IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON REAL-TIME EDGE ANALYTICS FOR BIG DATA IN INTERNET OF THINGS

Received January 8, 2018, accepted February 13, 2018, date of publication February 19, 2018, date of current version March 16, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2807779

Improving Label Noise Filtering by Exploiting Unlabeled Data

DONGHAI GUAN [1,2], HONGQIANG WEI[1], WEIWEI YUAN[1,2], GUANGJIE HAN [3], (Member, IEEE), YUAN TIAN[4], MOHAMMED AL-DHELAAN[4], AND ABDULLAH AL-DHELAAN[4]

[1]College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics , Nanjing 211106, China
[2]Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing 210032, China
[3]Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software, Dalian University of Technology, Dalian 116023, China
[4]Department of Computer Science, King Saud University, Riyadh 11362, Saudi Arabia

Corresponding author: Guangjie Han (hanguangjie@gmail.com)

This work was supported in part by the Natural Science Foundation of China under Grant 61672284, in part by the Natural Science Foundation of Jiangsu Province under Grant BK20171418,in part by the China Postdoctoral Science Foundation under Grant 2016M591841, in part by the Fundamental Research Funds for the Central Universities, No. DUT17RC(3)094, in part by the Open Project Foundation of Information Technology Research Base of Civil Aviation Administration of China under Grant CAAC-ITRB-201501 and Grant CAAC-ITRB-201602, and in part by the Deanship of Scientific Research at King Saud University through the Research Group under Grant RGP-264.

ABSTRACT With the significant growth in the scale of data, an increasing amount of training data is available in many machine learning tasks. However, it is difficult to ensure perfect labeling with a large volume of training data. Some labels can be incorrect, resulting in label noise, which could lead to deterioration in learning performance. A common way to address label noise is to apply noise filtering techniques to identify and remove noise prior to learning. Multiple noise filtering approaches have been proposed. However, almost all existing works focus on only mislabeled training data and ignore the existence of unlabeled data. In fact, unlabeled data are common in many applications, and their values have been extensively studied and recognized. Therefore, in this paper, we explore the effective use of unlabeled data to improve the noise filtering performance. To this end, we propose a novel noise filtering algorithm called enhanced soft majority voting by exploiting unlabeled data (ESMVU), which is an ensemble-learning-based filter that adopts a soft majority voting strategy. ESMVU provides a systematic way to measure the value of unlabeled data by considering different aspects, such as label confidence and the sample distribution. Finally, the effectiveness of the proposed method is confirmed by experiments and comparison with other methods.

INDEX TERMS Label noise, noise filtering, unlabeled data, soft majority voting.

I. INTRODUCTION

Learning algorithms and training data are the two main factors when constructing a generalization model. In general, a good model requires a suitable learning algorithm and a sufficient amount of training data. Traditionally, the scarcity of training data is the main obstacle to obtaining a good model. However, in the era of big data, the amount of training data is usually sufficient. However, the substantial increase in the amount of training data inevitably results in errors and noise in many training datasets, which may seriously degrade the quality of the generalization model. Thus, noise handling is an important research topic in big data analysis.

The noise contained in the training dataset can be divided into two main categories: attribute noise and label noise [1]. Attribute noise is defined as an imprecision or a mistake introduced in the attribute values, while label noise is caused by mislabeling. The two types of noise have been comprehensively studied in many works [2], which have suggested that removing attribute noise can decrease the predictive accuracy of a classifier if the same attribute noise is present when the classifier is subsequently used. However, eliminating label noise consistently improves the predictive accuracy. This work focuses on label noise only, and ''noise'' in this work uniquely refers to label noise.

In [3]–[8], label noise handing approaches mainly consist of two types: algorithm-level approaches and data-level approaches. The former adapts existing algorithms to properly handle noise or to be less influenced by noise, whereas

the latter mainly preprocesses the dataset to remove noisy instances prior to learning. Algorithm-level approaches are not always available because they depend on the particular adaptation of a classification algorithm and cannot be directly applied with other algorithms. By contrast, the data-level approach is independent of the classification algorithm; thus, in many applications with noise, the data-level approach is preferred. Noise filtering is a commonly used data-level approach. Removing noise from the training dataset produces several benefits, such as improved classification accuracy and reduced model complexity.

In most cases, detecting label noise is challenging, and many noise filtering methods have been proposed with different degrees of success. Most works favor k-nearest neighbor (kNN) and ensemble learning algorithms [9]–[17]. kNN algorithms compare each sample's label with those of its surrounding neighbors. A sample is identified as mislabeled if most of its neighbors have different labels from that of the sample. Ensemble learning algorithms generally employ multiple classifiers to guarantee a more reliable filtering of mislabeled samples. The diversity among multiple classifiers can be generated either by using different learning algorithms or by selecting different training samples.

Ensemble learning filters differ in terms of classifier construction and/or decision fusion. However, most works focus on only the mislabeled training dataset and ignore the effect of unlabeled data. As semi-supervised learning shows, compared to labeled data, unlabeled data are usually much easier to obtain and are common in many applications. The success of the existing semi-supervised learning methods inspires us to develop a new noise filter that can use unlabeled data to improve the noise filter performance on a mislabeled training dataset.

To this end, we propose a novel method, named enhanced soft majority voting by exploiting unlabeled data (ESMVU). ESMVU is an ensemble-learning-based filter. In contrast to most works that use single voting, ESMVU utilizes the concept of multiple voting to make voting more reliable. In addition, a soft majority voting concept, which outputs both a voting result and a corresponding degree of trust, is proposed. ESMVU provides a systematic way to measure the reliability of estimated labels so that unlabeled data can be used to improve noise filtering from a labeled noisy training dataset. To effectively utilize the information of unlabeled data, ESMVU also utilizes the sample distribution information for noise filtering. For this purpose, an LDC-kNN based editing procedure is included in ESMVU. This procedure is expected to be superior to traditional kNN editing by utilizing both distance and distribution information. Finally, the performance improvement of ESMVU is demonstrated with extensive experiments on KEEL datasets and UCI datasets and by comparison with the state-of-the-art approaches.

Therefore, ESMVU is proposed to overcome the limitation of existing methods whose label noise detection performance is highly dependent on the labeled training dataset. It should be noted that using unlabeled data is not free and sometimes inappropriate usage could further degrade the learning performance. The main contributions of ESMVU involve the reliable using unlabeled data by applying soft majority voting and data editing techniques.

The rest of this paper is arranged as follows. Section 2 presents related works on classification with noisy data. Section 3 introduces the details of the proposed method for mislabeled data filtering. In Section 4, we describe the experimental framework and analyze the experimental results. Finally, we present concluding remarks and briefly discuss the direction of future work in Section 5.

II. RELATED WORK

When the labels of training instances are polluted, the learning performance is degraded from different aspects, including classifier complexity and classification accuracy [18]–[29]. Unfortunately, mislabeled training data exist in many applications, such as wireless sensor network analysis [30]–[32] and big data analysis [33]. The complexity of a classifier mainly involves the size and interpretability of the classifier. In [3], it was shown that the size of decision trees can be increased by mislabeled data. The existing research also found that the complexity of SVM also increases when mislabeled data exist. Among all the negative aspects of mislabeled data, the most frequently reported consequence is the decreased classification accuracy. For example, the behavior of the nearest-neighbor classifier has been investigated when mislabeled training samples exist [22].

Mislabeling in practical datasets is common; therefore, techniques that eliminate or reduce its impact are needed. There are two main approaches to handle mislabeling. The first is the algorithm-level approach, which relies on algorithms that are naturally robust to label noise because studies have shown that some algorithms are less influenced than others by label noise. These algorithms directly model label noise during learning or are modified to consider label noise in an embedded fashion. C4.5 is an example of a robust learner that uses pruning strategies to reduce the chance of overfitting due to noise in the training data. There are some disadvantages to algorithm-level approaches. Because not all algorithms are robust to noise, this approach lacks versatility. Moreover, although some algorithms are robust to noise, label noise is not addressed in this approach.

