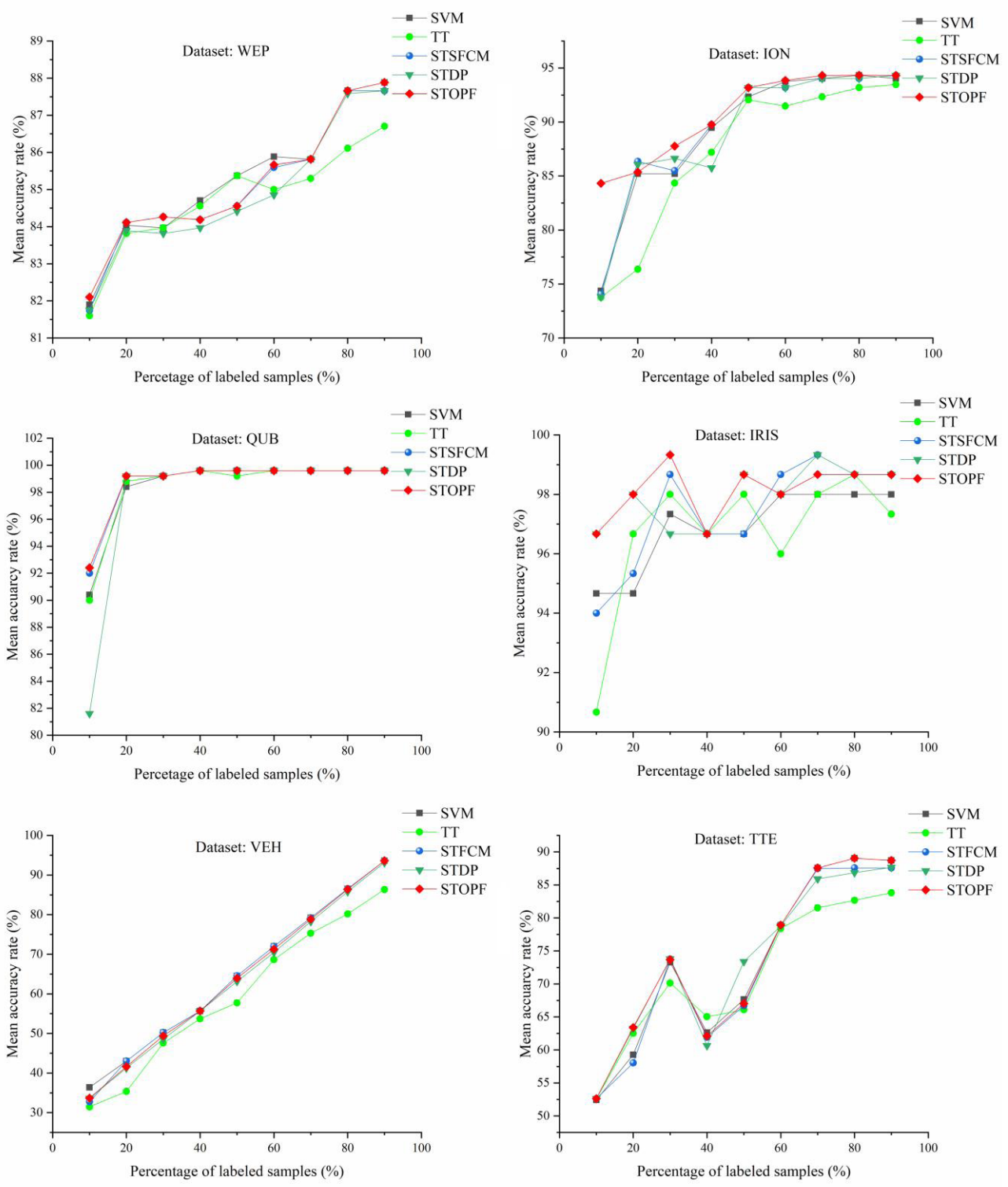

FIGURE 5. Contrasted algorithms with a base classifier SVM with respect to different percentages of labeled samples on 6 data sets.

