Neural Comput & Applic (2018) 29:673–683
https://doi.org/10.1007/s00521-016-2589-9

ORIGINAL ARTICLE

Novel mislabeled training data detection algorithm

Weiwei Yuan[1,2] · Donghai Guan[1,2] · Qi Zhu[1,2] · Tinghuai Ma[3]

Received: 24 February 2016 / Accepted: 6 September 2016 / Published online: 16 September 2016

Abstract As a kind of noise, mislabeled training data exist in many applications. Because of their negative effects on learning, many filter techniques have been proposed to identify and eliminate them. Ensemble learning-based filter (EnFilter) is the most widely used filter which employs ensemble classifiers. In EnFilter, first the noisy training dataset is divided into several subsets. Each noisy subset is then checked by the multiple classifiers which are trained based on other noisy subsets. It is noted that since the training data used to train multiple classifiers are noisy, the quality of these classifiers cannot be guaranteed, which might generate poor noise identification result. This problem is more serious when the noise ratio in the training dataset is high. To solve this problem, a straightforward but effective approach is proposed in this work. Instead of using noisy data to train the classifiers, nearly noise-free (NNF) data are used since they are supposed to train more reliable classifiers. To this end, a novel NNF data extraction approach is also proposed. Experimental results on a set of benchmark datasets illustrate the utility of our proposed approach.

Keywords Mislabeled data filtering · Ensemble learning · Noise-free data

✉ Donghai Guan
dhguan@nuaa.edu.cn

Weiwei Yuan
yuanweiwei@nuaa.edu.cn

Qi Zhu
zhuqi@nuaa.edu.cn

Tinghuai Ma
thma@nuist.edu.cn

[1] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211-106, Jiangsu, China

[2] Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing, Jiangsu 210-023, China

[3] Jiangsu Engineering Centre of Network Monitoring, Nanjing University of Information Science and Technology, Nanjing 210-044, Jiangsu, China

1 Introduction

As an important type of noises, label noises widely exist in many applications [1–6]. A label noise means the label given to an instance is incorrect; thus, a label noise corresponds to a mislabeled data [7–12]. In general, learning crucially relies on the accuracy of labels in the training dataset. Therefore, a significant amount of interest has been attracted to deal with mislabeled training data in machine learning community.

Although the importance of accurate labels is well recognized, incorrect/noisy labels cannot be thoroughly avoided in real-world applications. The main reasons include subjective nature of the labeling task and the insufficient information to make the true label. Labeling is usually given by the domain experts based on their domain knowledge and subjective understanding. Even for domain experts, the mislabeling cannot be thoroughly avoided. Mislabeling will be generated when the annotations given by multiple experts disagree with the general consensus. Mislabeling is common in the rapid developing domain, like bioinformatics. For example, there are nine subjective mislabeling among forty-nine breast tumor training data in [13]. On the other hand, mislabeling can also be made if the information provided to the expert is insufficient [14, 15]. For example, if certain

expensive tests are missing, a physician may make an incorrect diagnosis.

To deal with mislabeled data, many approaches have been developed which can be mainly categorized into two groups: robust algorithm design [16–22] and noise filter [23–35]. Robust algorithm design focuses on developing novel algorithms which are insensitive to noises during model training. But several works have shown that developing such a robust algorithm is usually difficult. Moreover, although with robust design, the existence of mislabeled data can still introduce serious negative effect to this approach. The filter approach identifies and filters mislabeled data prior to training, which shows satisfying performances in many works. The work described here belongs to filters.

A number of filters have been proposed to identify mislabeled data, among which ensemble learning-based filter (EnFilter) has shown promising performance and therefore being most widely used. In contrast to single classifier-based filters (ScFilter), EnFilter employs multiple classifiers and identifies noises based on their predictions. EnFilter is superior to ScFilter because in general the predictions of ensemble classifiers are more reliable compared to single classifier.

A variety of EnFilter algorithms have been proposed, but their basic ideas are similar. In most EnFilters, the noisy training dataset E is firstly randomly partitioned into several subsets $(E_1, E_2, \ldots, E_n)$. Each subset E_i then will be checked for mislabeled data separately. The checking is through the voting of multiple classifiers which are trained based on remaining subsets $(E \backslash E_i)$. Let $y()$ represents the mislabeled data identification function of an EnFilter. If there is an instance x belonging to E_i to be checked noise-free or mislabeled, then the decision on x is $y(x) = \text{vote}(\text{classifiers}(E \backslash E_i))$. It should be noted that because E is noisy, its subset $E \backslash E_i$ is also noisy. The classifiers trained based on $E \backslash E_i$ are not reliable even the voting mechanism could improve the reliability to some extent. This problem is more serious when the noise ratio in E is high.

To solve this issue, a straightforward but effective approach is proposed in this paper. The basic idea of our approach is: first extract nearly noise-free data from noisy training dataset E. So E is divided into two subsets: E_{nnf} (nearly noise-free subset) and E_{ques} (questionable subset); second E_{ques} is checked by the voting of multiple classifiers which are trained based on E_{nnf}. According to our approach, the identification on an instance x can be represented by $y(x) = \text{vote}(\text{classifiers}(E_{\text{nnf}}))$. Compared to conventional EnFilters, our mislabeled data identification result is expected to be more accurate because E_{nnf} is used for training instead of noisy instances used by existing methods.

Extracting nearly noise-free subset E_{nnf} from E is the key of our approach. If E_{nnf} is far from noise-free, our approach is not able to work well. Obtaining sufficient clean E_{nnf} is hard for conventional EnFilters which usually use single-voting mechanism. For single-voting-based EnFilters, their identification results can be easily influenced by the data partitioning. To alleviate this influence and provide more reliable identification performance, a multiple-voting mechanism is used to extract E_{nnf}. In addition, one-vote veto policy is adopted, by which an instance is put into E_{nnf} only when all the voters believe that this is a noise-free data.