The second approach to handling mislabeling is the data-level approach. The best-known methods of this type are mislabeled data filters, which identify mislabeled instances that can be eliminated from the training data prior to training. The separation of noise filtering and the learning phase has the advantage of avoiding the use of polluted instances in the classifier building process. In addition, filter approaches are cheap and easy to implement. Our work is included in this category.

Mislabeled filtering approaches include k-nearest neighbor, ensemble learning, and application oriented. Application-oriented approaches have been proposed to solve concrete problems; thus, they lack generality and are not

Algorithm 1 Majority Filter (MF)

Input: E (training set)
Parameters: n (number of subsets), y (number of learning algorithms), $A_1, A_2, \ldots, A_y$ (y learning algorithms)
Output: A (detected noisy subset of E)

1: form n disjoint almost equally sized subsets of E_i, where $\bigcup_i E_i = E$
2: $A \leftarrow \emptyset$
3: **for** $i = 1 \ldots n$ **do**
4: form $E_t \leftarrow E \backslash E_i$
5: **for** $j = 1, \ldots y$ **do**
6: induce H_j based on examples in E_t and A_j
7: **end for**
8: **for** every $e \in E_i$ **do**
9: $ErrorCounter \leftarrow 0$
10: **for** j=1,...,y **do**
11: **if** H_j incorrectly classifies e **then**
12: $ErrorCounter \leftarrow ErrorCounter + 1$
13: **end if**
14: **end for**
15: **if** $ErrorCounter > \frac{y}{2}$, **then**
16: $A \leftarrow A \cup \{e\}$
17: **end if**
18: **end for**
19: **end for**

considered here. K-nearest neighbor approaches assume that nearby samples should have the same labels; however, this assumption sometimes does not hold.

By contrast, ensemble learning filters are the most widely used. This type of filter is preferred because an ensemble classifier has better performance than each base-level classifier on a dataset if two conditions hold: (1) the probability of a correct classification by each individual classifier is greater than 0.5 and (2) the errors in the predictions of the base-level classifiers are independent. In this work, a novel ensemble learning based filter is proposed. As necessary background knowledge, conventional ensemble-learning-based filters are introduced in this section.

Ensemble-learning-based filters employ ensemble classifiers to detect mislabeled instances by constructing a set of base-level classifiers and using their classifications to identify mislabeled instances. The general approach is to tag an instance as mislabeled if x of the m base-level classifiers cannot classify it correctly. Majority filter (MF) and consensus filter (CF) are representative algorithms [3]. MF tags an instance as mislabeled if more than half of the m base-level classifiers classify it incorrectly. CF requires that all base-level classifiers disagree with the given class label for an instance to be eliminated from the training data.

The algorithm of the majority filter is shown in Algorithm 1. MF begins with n equal-sized disjoint subsets of the training set E (step 1) and the empty output set A of detected mislabeled examples (step 2). The main loop (steps 3–6) is repeated for each training subset E_i. In step 4, subset E_t, which includes all examples from E except those in E_i, is formed and is then used as the input of an arbitrary inductive learning algorithm that induces a hypothesis (a classifier) H_j (step 6). The examples from E_i for which a majority of the hypotheses does not give the correct classification are added to A as potentially noisy examples (step 14). CF is similar to MF; the only difference is in step 14. In CF, the example in E_i is regarded as mislabeled only when all the hypotheses incorrectly classify it. Compared with MF, CF is more conservative due to the stronger condition for noise identification, which results in fewer instances being eliminated from the training set. Consequently, CF has greater risk of retaining mislabeled data.

TABLE 1. Datasets used in the experiment.

Dataset	#EX	#AT
Nursery	12960	7
Heart	270	13
Balance-scale	625	4
Spambase	4597	57
Titanic	2201	3
Mushroom	5644	22
Splice	3190	60
Australian	690	14
Haberman	306	3
Twonorm	7400	20

Existing filters, such as MF and CF, retain or discard a training sample based on various noise-detection policies. Most of the existing approaches are supervised and use partial training samples for training before identifying noise in the remaining samples. Supervised noise filters may face difficulty if the amount of training data is limited because insufficient noisy training samples make is difficult to construct reliable classifiers for noise detection. In fact, in addition to labeled data, many unlabeled data are available in real applications. Additionally, unlabeled data are usually sufficient since they do not require labeling. For example, in webpage classification tasks, the number of unlabeled samples is almost countless. In this work, we consider the utilization of these ''cheap'' unlabeled data to improve the noise filtering performance to obtain a cleaner training dataset.

III. ESMVU: ENHANCED SOFT MAJORITY VOTING BY EXPLOITING UNLABELED DATA

This section is arranged as follows. In Section 3.1, we introduce the motivations and main concepts proposed in this paper. Then, the details of the proposed approach are presented in Section 3.2.

A. MAIN CONCEPTS PROPOSED IN THIS WORK

1) SOFT MULTIPLE MAJORITY VOTING

Our proposed ESMVU is an ensemble-learning-based noise filter based on the well-known majority filtering (MF) [3]. MF employs an ensemble of classifiers in which a sample is identified as mislabeled when it is classified incorrectly by more than half of the base-level classifiers.

MF consists of two main steps. The first step is data partitioning, which randomly divides the training data E into n equal-sized subsets $E_1, E_2, \ldots, E_n$. Cross-validation is used in the second step. Each E_i is held out, and the other n-1 subsets, $E_n \backslash E_i$, are used to train multiple classifiers, which are then used to detect the potentially mislabeled data in E_i. MF executes the above two steps only once, so it is a single voting noise filter.

Single voting is easily influenced by the data partitioning step. As training data are randomly assigned, the trained classifiers are different in different partitions. Consequently, the classifiers' detection results are not robust. The same instance can be identified with opposite results with two different partitions. Therefore, single voting is risky and unreliable.

A multiple voting noise filter has been proposed in [27] to reduce the effect of data partitioning. It is organized into two layers and consists of two single voting detectors. Each single voting detector generates one suspected mislabeled data index in the 1st layer voting; then, all the suspected indexes are combined in the 2nd layer to output the final summarized mislabeled data index. Multiple voting requires two voting mechanisms for the two layers. Accordingly, different variants of filters have been designed, including MF1, MFMF, and MFCF. Previous research shows that the performance of MFMF is good in most cases.

For MFMF, both layers are based on majority voting. Majority voting is effective in many cases; however, it is "rigid" in some sense. For instance, suppose there exist two samples, whose predicted labels set by MFMF are $\{+, +, -, +, -, +\}$ and $\{+, +, -, +, +, +\}$. Although the majority class for both samples is "+", the confidences of belonging to "+" are different. Based on the frequency of occurrence, the confidences of "+" are 4/6 and 5/6 for these two samples.

Soft multiple majority voting is proposed in this work to reflect the above confidence information. Because it contains both class information and confidence information, it is "soft". In this scheme, soft majority voting is applied in the first layer, and a predefined threshold is used in the second layer. The confidence of the class of each sample is compared with the predefined threshold, and an instance is identified as mislabeled when the confidence is lower than the predefined threshold. We use the following equation to compute the confidence value.

$$Confidence(x) = \frac{No.\ of\ \ classifiers\ correctly\ classify\ x}{No.\ of\ \ data\ partition\ *No.\ of\ \ classifiers} \tag{1}$$

2) LDC-kNN BASED DATA EDITING

Soft multiple majority voting reflects the confidence of class labels. The merits of using confidence are two-fold for ESMVU. First, it can select more reliable labeled data to construct ensemble classifiers. Second, it can select more reliable unlabeled data from the pool of the unlabeled dataset. Next we need a systematic way to fully utilize the value of these unlabeled samples. To this end, a hybrid approach is proposed to utilize different aspects of unlabeled samples. On one hand, the selected unlabeled data are put into the framework of the ensemble-learning-based filter. On the other hand, these unlabeled data are used to support data editing on the noisy training dataset. Because both the kNN-based filter and ensemble-learning-based filter might be biased, their combination is expected to maximize the value of the unlabeled data.