D. COMPUTATIONAL EFFCIENCY

We also illustrate the computational efficiency of STOPF by comparing running time of the central processing unit (CPU) among all self-labeled algorithms on 6 data sets. Note that each algorithm is only executed once, where initial labeled ratio is 10%. Likewise, settings of parameters we use in this experiment are set as Table 2. The corresponding results are depicted in Fig. 7. Not surprisingly, STOPF consumes the

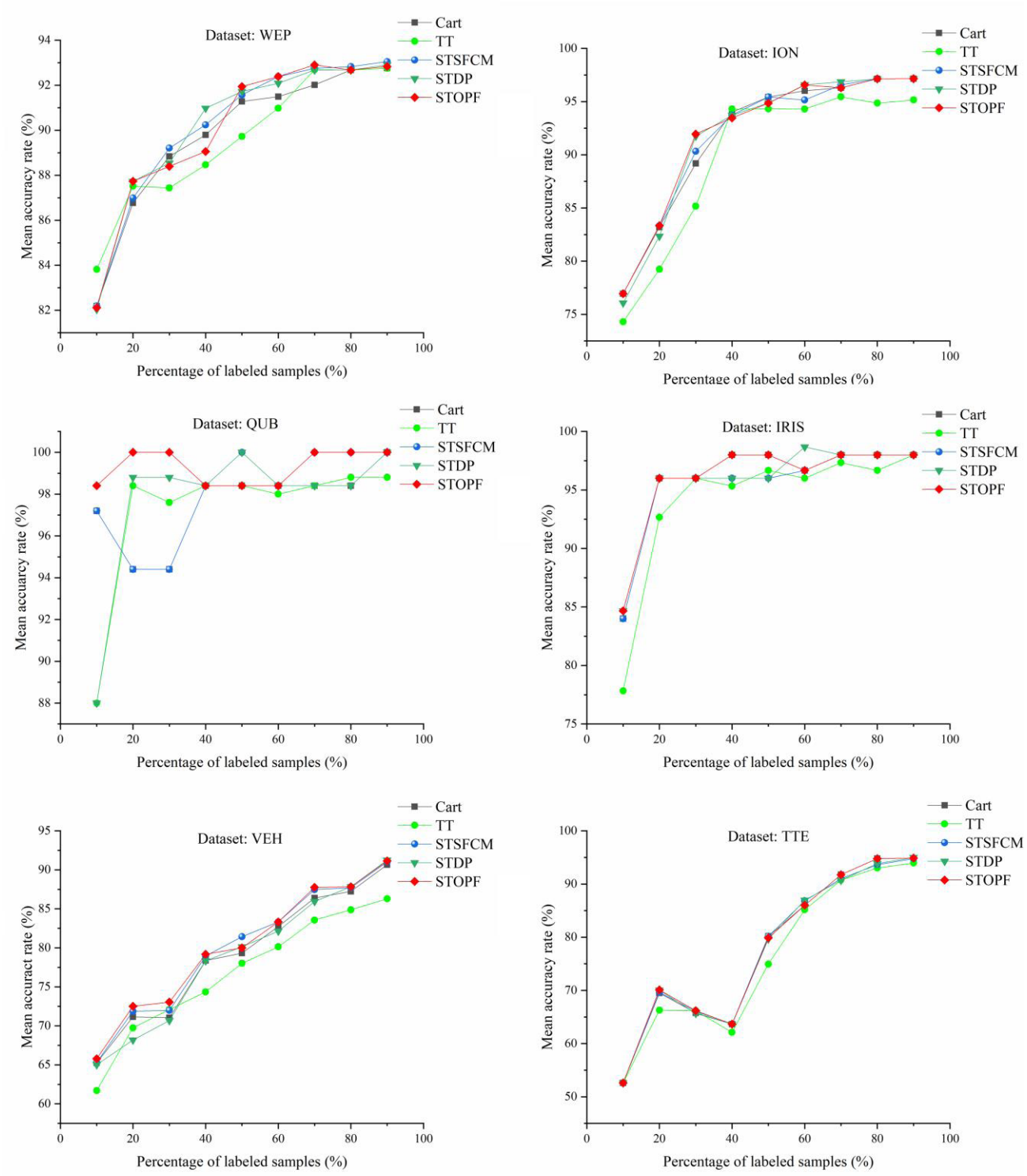

FIGURE 6. Contrasted algorithms with a base classifier Cart with respect to different percentages of labeled samples on 6 data sets.

most running time. The reason for this is that constructing OPF on the entire data set X is a relatively time-consuming and complex process. In detail, we have found that when building OPF, we need to consider all samples at each iteration. In fact, some relatively distant samples should not be considered, which increases the consumption of time. Therefore, we plan to overcome this shortcoming in the future. But STOPF has the advantage of improving