We compare our approach with conventional EnFilters. The experimental results show that our approach makes less number of errors during mislabeled data identification. In particular, when the noise ratio is high, the advantage of our approach is more obvious. In addition, our approach is straightforward and only few predefined parameters are required.

In the next section, we will briefly review ensemble learning-based noise filters. Section 3 presents our approach. The experimental evaluations are presented in Sect. 4. Section 5 concludes this work and presents future works.

2 Related works

Mislabeled training data widely exist in various applications. Both theoretical and empirical evidences have been done to show the consequences of label noise on learning, which mainly include: (1) deterioration of classification (regression) performances [36–40], (2) affecting learning requirements (e.g., number of necessary training data), (3) increasing the complexity of learned models. In addition to above harmfulness, mislabeled data can also degrade other related works. For example, Guan et al. showed that the consequences of mislabeled data will decrease feature selection and feature ranking performances [41]. In this work, we mainly consider the consequences of mislabeled data for classification tasks [42].

Based on whether removing mislabeled data, there are two ways to handle mislabeled training data for classification problems: (1) designing noise-robust models, (2) filtering out mislabeled data. For the former, even if mislabeled data are neither removed nor modeled, the harmfulness of mislabeled data is effectively alleviated by designing noise-robust models. Usually this method is relatively effective only when small amounts of label noise exist. For the latter method, the mislabeled data will be identified and removed/corrected prior to training. Compared to noise-robust model design, this method is used

more widely. Our work in this paper is in the scope of mislabeled data filtering.

To filter mislabeled data, there are various approaches. Some researchers make use of the existing ideas in outlier detection area and develop some types of measures to identify label noises. For example, because label noise may increase the complexity of inferred models, complexity measures can be used to identify mislabeled data. Mislabeled data can also be recognized by analyzing their impact on learning. For example, the leave-one-out perturbed classification matrix has been used to identify mislabeled data. In this method, a sample is regarded as mislabeled if its flip could improve the overall performances of the classifier. In addition, kNN-based methods have been used, because kNN classifiers are sensitive to label noises. This type of methods is mainly based on heuristics. For example, the condensed nearest neighbor algorithm keeps a subset of training data that allows classifying correctly all other training data. Similar to kNN-based methods, graph-based mislabeled data identifying algorithms exist. Their idea is similar to kNN-based methods with the exception that they represent training data by neighborhood graphs. In neighborhood graphs, the instances are linked to other neighbor instances by edges. In addition to above methods, in recently years, the most widely used method is ensemble learning-based methods, which are also called ensemble learning filters (EnFilter). By making use of the advantages of ensemble learning idea, this type of method is straightforward to use and usually shows satisfied noise detection accuracy [43–45].

Majority filtering (MF) and consensus filtering (CF) are two well-known mislabeled data filters, which show good performances in many works [23, 24]. They are ensemble learning-based filters (EnFilter). In this work, we propose a new filter based on them. As background knowledge, they are introduced in this section.

In MF and CF, a set of base-level classifiers are constructed and their classification results are fused to identify mislabeled data. Their decision rule is to tag a data as mislabeled if x of the m base-level classifiers cannot classify it correctly. MF is based on majority voting, wherein data are treated as mislabeled if more than half of the m-based level classifiers classify it incorrectly. CF is based on consensus voting, wherein training data are identified as mislabeled only if all base-level classifiers fail to classify the data as the class given by its training label.

MF and CF use ensemble learning idea because ensemble classifier is superior to each base-level classifier on classification if two conditions hold: (1) the probability of a correct classification by each individual classifier is > 0.5 and (2) the errors in predictions of the base-level classifiers are independent.

Shown in Table 1, majority filtering 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 is formed which includes all examples from E except those in E_i, which then is used as the input an arbitrary inductive learning algorithm that induces a hypothesis (a classifier) H_j (step 6). Those examples from E_i for which majority of the hypotheses does not give the correct classification are added to A as potentially noisy examples (step 14).

Table 1 Majority filtering algorithm

Algorithm 1: Majority Filtering (MF)
Input: E (training set)
Parameter: n (number of subjects), y (number of learning algorithms), $A_1, A_2, \ldots, A_y$ (y kinds of learning algorithms)
Output: A (detected noisy subset of E)
(1) form n disjoint almost equally sized subset of E_i, where $\bigcup_i E_i = E$
(2) $A \leftarrow \emptyset$
(3) **for** i=1, . . . , n **do**
(4) form $E_t \leftarrow E \backslash E_i$
(5) **for** j=1,. . . 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
(12) **then** $ErrorCounter \leftarrow ErrorCounter + 1$
(13) **end for**
(14) **if** $ErrorCounter > \frac{y}{2}$, **then** $A \leftarrow A \cup \{e\}$
(15) **end for**
(16) **end for**

Consensus filtering algorithm is similar to MF. Its only difference with MF is at step 14. In CF, the example in E_i is regarded as a mislabeled example only when all the hypotheses incorrectly classify it. Compared to MF, CF is more conservative due to the severer condition for noise identification, which results in fewer instances being eliminated from the training set. The drawback of CF is the added risk in retaining bad data.

3 Our proposed mislabeled data detection approach

Ensemble learning-based filters adopt ensemble learning idea to detect mislabeled data. They in general consist of two steps. The first step uses a K-fold cross-validation scheme, which creates K pairs of distinct training and validation datasets ($\mathrm{TD}_{i(i=1:k)}$ and $\mathrm{VD}_{i(i=1:k)}$). For each pair of sets, m learning algorithms are used to learn m classifiers using the training set and to classify the samples in the validation set. Since each instance belongs to exactly one validation set, m classifications are obtained for each sample. In the second step, different voting policies can be used to fuse the m classification results. For example, MF adopts majority vote and CF adopts consensus vote. For a given instance x, suppose x belongs to VD_i. Then, EnFilter will predict the label of x as $y(x) = \mathrm{vote}(\mathrm{classifiers}(\mathrm{TD}_i))$.