Several kNN-based data editing approaches exist, such as ENN, RENN, and All-kNN. These methods are based on the kNN algorithm, which considers the label information of a sample's neighbors and ignores the distribution information contained in the neighbors. In most cases, however, the importance of each neighbor is different, and a similarity measurement should be considered. A neighbor that is closer to a sample X is more important to X. Thus, distance information should be considered in data editing [34]. Our first choice is DW-kNN [35], which is a distance-weighted kNN. For a sample X, DW-kNN assigns the i^{th} nearest neighbors x_i a distance-based weight w_i as:

$$f(x) = \begin{cases} \dfrac{d_k - d_i}{d_k - d_1}, & d_k \neq d_1 \\ 1, & d_k = d_1. \end{cases} \tag{2}$$

In Equation 2, d_k and d_1 represent the maximum and minimum distances of the kNNs to the test instance x_i. The instance is assigned to the class in which the weights of the representatives of the kNNs sum to the greatest value. DW-kNN is distanced-weighted and superior to the original kNN, but it is unable to use distribution information. In fact, in many problems, the distribution information is important to improve classification performance. To this end, we use local-distribution-characteristics-based kNN (LDC-kNN) in our filtering framework [30]. LDC-kNN aims to quantify the membership degree based on fuzzy set theory [36].

LDC-kNN first finds the k-nearest neighbors of the query instance q. Then, the center $C(c_i)$ and the standard deviation are computed, where c_i is the cluster of the i^{th} class in the k-nearest neighbor. The computation is based on the following equations:

$$C(c_i) = \frac{1}{N_i} * \sum_{j=1}^{N_i} X_j^{c_i} \tag{3}$$

$$d(X_1 - X_2) = \sqrt{\|X_1 - X_2\|^2} = \sqrt{\sum_{i=1}^{d} (X_{1i} - X_{2i})^2} \tag{4}$$

$$\sigma(c_i) = \sqrt{\frac{1}{N_i} * \sum_{j=1}^{N_i} \left\| x_j^{c_i} - C(c_i)^2 \right\|}. \tag{5}$$

In these equations, N_i is the number of samples in class c_i around the query instance q. $X_j^{C_i}$ is the instance in class c_i with

$$X_j^{c_i} = (x_{j1}^{c_i}, x_{j2}^{c_i}, \ldots, x_{jd}^{c_i})$$

(d is the number of features in each instance). $d(X_1, X_2)$ represents the distance between X_1 and X_2.

The membership degree (MD) is based on the following equation. When $p = 0$ and $MD = N_i$, LDC-kNN is equivalent to kNN. When $p \to +\infty$ and $\frac{\sigma(c_i)}{d(q,C(c_i))} \neq 1$, N_i has little impact on MD; thus, it is important to choose an appropriate value for p.

$$MD_{c_i}(q) \propto N_i * (\frac{\sigma(c_i)}{d(q, C(c_i))})^p \quad (6)$$

The decision strategy of LDC-kNN is as follows. In most cases, only one class will be decided. Special cases are addressed in the following manner. (1) When $MD = 0$, we assign the query sample to the class with the maximum value of $N_i/(d(q, C(c_i)))^p$. (2) When $MD = \infty$, we assign the query instance to the class with the maximum value of $N_i * \sigma(c_i)^p$.

B. THE DETAILED ALGORITHM

The main concepts of our proposed approach have been presented in Section 3.1. We present our algorithm in detail in Section 3.2.

TABLE 2. Experimental results for the mushroom dataset.

Method	Test accuracy				
	0	0.1	0.2	0.3	0.4
3-NN					
None	**0.9973**	0.8936	0.7907	0.6976	0.5905
CF	0.9964	0.9229	0.8876	0.8137	0.7437
EF	0.9956	0.9211	0.8923	0.8182	0.7544
IPF	0.9938	**0.9385**	0.8963	0.8217	0.7660
ENN	0.9956	0.9081	0.8554	0.8077	0.7105
ESMVU	0.9925	0.9194	**0.8978**	**0.8283**	**0.7768**
Naive Bayes					
None	0.9273	0.8536	0.7807	0.6976	0.5905
CF	0.9264	0.8929	0.8476	0.8037	0.7467
EF	0.9256	0.8911	0.8423	0.8082	0.7556
IPF	0.9238	0.8985	0.8463	0.8107	0.7663
ENN	**0.9356**	0.8781	0.8054	0.7577	0.6805
ESMVU	0.9125	**0.8994**	**0.8578**	**0.8183**	**0.7738**
Decision Tree					
None	**1**	0.9156	0.8213	0.7319	0.6658
CF	0.9989	**0.9168**	0.8814	0.8026	0.7416
EF	0.9973	0.9158	0.8867	0.8079	0.7428
IPF	0.9973	**0.9168**	0.8867	0.8056	0.7418
ENN	**1**	0.9094	0.8429	0.7845	0.6953
ESMVU	0.9202	0.9148	**0.8895**	**0.8096**	**0.7637**

The whole process of our proposed approach, which can be divided into two layers, is shown in Algorithm 2. The first layer (steps 1-19) is the first filtering process, which provides training data for the learning algorithms in the second layer. In this layer, a relatively clean dataset, indReliable, is generated after filtering the original training dataset E. We propose the soft multiple majority voting method (MSMV), which is shown in Algorithm 3, to generate indReliable. In MSMV, we first need to select the number of data partitions (suppose the number is t). For every data partition, we divide the dataset into n disjoint almost equally sized subsets. Then, we take one of the n subsets as the test dataset and use the remaining subsets as the training dataset for y different learning algorithms to train y classifiers (steps 2-7 in Algorithm 3). Afterwards,

TABLE 3. Experimental results for the heart, spambase, and titanic datasets.

Method	Test accuracy				
	0	0.1	0.2	0.3	0.4
heart					
None	0.7778	0.6569	0.6143	0.5621	0.5213
CF	0.8333	0.7502	0.7351	0.6827	0.6209
EF	0.8333	0.7545	0.7386	0.6978	0.6242
IPF	**0.9444**	0.7614	0.7451	0.7386	0.6503
ENN	0.7778	0.7009	0.6579	0.6037	0.5882
ESMVU	0.8889	**0.7627**	**0.7484**	**0.7680**	**0.6764**
spambase					
None	0.8838	0.8096	0.7278	0.6332	0.5887
CF	**0.9034**	0.8771	0.8179	0.7819	0.7115
EF	0.9008	0.8786	0.8278	0.7990	0.7148
IPF	0.8975	**0.8823**	0.8382	0.8101	0.7395
ENN	0.8701	0.8214	0.7792	0.7143	0.6591
ESMVU	0.8923	0.8798	**0.8418**	**0.8464**	**0.8552**
titanic					
None	**0.8428**	0.7397	0.6846	0.5851	0.5217
CF	0.8103	0.7710	0.7543	0.7192	0.6986
EF	0.8179	0.7886	0.7666	0.7397	0.7123
IPF	0.8237	**0.7941**	**0.7693**	0.7430	0.7260
ENN	0.8053	0.7516	0.6879	0.6164	0.6041
ESMVU	0.8150	0.7872	0.7640	**0.7517**	**0.7737**

the classifiers are used to predict the labels for the samples in the test dataset. The entire process is conducted by cross-validation. In this way, each instance in every data partition has y predicted labels, so we obtain a total of $t * y$ predicted labels for every instance. These predicted labels include some labels that are the same as the sample's original label. The number of such labels is used to determine the confidence of the label for this sample based on Equation 1. Finally, the sample is regarded as reliable if the confidence is not less than the predefined threshold T (steps 19-20 in Algorithm 3).