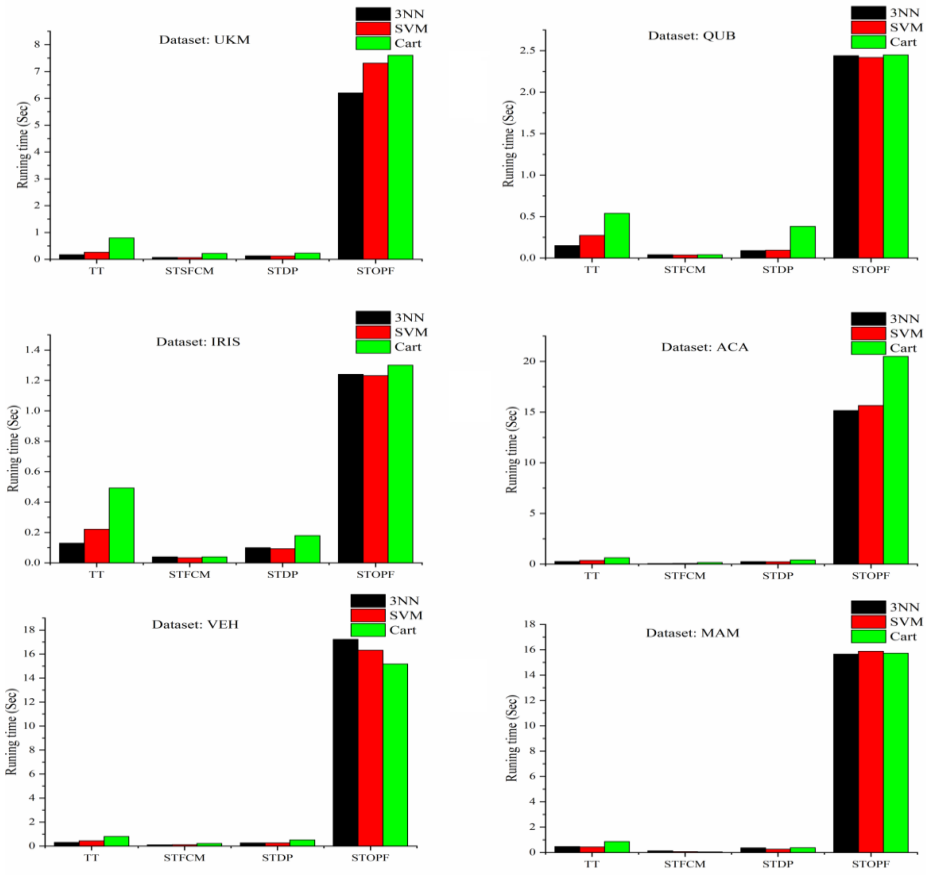

FIGURE 7. Computational efficiency of contrasted algorithms on 6 data sets.

classification accuracy of the trained classifier. Weighing the pros and cons of STOPF, the computational efficiency of STOPF is acceptable.

V. CONCLUSIONS

In this paper, we introduce a new self-training method based an optimum path forest containing three main parts. First of all, we propose to construct an optimum path forest to discover the potential spatial structure of feature space. Secondly, we use the structure to guide self-training method to iteratively mark unlabeled samples. Then, these samples are used to expand the labeled data. Thirdly, a desirable classifier can be trained with extended labeled data. In order to check the performance of our algorithm, we run some experiments, where 3 self-taught algorithms, 12 UCI data sets and 3 base classifiers including 3NN, SVM and Cart are adopted. The experimental results clearly demonstrate that our algorithm is more effective than previous works in improving the classification accuracy of base classifiers of 3NN, SVM and Cart. In our experiments, the effect of the proportion of labeled samples is also discussed. We have found that when the percentage of labeled samples is relatively low, our algorithm usually works better than others. From experiments of computational efficiency, we have also found that our algorithm is relatively time-consuming. The reason is that all samples have to be considered at each iteration, when constructing an OPF. In fact, some relatively distant samples should not be considered, which wastes time. In the future, we intend to solve this problem by adopting the technique of natural neighbors [36], [37].

In addition, we also find some characteristics of our proposed algorithm. First of all, our algorithm can train a good classifier without considering shapes and distribution of data sets. Especially, our algorithm can also train a more efficient classifier on data sets with some variations in density than previous algorithms. Secondly, in addition to parameters of the base classifier, our algorithm doesn't consider the selection of any parameters. Thirdly, our algorithm does not require the measurement of confidence. Fourthly, our proposed algorithm has a better performance than previous works on overlapping data sets to some extent, as long as initial labeled samples can protect their respective classes. But it is difficult to find such labeled samples by random selection. Therefore, we intend to overcome this shortcoming by using some technologies of border detection [38] and active learning [39] in future works.

REFERENCES

[1] I. Triguero, S. García, and F. Herrera, "Self-labeled techniques for semi-supervised learning: Taxonomy, software and empirical study," *Knowl. Inf. Syst.*, vol. 42, no. 2, pp. 245–284, 2015.