The motivation to use ensemble classifiers in EnFilter is that in general ensemble classifiers have better classification performance compared to each individual classifier. However, if the classification accuracy of each base classifier is too low, the improvement achieved by ensemble learning is also very limited. In particular, theoretically ensemble learning works only if the classification accuracy of each classifier is > 0.5. So as the training data to train multiple classifiers, TD_i plays an important role that affects the noise identification accuracy.

Most EnFilters, like MF and CF, create TD_i just by a random manner. For example, the noisy training dataset E is randomly partitioned into several subsets $(E_1, E_2, \ldots, E_n)$. Each subset E_i is the validation dataset VD_i, while remaining subsets $(E \backslash E_i)$ are used as TD_i. As the random subset of E, TD_i usually contains mislabeled data. Theoretically, the noise ratio in TD_i is identical to that of E. As we know, the classifiers trained from noisy data are not reliable, especially when the noise ratio is high. As a result, the noise prediction accuracy might be low although the voting mechanism could improve this accuracy to some extent.

Above analysis shows that conventional EnFilters suffer from a chicken-and-egg dilemma, because (1) good classifiers are necessary for EnFilters (2) learning in the presence of mislabeled data may produce poor classifiers.

To solve this "chicken-and-egg" issue, a straightforward but effective approach is proposed in this paper. Instead of training classifiers based on mislabeled data, noise-free data will be extracted from original noisy training dataset, which are used to train classifiers. Obviously, the classifiers trained from noise-free data are better than that trained from mislabeled data. Correspondingly, the mislabeled data detection accuracy can also be improved.

Now the main challenge is to develop an approach to extract the noise-free data from E. The ideal approach is able to extract all the noise-free data without any errors. But in practice, this ideal approach is almost impossible. So we try to extract nearly noise-free (NNF) data to use for training, which is represented by E_{nnf}. Although E_{nnf} will inevitably contain some noises, we try to keep the number of noises so small that their negative influence on training can be ignored.

Utilizing conventional EnFilters, it is hard to create the satisfactory E_{nnf}. It is mainly due to the inherent scheme of EnFilters. In the first step of EnFilter, like MF and CF, distinct training and validation datasets are created ($\mathrm{TD}_{i(i=1:k)}$ and $\mathrm{VD}_{i(i=1:k)}$). Given a instance x, if it belongs to VD_i, then the prediction on x can be represented by $y(x) = \mathrm{vote}(\mathrm{classifiers}(\mathrm{TD}_i))$. However, if it belongs to VD_j, then we have $y(x) = \mathrm{vote}(\mathrm{classifiers}(\mathrm{TD}_j))$. The generation of validation and training dataset adopts a random partitioning method by most EnFilters. For example, in MF and CF, the dataset E is randomly partitioned into several subsets. Then, each subset will iteratively be used as validation set, while the other subsets are used as training set. Because there is diversity between TD_i and TD_j, the prediction results of $y(x) = \mathrm{vote}(\mathrm{classifiers}(\mathrm{TD}_i))$ and $y(x) = \mathrm{vote}(\mathrm{classifiers}(\mathrm{TD}_j))$ can be also different. This indicates that the prediction result of conventional EnFilters is not reliable enough. Therefore, the E_{nnf} created by them might contain overmuch noises.

With this observation, we develop an approach which consists of multiple EnFilters and makes the prediction by aggregating the predictions of these multiple EnFilters. Suppose t EnFilters are included by our approach, then the ith EnFilter will give its own prediction about whether an instance is mislabeled $y_i(x) = \mathrm{vote}(\mathrm{classifiers}(\mathrm{TD}_i))$. The value of $y_i(x)$ is binary, where 1 means x is a mislabeled data and 0 means x is noise-free. The various results of $y_i(x)$ will be fused to make the final prediction. Our mission here is to guarantee the purity of noise-free data in E_{nnf}; therefore, any instance with even low mislabeled probability should also be excluded from E_{nnf}. This actually is a one-vote veto scheme. If there is at least one EnFilter predicts that $y_i(x) = 1$, then x is treated as a potential mislabeled data. Our nearly noise-free data extraction method is illustrated in Fig. 1.