MSMV is an ensemble-based filter that is similar to the multiple voting in [27]. However, it is expected to be superior because the confidence information of each instance is considered, which could improve the reliability of the filtering. As shown in Algorithm 2, the indReliable dataset is extracted and used to train several classifiers, which are then used to predict the labels for unlabeled data. The augmented dataset E' is a combination of unlabeled data and labeled data. Next, we take E' as the training dataset and E as the test dataset, apply the MSMV method, and generate the indReliable1 dataset. Afterwards, the unlabeled data and their predicted labels are regarded as the training data for LDC-kNN based editing to filter the dataset E to generate indReliable2. The first-layer filtering is complete at this step. The following process is conducted for the second-layer filtering (steps 25-35 in Algorithm 2). In this layer, indReliable, indReliable1, and indReliable2 are jointly used to train y different classifiers (steps 20-24 in Algorithm 2). Finally, the $3 * y$ classifiers are applied to E to obtain the final reliable dataset A, thus completing the data preprocessing. In ESMVU, we use ensemble learning in several places. Although this requires greater computational effort, it is acceptable because mislabeled

Algorithm 2 ESMVU: Enhanced Soft Majority Voting by Exploiting Unlabeled Data

Input: E (labeled data set), U (unlabeled data set)

Parameters: n (number of subjects), y (number of learning algorithms), $A_1, A_2, \ldots A_y$ (y kinds of learning algorithm), T_1, T_2, T_3 (three confidence thresholds), n (number of unlabeled data), k (number of nearest neighbors), m (number of classes)

Output: A (detected reliable subset of E)

indReliable $= MSMV(E, T_1)$
$A \leftarrow \emptyset$
for $j = 1, 2, \ldots, y$ **do**
  induce H_j based on instances in indReliable and algorithm A_j
end for
for every $t \in U$ **do**
  for $j = 1, 2, \ldots y$ **do**
    $pl_j(t) \leftarrow H_j(t)$
  end for
  if $pl_1(t) = pl_2(t) =, \ldots, = pl_y(t)$ **then**
    $T_U \leftarrow t \cup pl_1(t), U \leftarrow U \backslash t$
  end if
end for
$E' \leftarrow E \cup T_U$
indReliable1 $= MSMV(E', T_2)$
for every $t \in E$ **do**
  find the k nearest neighbors of instance t, use LDC-kNN to predict the label of instance t to be l'_t
  if $l'_t == l_t$ **then**
    indReliable2 $\leftarrow$indReliable2$\cup t$
  end if
end for
for $j = 1, 2, \ldots y$ **do**
  induce H_{j1} based on instances in indReliable and algorithm A_j
  induce H_{j2} based on instances in indReliable1 and algorithm A_j
  induce H_{j3} based on instances in indReliable2 and algorithm A_j
end for
for every $t \in E$ **do**
  for $i = 1 : 3$ **do**
    for $j = 1, 2, \ldots y$ **do**
      $pl_{ji}(t) \leftarrow H_{ji}(t)$
      if $pl_{ji}(t) == l_t$ **then**
        $sum = sum + 1$
      end if
    end for
  end for
  if $sum/(3 * y) >= T_3$ **then**
    $A \leftarrow A \cup t$
  end if
end for
return A (reliable data in E)

Algorithm 3 MSMV: Multiple Soft Majority Voting

Input: dataset E, T (threshold of filtering reliable data)

Parameters: n (number of subjects), y (number of learning algorithms), $A_1, A_2, \ldots A_y$ (y kinds of learning algorithm), t (number of times of data partitioning), n (number of subsets

Output: A (detected reliable subset of E)

for $p = 1, 2, \ldots t$ **do**
  form n disjoint almost equally sized subsets of E_{pi}, where $\cup_i E_{pi} = E$.
  for $i = 1, 2, \ldots n$ **do**
    form $E_t \leftarrow E \backslash E_{pi}$
    for $j = 1, 2, \ldots y$ **do**
      induce H_{pj} based on instances in E_t and algorithm A_j
    end for
    for every $e \in E_{pi}$ **do**
      for $j = 1, 2, \ldots y$ **do**
        $pl_j(e) \leftarrow H_j(e)$
        if $pl_j(e) == l_t$ **then**
          $trueCounter(e)_p \leftarrow trueCounter(e)_p + 1$
        end if
      end for
    end for
  end for
end for
for every $e \in E$ **do**
  $trueCounter(e) = \sum_{p=1}^{p=t} trueCounter(e)_p$
  if $trueCounter(e)/(t * y) >= T$ **then**
    $A \leftarrow A \cup e$
  end if
end for
return A (the reliable data in E

filtering is usually an off-line process, and some parts can be implemented by parallel processing.

IV. EXPERIMENT

This Section presents the details of the experimental study conducted to check the validity of the proposed method. Section 4.1 presents the experimental datasets and related noise filtering algorithms for comparison. The experimental configurations and results are presented in Section 4.2.

A. DATASETS AND RELATED ALGORITHMS USED IN THE EXPERIMENT

Ten datasets from the KEEL dataset and UCI repositories [37], [38] are used in the experiment. The detailed dataset information is shown in Table 1, where #*EX* and #*AT* represent the number of examples and number of attributes, respectively. The content of each dataset is as follows: Nursery database was developed to rank applications for nursery schools; Heart dataset was used to predict the absence or presence of heart disease; Balance-scale dataset was generated to

model psychological experimental results; Spambase dataset denotes whether the e-mail was considered spam or not; Titanic dataset records various attributes of passengers on the Titanic, including both survived and not; Mushroom dataset includes descriptions of hypothetical samples corresponding to different species of gilled mushrooms; Splice datset consists of 60 variables, representing a sequence of DNA bases, and an additional class variable. The task is to determine if the middle of the sequence is a splice junction and what is its type; Australian datset concerns credit card application and is used for Astralian credit approval; Haberman dataset records the status of patients (survived 5 years or longer, and the patient died within 5 year); Twonorm is a 2 class classification problem. Each class is drawn from a multivariate normal distribution. In these datasets, examples containing missing values are removed. To avoid the effect of class imbalance in multi-class datasets, we focus on the binary classification problem in this experiment. Thus, datasets that originally consist of more than two classes (Nursery, balance-scale, and splice) are converted into binary problems.

Each dataset in the experiment is randomly partitioned into two parts, labeled set (L) and unlabeled set (U), to explore the use of unlabeled data to improve label noise identification. Considering that the amount of unlabeled data is usually greater than the amount of labeled data, $1/3$ of instances in each dataset are selected as L. To test the performance of the noise filter, L is divided into two parts (30% for testing and 70% for training). The original datasets are noise-free; thus, we artificially introduce noise into the training set. Different noise ratios are tested, including 10%, 20%, 30%, and 40%. The average classification accuracy for each noise filter algorithm is obtained by averaging the classification accuracy from ten trials.

In the experiments, we select the following representative noise filtering algorithms for comparison.

1. Edited Nearest Neighbor (ENN) [8]. This algorithm removes instances whose class does not match the majority of its k-nearest neighbors.

2. Classification Filter (CF) [26]. CF divides the training dataset into n subsets. A set of classifiers is trained based on the union of any $n-1$ subsets. The examples misclassified in the remaining subset are then eliminated from the training dataset.

3. Ensemble Filter (EF) [3]. EF classifies the training dataset using n-fold cross-validation with several different classification algorithms. Then, noisy instances are identified using a voting mechanism (consensus or majority) and eliminated from the training dataset.

4. Iterative-Partitioning Filter (IPF) [25]. IPF removes noisy instances through multiple iterations. In each iteration, the training dataset is divided into n subsets, and C4.5 is built over each of these subsets to evaluate all the instances. Then, the misclassified instances are removed (using the consensus or majority voting scheme), and a new iteration is started.

B. EXPERIMENTAL RESULTS

In this section, the experimental results of all the filters, namely, CF, EF, IPF, ENN, and our proposed ESMVU, are presented and compared. EF and IPF are ensemble-learning-based filters, and ESMVU is a hybrid two-layer algorithm. In the first layer, ESMVU combines ensemble-based filtering (multiple soft majority voting) and LDC-kNN-based data editing to generate a relatively reliable training dataset, which is then used to construct the classifiers for the second layer. The main parameters of each algorithm are set as follows. For CF, the classifier is C4.5 and n is 5. For EF, majority voting is used and n is 3. The parameters of IPF are the same as those of EF. ENN uses the Euclidean distance and a k of 3. For ESMVU, the two thresholds in the first layer are 0.8 and 0.9. For LDC-kNN data editing, k is 10. The threshold of the second layer in ESMVU is set to $7/9$.

We first randomly select one dataset (mushroom) to study. Table 2 shows the classification accuracy of the different classification algorithms (3-NN, decision tree, naive Bayes) and noise filters. The best results at each noise level are highlighted in bold. This table provides several important observations:

(1) In most cases, the classification accuracy is highest when the noise ratio is 0 (training data are noise-free). As the noise ratio increases, the classification results worsen. Therefore, noise filtering is necessary, especially for datasets with high noise ratios.