[2] H. Gan, Z. Li, W. Wu, Z. Luo, and R. Huang, "Safety-aware graph-based semi-supervised learning," *Expert Syst. Appl.*, vol. 107, no. 1, pp. 243–254, Oct. 2018.
[3] O. Kilinc and I. Uysal, "Gar: An efficient and scalable graph-based activity regularization for semi-supervised learning," *Neurocomputing*, vol. 296, pp. 46–54, Jun. 2018.
[4] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using EM," *Mach. Learn.*, vol. 39, nos. 2–3, pp. 103–134, 2000.
[5] Y. Li, Y. Wang, C. Bi, and X. Jaing, "Revisiting transductive support vector machines with margin distribution embedding," *Knowl.-Based Syst.*, vol. 152, no. 15, pp. 200–214, Jul. 2018.
[6] S. Wang, L. Wu, L. Jiao, and H. Liu, "Improve the performance of co-training by committee with refinement of class probability estimations," *Neurocomputing*, vol. 136, pp. 30–40, Jul. 2014.
[7] Y. Hong and W. Zhu, "Spatial co-training for semi-supervised image classification," *Pattern Recognit. Lett.*, vol. 63, pp. 59–65, Oct. 2015.
[8] I. Triguero, J. A. Sáez, J. Luengo, S. García, and F. Herrera, "On the characterization of noise filters for self-training semi-supervised in nearest neighbor classification," *Neurocomputing*, vol. 132, pp. 30–41, May 2014.
[9] L. Jurica, M. Ceci, D. Kocev, and D. Sašo, "Self-training for multi-target regression with tree ensembles," *Knowl.-Based Syst.*, vol. 123, pp. 41–60, May 2017.
[10] M. S. Hajmohammadi and R. Ibrahim, "Combination of active learning and self-training for cross-lingual sentiment classification with density analysis of unlabelled samples," *Inf. Sci.*, vol. 317, pp. 67–77, Oct. 2015.
[11] L. Shi, X. Ma, L. Xi, Q. Duan, and J. Zhao, "Rough set and ensemble learning based semi-supervised algorithm for text classification," *Expert Syst. Appl.*, vol. 38, no. 5, pp. 6300–6306, May 2011.
[12] T. H. N. Le, K. Luu, C. Zhu, and M. Savvides, "Semi self-training beard/moustache detection and segmentation simultaneously," *Image Vis. Comput.*, vol. 58, pp. 214–223, Feb. 2017.
[13] D. T. Vo and E. Bagheri, "Self-training on refined clause patterns for relation extraction," *Inf. Process. Manage.*, vol. 54, no. 4, pp. 686–706, Jul. 2017.
[14] D. Dalva, U. Guz, and H. Gurkan, "Effective semi-supervised learning strategies for automatic sentence segmentation," *Pattern Recognit. Lett.*, vol. 105, no. 1, pp. 76–86, Apr. 2018.
[15] Z. Wei, H. Wang, and R. Zhao, "Semi-supervised multi-label image classification based on nearest neighbor editing," *Neurocomputing*, vol. 119, pp. 462–468, Nov. 2013.
[16] M. M. Adankon and M. Cheriet, "Help-Training for semi-supervised support vector machines," *Pattern Recognit.*, vol. 44, no. 9, pp. 2220–2230, Sep. 2011.
[17] S. Karlos, N. Fazakis, A. P. Panagopoulou, S. Kotsiantis, and K. Sgarbas, "Locally application of naive Bayes for self-training," *Evolving Syst.*, vol. 8, no. 1, pp. 1–16, Mar. 2016.
[18] D. Wu, M. S. Shang, X. Luo, J. Xu, H. Y. Yan, W. H. Deng, "Self-training semi-supervised classification based on density peaks of data," *Neurocomputing*, vol. 275, pp. 180–191, Jan. 2017.
[19] J. Tanha, M. van Someren, and H. Afsarmanesh, "Semi-supervised self-training for decision tree classifiers," *Int. J. Mach. Learn. Cybern.*, vol. 8, no. 1, pp. 355–370, Feb. 2015.
[20] S. Gu and Y. Jin, "Multi-train: A semi-supervised heterogeneous ensemble classifier," *Neurocomputing*, vol. 249, pp. 202–211, Aug. 2017.
[21] J. J. Liu, S. Zhao, and G. Wang, "SSEL-ADE: A semi-supervised ensemble learning framework for extracting adverse drug events from social media," *Artif. Intell. Med.*, vol. 84, pp. 34–49, Jan. 2017.
[22] M. Li and Z. H. Zhou, "SETRED: Self-training with editing," in *Advances in Knowledge Discovery and Data Mining*, May 2005, pp. 611–621.