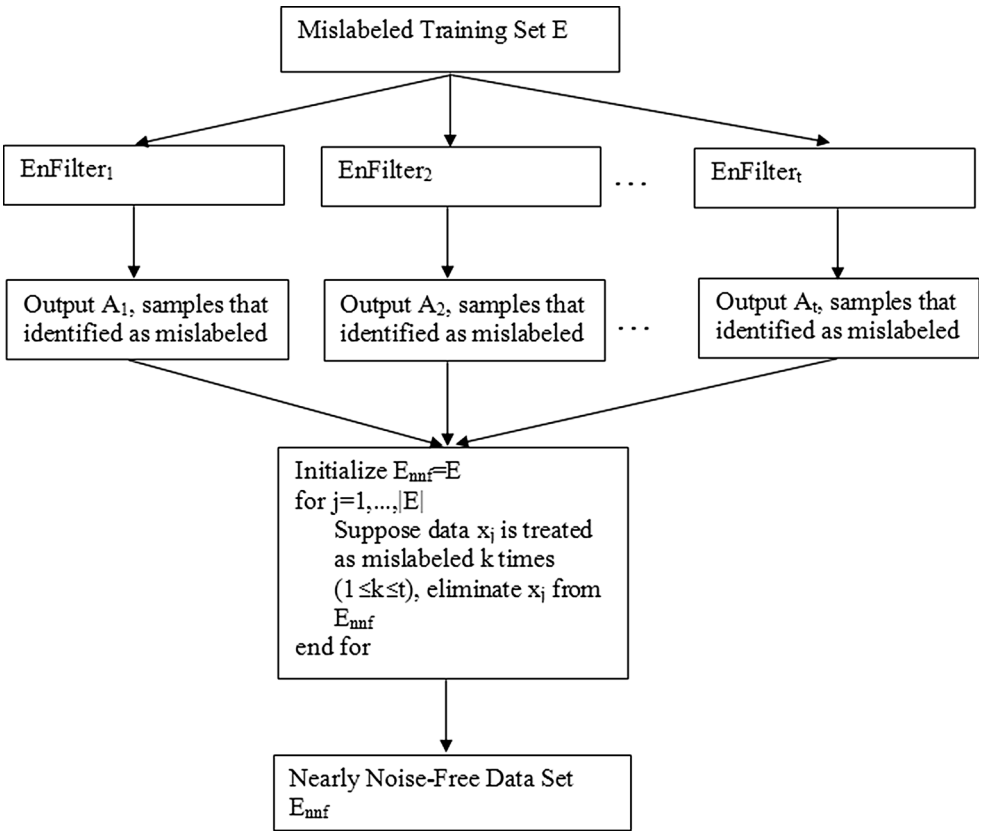

Fig. 1 Nearly noise-free data extraction scheme

In Fig. 1, each EnFilter can use either majority filter (MF) or consensus filter (CF). Since CF is more conservative to identify a data as mislabeled, there is higher probability to retain a mislabeled in E_{nnf} compared to MF. Thus, MF is more preferred to use as the base EnFilter in Fig. 1. As Fig. 1 contains multiple majority filters, the corresponding algorithm is called multiple majority filter based nearly noise-free data extraction method (MMF_{nnf}). The detailed algorithm of (MMF_{nnf}) is shown in Table 2.

In this algorithm, our main idea is to make use of majority filter to remove all the label noises and generate a nearly noise-free dataset. To compensate the instability of majority filter, an ensemble learning idea is used. As shown in step 1, the label noise identification by majority filter will be executed for t times. With t increasing, we have higher probability to detect all the label noises. But meanwhile the higher value of t will consume more computation time. Similar to the selection of individual classifiers in ensemble learning, it is not guaranteed that higher value of t brings better performance. The selection of this value can be determined by experiments.

In the following, we mathematically analyze the probabilities that E_{nnf} contains noises when using MMF_{nnf}.

Referred to the introduction of MF and CF in Sect. 2, they construct multiple classifiers and utilize their voting to make predictions. In identifying mislabeled instances, two types of error can be made. The first type ($E1$) occurs when declaring a correctly labeled example as mislabeled and is subsequently discarded. The second type of error ($E2$) corresponds to declare a mislabeled example as correctly labeled.

Let $P(E1_i)$ and $P(E2_i)$ be the probabilities that classifier i makes an $E1$ and $E2$ error, respectively. To simplify the analysis, it is assumed that all m various classifiers have the same probability of making an $E1$ error. That means $P(E1_i) = P(E1)$. Similarly it is assumed that $P(E2_i) = P(E2)$.

For majority filtering (MF), it makes an $E1$ (or $E2$) error when more than half of these m classifiers fail to classify the instance correctly. Therefore,

$$P(E1_{\text{MF}}) = \sum_{j>m/2}^{j=m} P(E1)^j (1-P(E1))^{m-j} \binom{m}{j}$$

$$P(E2_{\text{MF}}) = \sum_{j>m/2}^{j=m} P(E2)^j (1-P(E2))^{m-j} \binom{m}{j}$$

For consensus filtering, it makes an $E1$ error when all m classifiers incorrectly predict it. However, CF does not make an $E2$ error only when all m classifiers correctly predict it. Therefore,

$$P(E1_{\text{CF}}) = P(E1)^m$$

$$P(E2_{\text{CF}}) = 1 - (1-P(E2))^m$$

Table 2 Algorithm: multiple majority filter based nearly noise-free data extraction algorithm: MMF_{nnf}

MMF$_{nnf}$
Input: E (training set)
Parameter: n (number of subsets), y (number of learning algorithms),
t (number of times of subsets partitioning), A_1, A_2, …, A_y(y kinds of learning algorithms)
Output: A (detected noisy subset of E), E_{nnf} (noise-free subset of E)
(1) **for** p=1,…, t **do**
(2) form n disjoint almost equally sized subset of E_{pi}, where $\bigcup_i E_{pi} = E$
(3) $A^p \leftarrow \emptyset$
(4) **for** i=1, …, n **do**
(5) form $E_t \leftarrow E \backslash E_{pi}$
(6) **for** j=1,…y **do**
(7) induce H_{pj} based on examples in E_t and A_j
(8) **end for**
(9) **for** every $e \in E_{pi}$ **do**
(10) $ErrorCounter \leftarrow 0$
(11) **for** j=1,…,y **do**
(12) **if** H_{pj} incorrectly classifies e
(13) **then** $ErrorCounter \leftarrow ErrorCounter + 1$
(14) **end for**
(15) **if** $ErrorCounter > \frac{y}{2}$, **then** $A^p \leftarrow A^p \cup \{e\}$
(16) **end for**
(17) **end for**
(18) **end for**
(19) $A \leftarrow \emptyset$
(20) **for** every $e \in E$ **do**
(21) $ErrorCounter \leftarrow 0$
(22) **for** j=1,…,p **do**
(23) **if** $e \in A^p$
(24) **then** $ErrorCounter \leftarrow ErrorCounter + 1$
(25) **end for**
(26) **if** $ErrorCounter > 0$, then $A \leftarrow A \cup \{e\}$
(27) **end for**
(28) $E_{nnf} \leftarrow E \backslash A$

In general, $P(E1_{\text{MF}}) > P(E1_{\text{CF}})$ and $P(E2_{\text{CF}}) > P(E2_{\text{MF}})$. MMF_{nnf} algorithm in Table 2 consists of multiple (suppose this value is t) majority filters and combines their predictions based on one-vote veto.