(2) ESMVU shows superior performance in most cases. The advantage of ESMVU is especially clear when the noise ratio is greater than 0.2, which suggests that ESMVU is suitable for training data preprocessing when the noise ratio is relatively large.

(3) When IPF and EF are compared using decision tree, IPF is not better than EF when the noise ratio is greater than 0.2, indicating that iteration of the noise filtering process has little impact on the classification accuracy when the noise ratio is high.

(4) The classification results of different classification algorithms are also different, but the main experimental trends are consistent. Therefore, in the following experiments, only the classification results based on 3-NN are reported.

The results for the mushroom dataset are presented in Fig. 1, which shows that the superiority of ESMVU becomes more obvious as the noise ratio increases.

In the following, the detailed experimental results on the other nine datasets are presented. Due to space limitations, the results for three datasets are presented in each table, and 3-NN is selected as the classification algorithm. Table 3 includes the experimental results for the heart, spambase, and titanic datasets. For the heart dataset, ESMVU is always the best noise filtering method. This dataset contains only 270 instances, among which 90 are used as labeled data. The noisy training dataset is a subset of these 90 instances, which might be insufficient for training. In this case,

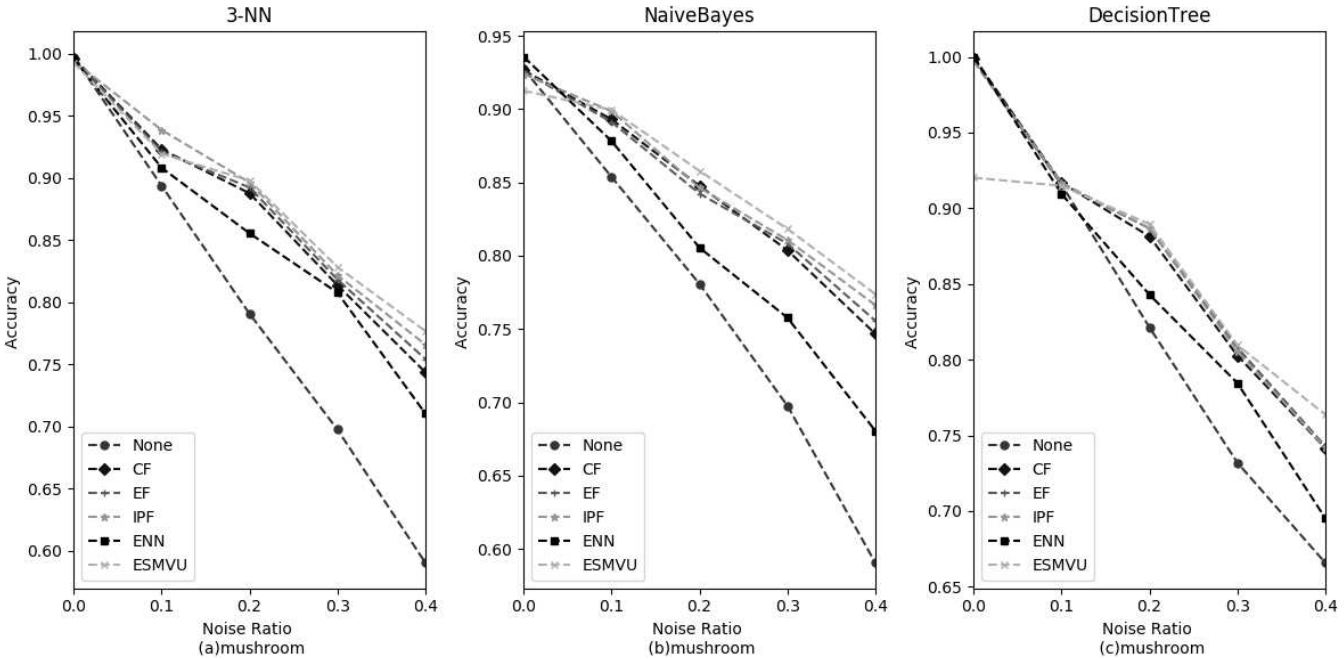


FIGURE 1. Classification comparison for the mushroom dataset.

unlabeled data are important to improve the classification accuracy. In addition, when only labeled data are used, IPF outperforms CF and EF. This observation reflects the value of iteration in noise filtering. For the spambase dataset, ESMVU is the best when the noise ratio is not less than 0.2. When the noise ratio increases from 0.3 to 0.4, the classification accuracy improves. The same observation is obtained from the titanic dataset. The additional information gained from unlabeled data may result in improved classification accuracy. IPF is the second best filter in most cases due to the iteration mechanism used in IPF. The experimental results for the titanic dataset are similar to those for the spambase dataset, demonstrating that the advantage of ESMVU is more distinct when the noise ratio is higher. These observations are shown in Fig. 2.

In Table 4, the test accuracies of different methods are presented for the splice, australian, and twonorm datasets. The observations in Table 4 are similar to those of Table 3. For all three datasets, when the noise ratio is low (for example, 0.1), ESMVU is usually not the best method. However, when the noise ratio is high (for example, 0.3 or 0.4), ESMVU outperforms the other filters. The relation between the noise ratio and the classification accuracy in this table is shown in Fig. 3.

Table 5 gives the experimental results for the nursery, balance-scale, and haberman datasets. For the nursery dataset, the classification accuracies under each noise ratio are quite similar because the number of instances in nursery is much larger than that in the other datasets. In this case, although some training instances are mislabeled, the remaining noise-free instances are sufficient to train a reliable classifier. Therefore, the adverse effects of mislabeled data

TABLE 4. Experimental results for the splice, australian, and twonorm datasets.

Method	Test accuracy				
	0	0.1	0.2	0.3	0.4
splice					
None	**0.8415**	0.7165	0.6693	0.5785	0.5625
CF	0.8327	0.7795	0.7165	0.7008	0.6772
EF	0.8336	0.7953	0.7402	0.7086	0.6851
IPF	0.8407	**0.8032**	**0.7559**	0.7161	0.6929
ENN	0.8039	0.7480	0.7077	0.6538	0.6077
ESMVU	0.8329	0.7638	0.7487	**0.7244**	**0.7108**
australian					
None	0.8652	0.7522	0.6913	0.5707	0.5391
CF	0.8609	0.8174	0.8022	0.7391	0.6957
EF	0.8435	0.8217	0.8152	0.7717	0.7391
IPF	0.8478	**0.8349**	**0.8239**	0.8043	0.7609
ENN	**0.8913**	0.7826	0.7609	0.6304	0.5652
ESMVU	0.8695	0.8260	0.8136	**0.8116**	**0.7826**
twonorm					
None	0.9416	0.8347	0.7708	0.6707	0.5866
CF	0.9448	0.9222	0.9006	0.8215	0.7452
EF	**0.9493**	0.9386	0.9337	0.9141	0.8624
IPF	0.9489	0.9364	**0.9371**	0.9037	0.7723
ENN	**0.9493**	0.9310	0.8783	0.7809	0.7368
ESMVU	0.9453	**0.9452**	**0.9371**	**0.9452**	**0.8850**

are weak for this dataset. In addition, the performances of each filter are similar. For the balance-scale dataset, ESMVU has relatively poor performance. However, when the noise ratio increases to 0.2, ESMVU improves. For the haberman dataset, ESMVU is superior to the other filters. The results in Fig. 4 clearly reflect the relation of the noise ratio and classification accuracy.