[23] Y. Wang, X. Xu, H. Zhao, and Z. Hua, "Semi-supervised learning based on nearest neighbor rule and cut edges," *Knowl.-Based Syst.*, vol. 23, no. 6, pp. 547–554, Aug. 2000.
[24] H. Gan, N. Sang, R. Huang, X. Tong, and Z. Dan, "Using clustering analysis to improve semi-supervised classification," *Neurocomputing*, vol. 101, no. 3, pp. 290–298, Feb. 2013.
[25] M. Fatehi and H. H. Asadi, "Application of semi-supervised fuzzy c-means method in clustering multivariate geochemical data, a case study from the Dalli Cu-Au porphyry deposit in central Iran," *Ore Geology Rev.*, vol. 81, pp. 245–255, Mar. 2017.
[26] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, Jun. 2014.
[27] D. Cheng, Q. Zhu, J. Huang, L. Yang, and Q. Wu, "Natural neighbor-based clustering algorithm with local representatives," *Knowl.-Based Syst.*, vol. 123, pp. 238–253, May 2017.
[28] A. S. Seyedia, A. Lrahman, P. Moradi, and N. N. Qader, "Dynamic graph-based label propagation for density peaks clustering," *Expert Syst. Appl.*, vol. 115, pp. 314–328, Jan. 2019.
[29] W. P. Amorim, A. X. Falcão, J. P. Papa, and M. H. Carvalho, "Improving semi-supervised learning through optimum connectivity," *Pattern Recognit.*, vol. 60, pp. 72–85, Dec. 2016.
[30] W. P. Amorim, A. X. Falcão, and J. P. Papa, "Multi-label semi-supervised classification through optimum-path forest," *Inf. Sci.*, vol. 465, pp. 86–104, Oct. 2018.
[31] S. Chen, T. Sun, F. Yang, H. Sun, and Y. Guan, "An improved optimum-path forest clustering algorithm for remote sensing image segmentation," *Comput. Geosci.*, vol. 112, pp. 38–46, Mar. 2018.
[32] A. S. Iwashita *et al.*, "A path- and label-cost propagation approach to speedup the training of the optimum-path forest classifier," *Pattern Recognit. Lett.*, vol. 40, pp. 121–127, Apr. 2014.
[33] J. P. Papa, A. X. Falcão, C. T. N. Suzuki, and N. D. A. Mascarenhas, "A discrete approach for supervised pattern recognition," in *Proc. Int. Workshop Combinat. Image Anal.*, Apr. 2008, pp. 136–147.
[34] R. Liu, H. Wang, and X. Yu, "Shared-nearest-neighbor-based clustering by fast search and find of density peaks," *Inf. Sci.*, vol. 450, pp. 200–226, Jun. 2018.
[35] Z.-H. Zhou and M. Li, "Tri-training: Exploiting unlabeled data using three classifiers," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 11, pp. 1529–1541, Nov. 2005.
[36] Q. Zhu, J. Feng, and J. Huang, "Natural neighbor: A self-adaptive neighborhood method without parameter K," *Pattern Recognit. Lett.*, vol. 80, pp. 30–36, Sep. 2016.
[37] L. Yang, Q. Zhu, J. Huang, D. Cheng, Q. Wu, and X. Hong, "Natural neighborhood graph-based instance reduction algorithm without parameters," *Appl. Soft Comput.*, vol. 70, pp. 279–287, Sep. 2018.
[38] Y. Wang, W. X. Peng, C. H. Qiu, J. Jun, and S. R. Xia, "Fractional-order Darwinian PSO-based feature selection for media-adventitia border detection in intravascular ultrasound images," *Ultrasonics*, vol. 92, pp. 1–7, Feb. 2018.
[39] P. T. M. Saito *et al.*, "Active semi-supervised learning using optimum-path forest," in *Proc. 22nd Int. Conf. Pattern Recognit.*, Aug. 2014, pp. 3798–3803.

JUNNAN LI received the bachelor's degree in computer science from Chongqing Normal University, in 2015. He is currently pursuing the Ph.D. degree with the Computer Science College, Chongqing University. His research interests are semi-supervised learning and data mining.

QINGSHENG ZHU received the B.S., M.S., and Ph.D. degrees in computer science from Chongqing University, in 1983, 1986, and 1990, respectively, where he is currently a Professor with the College of Computer Science, and also the Director of the Chongqing Key Laboratory of Software Theory and Technology. His main research interests include Ecommerce, data mining, and service oriented computing.