Let $P(E1_{\text{MF}_i})$ denote the probability that MF_i makes an $E1$ error. With the value i changes, this probability could also change. However, to simplify the analysis, we assume that each $P(E1_{\text{MF}_i})$ is identical and equals to $P(E1_{\text{MF}})$. We make the similar assumption for $E2$ error. MMF_{nnf} consists of multiple MF filters. If one (or more) MF filter makes an $E1$ error, then MMF_{nnf} finally makes an $E1$ error. On the other hand, MMF_{nnf} makes an $E2$ error only if all the MF filters mistakenly declare the mislabeled instance as correctly labeled one. Therefore,

$$P(E1_{\text{MMF}_{\text{nnf}}}) = 1 - (1 - P(E1_{\text{MF}}))^t$$
$$P(E2_{\text{MMF}_{\text{nnf}}}) = P(E2_{\text{MF}})^t$$

If the nearly noise-free data extraction method is based on consensus filter, we call this method MCF_{nnf}. Then, we have

$$P(E1_{\text{MCF}_{\text{nnf}}}) = 1 - (1 - P(E1_{\text{CF}}))^t$$
$$P(E2_{\text{MCF}_{\text{nnf}}}) = P(E2_{\text{CF}})^t$$

An ideal noise identification method is expected to minimize the $P(E1)$ and $P(E2)$ simultaneously. However, $E1$ and $E2$ errors are always conflicting. In our noise-free data extraction method, our main goal is to keep E_{nnf} clean. In other words, minimizing $E2$ error is our main goal. We already know that $P(E2_{\text{CF}}) > P(E2_{\text{MF}})$, therefore $P(E2_{\text{MMF}_{\text{nnf}}}) < P(E2_{\text{MCF}_{\text{nnf}}})$. This means MMF_{nnf} could extract noise-free subset with higher purity compared to MCF_{nnf}. This analysis motivates us to use MMF_{nnf} instead of MCF_{nnf} in this paper. As we can see that, when t is a big value, $P(E2_{\text{MMF}_{\text{nnf}}})$ is a very small value which can be almost ignored.

4 Experimental work

In this section, a set of experiments are conducted to verify the effectiveness of our proposed approach. To test its performance, several representative ensemble learning-based

Table 3 Datasets in the experiment

Dataset	Num. of features	Num. of instances	Num. of classes
Vote	16	435	2
Australian	14	690	2
Heart	13	303	2
Indian	10	583	2
Horse	22	368	2
Credit	15	690	2
Spect	22	267	2
Spect1	44	267	2
Wine	13	178	3
Car	6	260	4
Wine_Quality	11	1518	5

filters are used to compare. These filters include conventional EnFilters (MF and CF) [22, 23], and recently proposed EnFilters (MF_{MF} and CF_{MF}) [35]. MF_{MF} consists of multiple majority filters. If more than half of these MFs identify an instance as mislabeled, then MF_{MF} will predict these data mislabeled. Likewise, CF_{MF} consists of multiple consensus filters, wherein an instance is predicted as mislabeled if more than half of CFs believe these data are mislabeled.

Eleven datasets from UCI repository are used in this experiment. Information on these datasets is tabulated in Table 3. In Table 3, the datasets include both binary classes (first eight datasets) and multiple classes (last three datasets). Note that the last dataset is a subset of original Wine_Quality-red dataset in UCI (we only keeps the data with class labels 5, 7, 9, 11, and 13).

In the experiments, MF and CF are configured as follows: number of subset is 3 ($n = 3$); three learning algorithms are used ($y = 3$) including naive Bayes, decision tree, and 3-Nearest Neighbor. The configurations of MF_{MF} and CF_{MF} are basically identical to MF/CF configurations. One additional parameter is the number of times of subset partitioning, which equals to twenty in the experiments.

Our proposed algorithm (referring to Table 2) is based on MF. Its configurations are also identical to MF configurations. The parameter t, number of times of subsets partitioning, is 40 in the experiment. The value of t determines the purity of extracted nearly noise-free (NNF) data. If t is bigger, the NNF data are more pure. But meanwhile, the number of instances in NNF is less. Therefore, the wise selection of t is dependent on the characteristics of datasets. Through our experiment, when t reaches to 40, the experimental results tend to stable.

In the experiments, each benchmark dataset was divided into a training set and a test set. Training set includes mislabeled data. Each filter algorithms remove mislabeled data from the training set. Each filter generates its own filtered training set, which then used to train a classifier to classify the test set. The classification accuracy and F1 score reflect the performance of each filter.

To evaluate the performance for each filter, each dataset D was processed as follows:

1. Three trials derived from threefold cross-validation of D were used to evaluate the performance of each mislabeled data filtering algorithm. During each trial, 2/3 of D, or Tr, was used as a training set. The remaining 1/3 of D was used as a test set, represented by Te. We artificially changed some labels that were originally correct in Tr, according to predefined mislabeled ratios to generate mislabeled data. For example, if we wanted to evaluate the algorithm performance on Tr under a 30 % mislabeled ratio, we randomly selected 30 % of the samples from Tr and changed correct labels to incorrect labels.
2. For each of three trials, there are two results are recorded for each filter, which include the classification accuracy and F1 score on Te. By averaging the results of three trials, we can get the average classification accuracy and F1 score.
3. Considering that the partitioning of D and that the mislabeled data generated could influence experiment results, we executed each experiment 10 times and get 10 classification accuracies and F1 scores (executed the previous two steps 10 times).
4. Finally, the reported classification accuracy and F1 score were the average of these 10 classification accuracies F1 scores.

In the following, the experimental results on each dataset will be presented. Table 4 shows the comparisons of each filter on the Vote dataset. This table consists of five columns. The first four columns correspond to four noise ratios (30, 35, 40, and 45 %). The last column in Table 4 represents the average performance of each filter based on all the noise ratios. For each filter under each noise ratio, the experimental results are shown which includes two

parts: classification accuracy and F1 score. They are obtained by classifying a test dataset based on the noisy training dataset that cleaned by a filter. For example, for CF algorithm and 30 % noise ratio, the experiment result is 84.8 %/81.7 %. It means the classification accuracy on the test dataset is 84.8 %, while the F1 score is 81.7 %.