ESMVU consists of several parameters, among which the threshold that determines the generation of the indReliable dataset is one of the most important. Because indReliable is

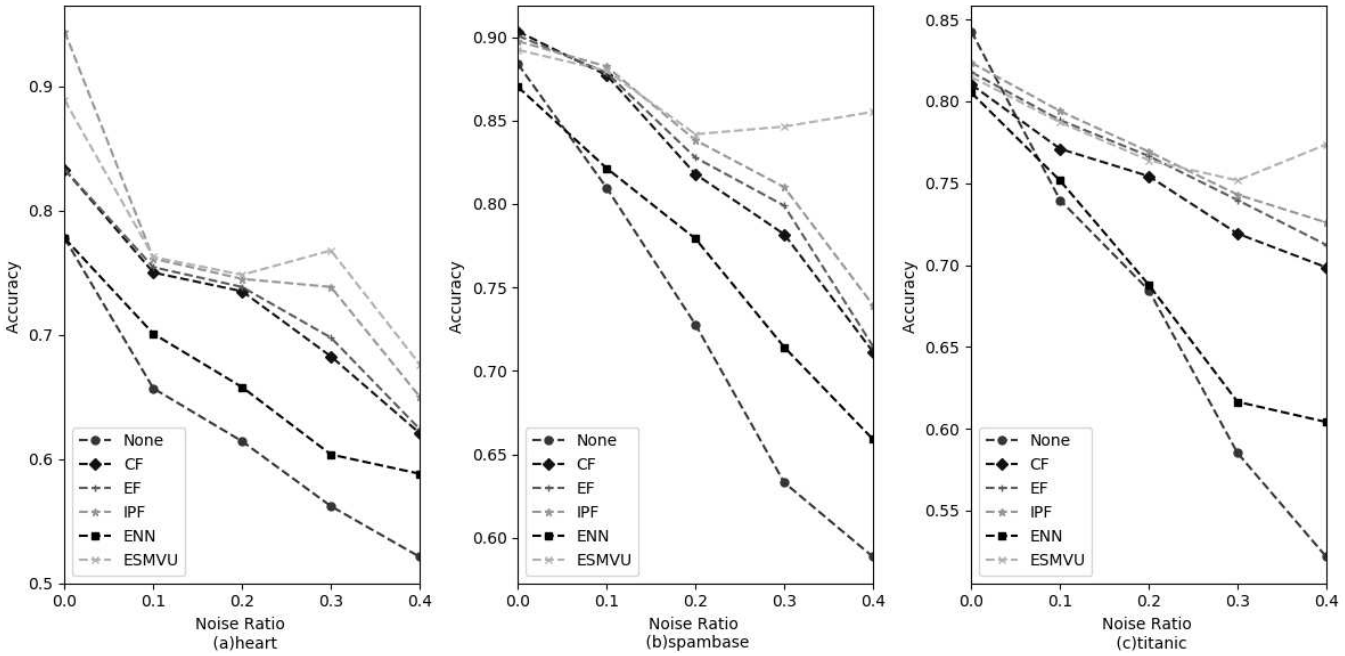


FIGURE 2. Classification comparison for the heart, spambase, and titanic datasets.

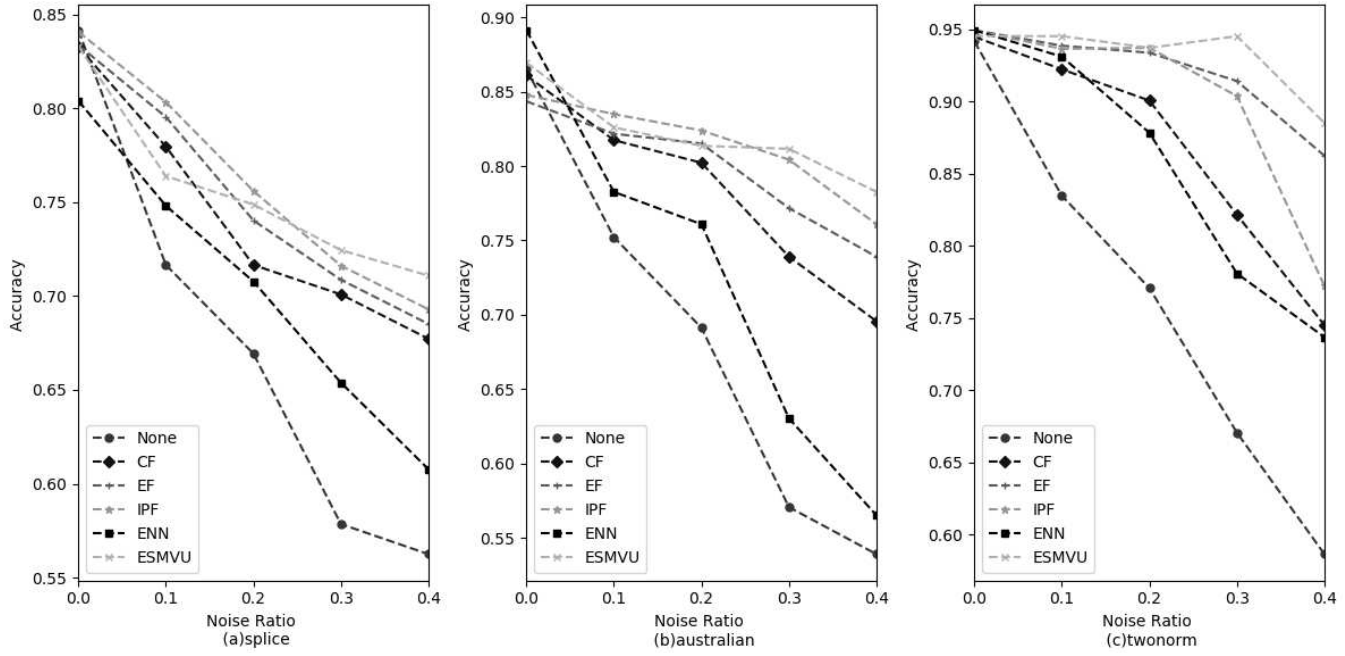


FIGURE 3. Classification comparison for the splice, australian, and twonorm datasets.

used to label the unlabeled data, which is the key point in ESMVU, we consider the impact of this threshold in Table 6, where the best results at each noise ratio are highlighted in bold. Several observations are found. First, when the noise ratio is fixed, the classification accuracy increases when the threshold changes from 0.5 to 0.8. This is expected because a higher threshold would result in the removal of more mislabeled data and provide a relatively cleaner training dataset. However, when the threshold changes from 0.8 to 0.9, the classification accuracy worsens, indicating that a higher threshold does not always result in higher classification accuracy. When the threshold is sufficiently high, many noise-free instances with relatively medium confidence are identified as noises and removed. In this case, only a small amount

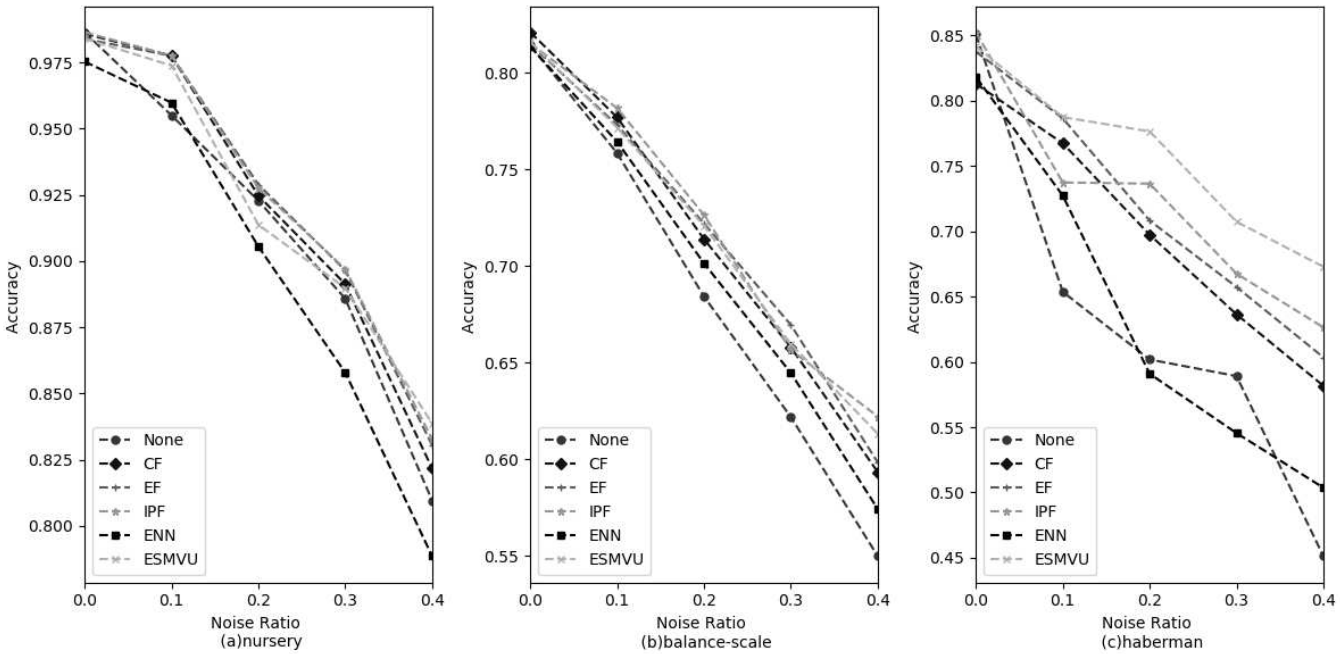


FIGURE 4. Classification comparison for the nursery, balance-scale, and haberman datasets.