After obtaining nearly noise-free data, we can either use majority vote or consensus vote idea to identify noises in the remaining data. In Table 4, OurMethod1 uses consensus vote, while OurMethod2 uses majority vote.

Table 4 shows that for all the four noise ratios, our two approaches make higher classification accuracies and F1 scores. Compared to other methods, the advantage of our approach is more obvious when the noise ratio is high.

Tables 5, 6, 7, 8, 9, 10, 11, 12, 13 and 14 compares the performances of each filters based on other datasets. The results show that our proposed approaches are better than others with respect to both classification accuracies and F1 scores.

The average experimental results on all the 11 datasets are summarized in Table 15. It clearly shows that our proposed two approaches make higher classification accuracies and higher F1 scores. The good performances of our approaches are determined by the nearly noise-free data extraction method. By using our method, only few mislabeled data are included in the nearly noise-free (NNF) dataset. Therefore, the classifiers trained based on NNF are more reliable compared to existing approaches which directly use noisy data to train classifiers. Not only with

Table 4 Performance evaluation on Vote

Methods	30 % noise	35 % noise	40 % noise	45 % noise	Ave. (Accu/F1)
CF	84.8 %/81.7 %	78.4 %/74.6 %	74.1 %/69.1 %	66.4 %/59.1 %	75.9 %/71.1 %
MF	89.4 %/87.2 %	74.5 %/81.4 %	78.0 %/73.8 %	68.8 %/59.1 %	77.7 %/75.4 %
CFMF	86.1 %/83.0 %	80.1 %/76.5 %	75.9 %/71.9 %	65.8 %/56.2 %	77.0 %/71.9 %
MFMF	90.4 %/88.3 %	88.6 %/86.4 %	84.3 %/81.8 %	75.8 %/66.8 %	84.8 %/80.8 %
OurMethod1	91.4 %/89.2 %	88.9 %/86.5 %	89.0 %/86.6 %	80.4 %/73.2 %	87.4 %/83.9 %
OurMethod2	90.6 %/88.4 %	89.4 %/87.0 %	89.2 %/87.0 %	83.9 %/78.5 %	88.3 %/85.2 %

Table 5 Performance evaluation on Australian

Methods	30 % noise	35 % noise	40 % noise	45 % noise	Ave. (Accu/F1)
CF	71.7 %/75.1 %	68.7 %/72.0 %	61.0 %/65.1 %	59.3 %/61.8 %	65.2 %/68.5 %
MF	76.3 %/79.9 %	73.7 %/77.2 %	68.4 %/72.8 %	62.0 %/64.9 %	70.1 %/73.7 %
CFMF	71.7 %/75.1 %	69.9 %/73.2 %	63.4 %/67.6 %	57.1 %/60.3 %	65.5 %/69.1 %
MFMF	81.3 %/84.4 %	77.6 %/81.1 %	72.3 %/77.4 %	62.5 %/64.8 %	73.4 %/76.9 %
OurMethod1	78.4 %/82.2 %	78.2 %/81.4 %	73.4 %/78.2 %	64.9 %/67.1 %	73.7 %/77.2 %
OurMethod2	80.7 %/84.3 %	78.7 %/82.7 %	74.6 %/79.6 %	65.1 %/67.0 %	74.8 %/78.4 %

Table 6 Performance evaluation on Heart

Methods	30 % noise	35 % noise	40 % noise	45 % noise	Ave. (Accu/F1)
CF	71.3 %/73.6 %	67.7 %/70.2 %	64.5 %/66.5 %	58.2 %/59.3 %	65.4 %/67.4 %
MF	76.2 %/79.0 %	72.4 %/76.0 %	71.4 %/73.4 %	61.9 %/64.0 %	70.5 %/73.1 %
CFMF	72.7 %/75.7 %	66.0 %/69.4 %	62.8 %/65.0 %	57.7 %/59.0 %	64.8 %/67.3 %
MFMF	79.0 %/81.6 %	74.2 %/77.3 %	77.5 %/79.7 %	62.3 %/64.6 %	73.3 %/75.8 %
OurMethod1	79.8 %/82.1 %	73.6 %/76.5 %	71.9 %/74.9 %	61.9 %/64.5 %	71.8 %/74.5 %
OurMethod2	80.5 %/83.0 %	76.7 %/79.6 %	76.9 %/79.4 %	66.3 %/67.5 %	75.1 %/77.4 %

Table 7 Performance evaluation on Indian

Methods	30 % noise	35 % noise	40 % noise	45 % noise	Ave. (Accu/F1)
CF	59.2 %/68.7 %	58.5 %/68.0 %	52.5 %/62.0 %	55.0 %/64.4 %	56.7 %/65.8 %
MF	61.4 %/71.0 %	59.5 %/69.3 %	60.7 %/69.6 %	55.6 %/63.8 %	59.2 %/68.4 %
CFMF	60.4 %/69.9 %	60.0 %/70.3 %	54.8 %/64.5 %	54.4 %/63.2 %	56.4 %/67.0 %
MFMF	64.4 %/73.6 %	61.7 %/72.5 %	62.7 %/70.1 %	52.2 %/58.1 %	60.3 %/68.6 %
OurMethod1	65.4 %/75.2 %	66.9 %/77.4 %	62.8 %/71.0 %	55.0 %/62.5 %	62.5 %/71.5 %
OurMethod2	64.9 %/74.5 %	66.2 %/75.5 %	64.1 %/70.3 %	53.9 %/59.8 %	62.3 %/70.0 %

Table 8 Performance evaluation on Horse

Methods	30 % noise	35 % noise	40 % noise	45 % noise	Ave. (Accu/F1)
CF	68.6 %/73.9 %	58.4 %/62.7 %	60.0 %/66.4 %	56.6 %/62.0 %	60.9 %/66.3 %
MF	70.5 %/75.7 %	62.6 %/66.5 %	59.9 %/65.8 %	58.3 %/63.3 %	62.8 %/67.8 %
CFMF	69.1 %/74.8 %	61.1 %/65.2 %	62.7 %/68.2 %	56.8 %/61.8 %	62.4 %/67.5 %
MFMF	75.2 %/80.1 %	65.3 %/70.0 %	66.1 %/72.0 %	59.5 %/64.1 %	66.5 %/71.6 %
OurMethod1	73.6 %/78.3 %	64.4 %/70.1 %	63.3 %/70.7 %	58.4 %/65.4 %	64.9 %/71.1 %
OurMethod2	74.4 %/79.2 %	64.5 %/70.6 %	67.4 %/75.9 %	61.2 %/67.7 %	66.9 %/73.4 %

Table 9 Performance evaluation on Credit

Methods	30 % noise	35 % noise	40 % noise	45 % noise	Ave. (Accu/F1)
CF	70.0 %/66.0 %	63.3 %/56.2 %	60.5 %/57.2 %	54.4 %/50.8 %	62.1 %/57.6 %
MF	75.7 %/71.5 %	70.1 %/62.1 %	65.2 %/61.4 %	59.2 %/56.4 %	67.6 %/62.9 %
CFMF	71.6 %/67.1 %	65.8 %/59.1 %	59.9 %/56.9 %	54.3 %/50.4 %	62.9 %/58.4 %
MFMF	78.5 %/74.9 %	73.7 %/64.7 %	70.1 %/66.3 %	63.1 %/58.7 %	71.4 %/66.2 %
OurMethod1	79.1 %/74.9 %	75.2 %/66.3 %	70.9 %/67.4 %	62.5 %/58.1 %	71.9 %/66.7 %
OurMethod2	80.9 %/76.9 %	77.0 %/68.3 %	72.1 %/68.2 %	63.6 %/59.6 %	73.4 %/68.3 %

Table 10 Performance evaluation on Spect

Methods	30 % noise	35 % noise	40 % noise	45 % noise	Ave. (Accu/F1)
CF	68.3 %/77.3 %	63.4 %/72.3 %	61.0 %/71.8 %	52.6 %/62.8 %	61.3 %/71.1 %
MF	69.4 %/77.8 %	69.2 %/77.3 %	63.4 %/73.8 %	55.2 %/64.3 %	64.3 %/73.3 %
CFMF	70.5 %/79.0 %	65.2 %/74.4 %	61.6 %/72.0 %	55.2 %/66.0 %	63.1 %/72.9 %
MFMF	69.8 %/78.5 %	70.9 %/79.7 %	63.2 %/72.5 %	54.6 %/65.2 %	64.6 %/74.0 %
OurMethod1	71.9 %/80.2 %	75.3 %/82.7 %	64.6 %/74.4 %	56.9 %/65.4 %	67.2 %/75.7 %
OurMethod2	72.7 %/80.7 %	74.0 %/82.1 %	70.0 %/78.5 %	61.2 %/69.0 %	69.5 %/77.6 %

Table 11 Performance evaluation on Spect1

Methods	30 % noise	35 % noise	40 % noise	45 % noise	Ave. (Accu/F1)
CF	61.0 %/72.5 %	62.1 %/73.2 %	56.9 %/68.8 %	58.0 %/70.3 %	59.5 %/71.2 %
MF	61.2 %/73.1 %	69.2 %/79.7 %	63.4 %/75.3 %	60.8 %/72.9 %	63.7 %/75.3 %
CFMF	64.0 %/74.5 %	63.4 %/74.5 %	57.8 %/69.8 %	58.8 %/70.4 %	61.0 %/72.3 %
MFMF	67.7 %/77.8 %	68.9 %/79.6 %	66.2 %/76.5 %	64.4 %/75.9 %	66.8 %/77.5 %
OurMethod1	77.0 %/85.4 %	69.3 %/79.4 %	75.7 %/84.6 %	67.2 %/78.2 %	72.3 %/81.9 %
OurMethod2	75.7 %/83.8 %	72.8 %/82.5 %	75.5 %/84.6 %	69.1 %/80.1 %	73.3 %/82.8 %

Table 12 Performance evaluation on Wine

Methods	30 % noise	35 % noise	40 % noise	45 % noise	Ave. (Accu/F1)
CF	89.5 %/86.5 %	89.0 %/85.5 %	83.7 %/79.3 %	73.8 %/69.2 %	84.0 %/80.1 %
MF	94.3 %/92.2 %	91.8 %/88.9 %	90.4 %/87.1 %	81.1 %/77.9 %	88.6 %/86.5 %
CFMF	91.5 %/89.0 %	88.7 %/84.8 %	85.6 %/81.2 %	81.4 %/76.2 %	81.4 %/82.8 %
MFMF	93.8 %/92.0 %	92.7 %/90.2 %	92.4 %/89.6 %	87.0 %/84.9 %	91.5 %/89.2 %
OurMethod1	95.4 %/93.8 %	94.3 %/92.1 %	92.9 %/89.9 %	91.8 %/87.9 %	93.6 %/90.9 %
OurMethod2	96.6 %/95.2 %	94.0 %/92.2 %	93.0 %/90.2 %	90.1 %/88.7 %	93.2 %/91.6 %

higher average classification accuracy and F1 score, our proposed methods are also robust. Among 11 experimental datasets, our proposed OurMethod1 defeats existing methods in 9 datasets, while OurMethod2 defeats existing methods in all 11 datasets. The experimental datasets include both binary classes and multiple classes. In addition, with the mislabeled ratio increasing, the advantages of our proposed methods are more distinct. Currently the