TABLE 5. Experimental results for the nursery, balance-scale, and haberman datasets.

Method	Test accuracy				
	0	0.1	0.2	0.3	0.4
Nursery					
None	**0.9864**	0.9550	0.9228	0.8858	0.8096
CF	0.9858	**0.9775**	0.9247	0.8913	0.8217
EF	0.9839	**0.9775**	**0.9289**	0.8965	0.8305
IPF	**0.9864**	**0.9775**	0.9276	**0.8970**	0.8328
ENN	0.9753	0.9598	0.9055	0.8579	0.7886
ESMVU	0.9840	0.9738	0.9137	0.8898	**0.8387**
balance-scale					
None	0.8142	0.7584	0.6842	0.6218	0.5498
CF	**0.8210**	0.7769	0.7139	0.6579	0.5933
EF	0.8149	0.7729	0.7220	**0.6695**	0.5983
IPF	0.8152	**0.7817**	**0.7264**	0.6565	**0.6221**
ENN	0.8136	0.7643	0.7014	0.6446	0.5742
ESMVU	0.8160	0.7716	0.7208	0.6596	0.6132
haberman					
None	0.8507	0.6536	0.6018	0.5891	0.4513
CF	0.8128	0.7673	0.6973	0.6364	0.5817
EF	0.8375	0.7864	0.7082	0.6573	0.6035
IPF	**0.8523**	0.7373	0.7364	0.6675	0.6264
ENN	0.8181	0.7273	0.5909	0.5455	0.5037
ESMVU	0.8416	**0.7873**	**0.7764**	**0.7073**	**0.6728**

of training data is retained. Although the retained data are usually noise-free, a good classifier cannot be constructed when the number of training data is small. Meanwhile, if the threshold is too small, excessive mislabeled training data will be retained, which may result in poor performance. In our study, the optimal threshold is 0.8.

The following conclusions can be made based on the experimental results in this section: 1) for most of the datasets, the performance of ESMVU is superior to other approaches.

TABLE 6. Analysis of the influence of the main parameters.

Threshold	Test accuracy				
	0	0.1	0.2	0.3	0.4
0.9	**0.7985**	0.7737	**0.7843**	0.7777	0.7544
0.8	0.7910	**0.7849**	0.7718	**0.8021**	**0.7776**
0.7	0.7789	0.7688	0.7448	0.7345	0.7252
0.6	0.7783	0.7516	0.7429	0.7332	0.7198
0.5	0.7841	0.7028	0.6536	0.5912	0.5096

This verifies that in ESMVU, unlabeled instances are used in a reasonable manner; 2) the superiority of ESMVU is more obvious when the noise ratio is higher. For example, when the noise ratio is 40%, the performance improvement of ESMVU is highest; 3) selecting a suitable threshold value for ESMVU is important. This selection can be derived from experimental results. If the experiments are not available, it is suggested to select a medium value (such like 0.7 and 0.8).

V. CONCLUSION AND FUTURE WORKS

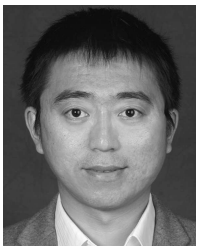
In this work, we propose a novel approach, enhanced soft majority voting by exploiting unlabeled data (ESMVU) for mislabeled data filtering. In contrast to the existing approaches, ESMVU uses unlabeled data to aid the identification process of mislabeled training data. In ESMVU, the multiple voting framework and confidence measurement are adopted to improve the classification accuracy. The performance of ESMVU is evaluated with UCI and KEEL datasets. The experimental results show that ESMVU improves the performance of existing noise filtering methods.

Noise correction is another way to address mislabeled data. In this study, we consider noise filtering because we assume

that the amount of training data is usually large in big data analysis. In the future, we will consider noise correction and compare its performance with that of noise filtering. In addition, the big data and heterogeneous data will be also considered in the future work.

REFERENCES

[1] X. Zhu and X. Wu, "Class noise vs. attribute noise: A quantitative study," *Artif. Intell. Rev.*, vol. 22, no. 3, pp. 177–210, Nov. 2004.
[2] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986.
[3] C. E. Brodley and M. A. Friedl, "Identifying mislabeled training data," *J. Artif. Intell. Res.*, vol. 11, no. 1, pp. 131–167, 1999.
[4] D. Gamberger, N. Lavrac, and S. Dzeroski, "Noise elimination in inductive concept learning: A case study in medical diagnosis," in *Algorithmic Learning Theory*. Berlin, Germany: Springer, 1996, pp. 199–212.
[5] D. Gamberger, N. Lavrac, and S. Dzeroski, "Noise detection and elimination in data preprocessing: Experiments in medical domains," *Appl. Artif. Intell.*, vol. 14, no. 2, pp. 205–223, Nov. 2000.
[6] J. R. Rico-Juan and J. M. Inesta, "Adaptive training set reduction for nearest neighbor classification," *Neurocomputing*, vol. 138, pp. 316–324, Aug. 2014.
[7] J. Calvo-Zaragoza, J. J. Valero-Mas, and J. R. Rico-Juan, "Improving kNN multi-label classification in Prototype Selection scenarios using class proposals," *Pattern Recognit.*, vol. 48, no. 5, pp. 1608–1622, May 2015.
[8] S. Kanj, F. Abdallah, T. Denoeux, and K. Tout, "Editing training data for multi-label classification with the k-nearest neighbor rule," *Pattern Anal. Appl.*, vol. 19, no. 1, pp. 145–161, Feb. 2015.
[9] F. Roli, "Multiple classifier systems," in *Encyclopedia Biometrics*. Boston, MA, USA: Springer, 2015, pp. 1142–1147.
[10] M. Wozniak, M. Grana, and E. Corchado, "A survey of multiple classifier systems as hybrid systems," *Inf. Fusion*, vol. 16, pp. 3–17, Mar. 2014.
[11] L. I. Kuncheva and J. J. Rodriguez, "A weighted voting framework for classifiers ensembles," *Knowl. Inf. Syst.*, vol. 38, no. 2, pp. 259–275, Feb. 2014.
[12] S. Sun, "Local within-class accuracies for weighting individual outputs in multiple classifier systems," *Pattern Recognit. Lett.*, vol. 31, no. 2, pp. 119–124, Jan. 2010.
[13] J. A. Saez, M. Galar, J. Luengo, and F. Herrera, "Tackling the problem of classification with noisy data using multiple classifier systems: Analysis of the performance and robustness," *Inf. Sci.*, vol. 247, pp. 1–20, Oct. 2013.
[14] J. A. Sáez, M. Galar, J. Luengo, and F. Herrera, "Analyzing the presence of noise in multi-class problems: Alleviating its influence with the one-vs-one decomposition," *Knowl. Inf. Syst.*, vol. 38, no. 1, pp. 179–206, 2014.
[15] R. Barandela, R. M. Valdovinos, and J. S. Sanchez, "New applications of ensembles of classifiers," *Pattern Anal. Appl.*, vol. 6, no. 3, pp. 245–256, Dec. 2003.
[16] J. S. Sanchez and L. I. Kuncheva, "Data reduction using classifier ensembles," in *Proc. ESANN*, Jan. 2007, pp. 379–384.
[17] I. Kononenko and M. Kukar, *Machine Learning and Data Mining: Introduction to Principles and Algorithms*. Sawston, U.K: Horwood Publishing, 2007.
[18] R. Y. Wang, V. C. Storey, and C. P. Firth, "A framework for analysis of data quality research," *IEEE Trans. Knowl. Data Eng.*, vol. 7, no. 4, pp. 623–640, Aug. 1995.
[19] C. M. Teng, "Polishing blemishes: Issues in data correction," *IEEE Intell. Syst.*, vol. 19, no. 2, pp. 34–39, Mar. 2004.
[20] J. A. Saez, M. Galar, J. Luengo, and F. Herrera, "INFFC: An iterative class noise filter based on the fusion of classifiers with noise sensitivity control," *Inf. Fusion*, vol. 27, pp. 19–32, Jan. 2016.
[21] P. Smyth, "Bounds on the mean classification error rate of multiple experts," *Pattern Recognit. Lett.*, vol. 17, no. 12, pp. 1253–1257, 1996.
[22] G. Lugosi, "Learning with an unreliable teacher," *Pattern Recognit.*, vol. 25, no. 1, pp. 79–87, Jan. 1992.
[23] S. L. Salzberg, "C4.5: Programs for machine learning," *Mach. Learn.*, vol. 16, no. 3, pp. 235–240, Sep. 1994.
[24] W. W. Cohen, "Fast effective rule induction," in *Proc. ICML*, Jul. 1995, pp. 115–123.
[25] T. M. Khoshgoftaar and P. Rebours, "Improving software quality prediction by noise filtering techniques," *J. Comput. Sci. Technol.*, vol. 22, no. 3, pp. 387–396, May 2007.
[26] D. Gamberger, N. Lavrac, and C. Groselj, "Experiments with noise filtering in a medical domain," in *Proc. ICML*, 1999, pp. 143–151.
[27] D. Guan, W. Yuan, T. Ma, and S. Lee, "Detecting potential labeling errors for bioinformatics by multiple voting," *Knowl.-Based Syst.*, vol. 66, pp. 28–35, Aug. 2014.
[28] V. Sheng, J. Zhang, B. Gu, and X. Wu, "Majority voting and pairing with multiple noisy labeling," *IEEE Trans. Knowl. Data Eng.*, to be published, doi: 10.1109/TKDE.2017.2659740.
[29] C. E. Brodley and M. A, Friedl, "Identifying and eliminating mislabeled training instances," in *Proc. AAAI*, 1996, pp. 799–805.
[30] G. Han, X. Yang, L. Liu, W. Zhang, and M. Guizani, "A disaster management-oriented path planning for mobile anchor node-based localization in wireless sensor networks," *IEEE Trans. Emerg. Topics Comput.*, to be published, doi: 10.1109/TETC.2017.2687319.
[31] G. Han, L. Liu, S. Chan, R. Yu, and Y. Yang, "HySense: A hybrid mobile CrowdSensing framework for sensing opportunities compensation under dynamic coverage constraint," *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 93–99, Mar. 2017.
[32] G. Han, J. Jiang, M. Guizani, and J. J. P. C. Rodrigues, "Green routing protocols for wireless multimedia sensor networks," *IEEE Wireless Commun.*, vol. 23, no. 6, pp. 140–146, Dec. 2016.
[33] G. Jia, G. Han, J. Jiang, and L. Liu, "Dynamic adaptive replacement policy in shared last-level cache of DRAM/PCM hybrid memory for big data storage" *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 1951–1960, Apr. 2017.
[34] C. Mao, B. Hu, M. Wang, and P. Moore, "Learning from neighborhood for classification with local distribution characteristics," in *Proc. IJCNN*, Jul. 2015, pp. 1–8.
[35] S. A. Dudani, "The distance-weighted k-nearest-neighbor rule," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. SMS-6, no. 4, pp. 325–327, Apr. 1976.
[36] G. J. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic*, vol. 4. Englewood Cliffs, NJ, USA: Prentice-Hall, 1995.
[37] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, and S. García, "KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *J. Multiple-Valued Logic Soft Comput.*, vol. 17, nos. 2–3, pp. 255–287, 2011.
[38] *UCI Machine Learning Repository*. Accessed: 2013. [Online]. Available: http://archive.ics.uci.edu/ml/index.php



DONGHAI GUAN received the Ph.D. degree from Kyung Hee University (KHU), Suwon, South Korea, in 2009. From 2009 to 2011, he was a Research Professor with the Department of Computer Engineering, KHU. From 2012 to 2014, he was an Assistant Professor with KHU. He is currently an Associate Professor with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China. He has authored over 70 research papers in related international conferences and journals. His current research interests include machine learning, artificial intelligence, recommender systems, social network analysis, and ubiquitous computing.

HONGQIANG WEI received the B.S. degree from the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China, in 2016, where he is currently pursuing the master's degree. His research interests mainly focus on machine learning and data mining.

WEIWEI YUAN received the B.S. and M.S. degrees from Harbin Engineering University, China, in 2002 and 2005, respectively, and the Ph.D. degree from the Department of Computer Engineering, Kyung Hee University, South Korea, in 2010. She is currently an Associate Professor with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China. Her current research interests include pattern recognition, social computing, and recommender systems.

GUANGJIE HAN (S'01–M'05) received the Ph.D. degree from Northeastern University, Shenyang, China, in 2004. From 2004 to 2006, he was the Product Manager with the ZTE Company. Until 2008, he was a Post-Doctoral Researcher with the Department of Computer Science, Chonnam National University, Gwangju, South Korea. From 2010 to 2011, he was a Visiting Research Scholar with Osaka University, Suita, Japan. He is currently a Professor with the Department of Information and Communication System, Hohai University, Changzhou, China. He has authored over 280 papers, published in international conference proceedings and journals. He holds 110 patents. His current research interests include sensor networks, computer communications, mobile cloud computing, and multimedia communication and security. He is a member of the ACM. He received the Best Paper Award for ComManTel 2014, ComComAP 2014, Chinacom 2014, and Qshine 2016. He has served as the co-chair for more than 50 international conferences/workshops and as a technical program committee member for more than 150 conferences. He has served on the editorial boards of 14 international journals, including the IEEE ACCESS, the *Telecommunication Systems*, the *International Journal of Ad Hoc and Ubiquitous Computing*, the *Journal of Internet Technology*, and the *KSII Transactions on Internet and Information Systems*. He guest edited a number of special issues in the IEEE journals and magazines. He has served as a reviewer for more than 50 journals.

YUAN TIAN received the M.S. and Ph.D. degrees from Kyung Hee University. She is currently an Assistant Professor with the College of Computer and Information Sciences, King Saud University, Saudi Arabia. Her research interests are broadly divided into privacy and security related to cloud computing, bioinformatics, multimedia, cryptography, smart environments, and big data. She is a member of the technical committees of several international conferences. She is an active reviewer for many international journals.

MOHAMMED AL-DHELAAN received the M.S. degree and the Ph.D. degree in computer science from The George Washington University, Washington, DC, USA, in 2009 and 2014, respectively. He then moved to become an Assistant Professor with the Computer Science Department, King Saud University, Saudi Arabia. He has supervised M.S. students and taught several courses at both the undergraduate and graduate levels. His research interests include natural language processing and data mining. Specifically, he is focused on graph-based ranking, extracting keyphrases, statistical topic models, and text summarization. He has participated as a PC member at many conferences, where he reviewed several research articles.

ABDULLAH AL-DHELAAN received the B.S. degree (Hons.) in statistics from King Saud University, Riyadh, Saudi Arabia, in 1982, and the M.S. and Ph.D. degrees in computer science from Oregon State University in 1986 and 1989, respectively. He is currently the Vice Dean for Academic Affairs , the Dean of Graduate Studies, and a Professor of computer science with King Saud University. His current research interests include mobile ad hoc networks, sensor networks, cognitive networks, network security, image processing, and high-performance computing. He guest edited several special issues for the *Telecommunication Journal* (Springer) and the *International Journal of Computers and Their Applications* (ISCA). Moreover, he is currently on the editorial boards of several journals and in the organizing committees for several reputable international conferences.