Table 13 Performance evaluation on Car

Methods	30 % noise	35 % noise	40 % noise	45 % noise	Ave. (Accu/F1)
CF	91.3 %/84.8 %	89.0 %/82.5 %	81.9 %/71.4 %	78.6 %/69.2 %	85.2 %/77.0 %
MF	94.7 %/90.4 %	90.3 %/84.3 %	87.0 %/78.5 %	81.7 %/73.5 %	88.4 %/81.7 %
CFMF	92.3 %/86.9 %	90.8 %/84.3 %	86.9 %/79.2 %	74.6 %/67.3 %	86.12 %/79.4 %
MFMF	90.9 %/85.3 %	90.8 %/84.4 %	84.8 %/76.7 %	77.1 %/69.7 %	85.9 %/79.0 %
OurMethod1	93.6 %/88.5 %	91.5 %/85.6 %	88.6 %/80.3 %	81.7 %/72.4 %	88.9 %/81.7 %
OurMethod2	94.0 %/89.5 %	91.7 %/85.8 %	89.4 %/82.1 %	81.5 %/74.4 %	89.2 %/83.0 %

Table 14 Performance evaluation on Wine_Quality

Methods	30 % noise	35 % noise	40 % noise	45 % noise	Ave. (Accu/F1)
CF	56.8 %/58.0 %	54.4 %/55.1 %	52.8 %/53.5 %	49.4 %/49.9 %	53.4 %/54.1 %
MF	57.8 %/59.6 %	55.1 %/56.2 %	56.5 %/59.5 %	51.2 %/50.8 %	55.2 %/56.5 %
CFMF	58.6 %/59.6 %	56.4 %/57.1 %	54.2 %/55.5 %	50.2 %/51.6 %	54.9 %/56.0 %
MFMF	58.5 %/61.0 %	55.8 %/56.3 %	55.7 %/58.1 %	53.2 %/52.3 %	55.8 %/56.9 %
OurMethod1	59.4 %/61.4 %	56.4 %/57.5 %	54.9 %/57.4 %	52.6 %/52.2 %	55.8 %/57.1 %
OurMethod2	59.2 %/61.5 %	57.3 %/58.7 %	55.4 %/58.4 %	52.8 %/53.3 %	56.2 %/58.0 %

Table 15 Performance evaluation on all the datasets

	Methods					
	CF (%)	MF (%)	CFMF (%)	MFMF (%)	OurMethod1 (%)	OurMethod2 (%)
Average classification accuracy	64.4	69.8	66.9	72.2	73.6	74.8
Average F1 score	68.2	72.2	69.5	74.2	75.7	76.9

maximal mislabeled ratio in the experiments is 0.45. This is enough high because the noise ratio in most practical datasets is hard to exceed this value.

5 Conclusions and future works

Ensemble learning-based filter is widely used to identify mislabeled data from a noisy training dataset. Its basic idea is to divide the training dataset into several subsets; then, each subset is checked by the multiple classifiers which are trained based on other noisy subsets. Because the data that used to train multiple classifiers are noisy, the quality of trained classifiers might be poor, which can result in poor noise identification results.

In this work, a novel approach is proposed. Its key idea is to extract the nearly noise-free data firstly, which then used to train multiple classifiers to identify noises. To enable our approach, a nearly noise-free data extraction method is proposed. This method is designed with tough noise-free data checking rule, which can almost eliminate all the potential noisy data. Experimental results verify that our proposed approach has better noise identification performance compared to existing approaches.

In this work, after extracting nearly noise-free data, we directly use these data to train multiple classifiers, which are then used to identify noises from remaining data. So this is in the scope of supervised learning. In fact, for this problem, semi-supervised learning can also be used. After we extracting nearly noise-free data, the remaining data are potential noisy and their labels might be incorrect. But in fact, although their labels are almost useless, the data are still useful. These data can be regarded as unlabeled data in the semi-supervised learning scenario, which can be used to boost the performance of pure supervised learning. This will be studied in our future work.

Acknowledgments This research was supported by "the Fundamental Research Funds for the Central Universities" No. NS2016089.

References

1. Guan D, Yuan W, Lee YK (2009) Nearest neighbor editing aided by unlabeled data. Inf Sci 179(13):2273–2282
2. Van J, Khoshgoftaar T, Huang H (2007) The pairwise attribute noise detection algorithm. Knowl Inf Syst 11(2):171–190
3. Van J, Khoshgoftaar T (2009) Knowledge discovery from imbalanced and noisy data. Data Knowl Eng 68(12):1513–1542

4. Zhu XQ, Wu XD (2004) Class noise vs. attribute noise: a quantitative study. Artif Intell Rev 22(3):177–210
5. Zhu XQ, Wu XD, Yang Y (2004) Dynamic classifier selection for effective mining from noisy data streams. In: Proceedings of fourth IEEE international conference on data mining, pp 305–312
6. Ma T, Zhou J, Tang M (2015) Social network and tag sources based augmenting collaborative recommender system. IEICE Trans Inf Syst 98(4):902–910
7. Bi Y, Jeske DR (2010) The efficiency of logistic regression compared to normal discriminant analysis under class-conditional classification noise. J Multivar Anal 101(7):1622–1637
8. Nettleton D, Orriols-Puig A, Fornells A (2010) A study of the effect of different types of noise on the precision of supervised learning techniques. Artif Intell Rev 33(4):275–306
9. Zhang J, Yang Y (2003) Robustness of regularized linear classification methods in text categorization. In: Proceedings of the 26th annual international ACM SIGIR conference on research and development in information retrieval, pp 190–197
10. Opitz D, Maclin R (1999) Popular ensemble methods: an empirical study. J Artif Intell Res 11:169–198
11. Dietterich TG (2000) An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization. Mach Learn 40(2):139–157
12. Ratsch G, Onoda T, Muller K (2001) Soft margins for AdaBoost. Mach Learn 42(3):287–320
13. West M et al (2001) Predicting the clinical status of human breast cancer by using gene expression profiles. In: Proceedings of the national academy of sciences, pp 11462–11467
14. Hickey RJ (2006) Noise modelling and evaluating learning from examples. Artif Intell 82(1):157–179
15. Pechenizkiy M, Tsymbal A, Puuronen S, Pechenizkiy O (2006) Class noise and supervised learning in medical domains: the effect of feature extraction. In: Proceedings of 19th IEEE symposium on computer-based medical systems, pp 708–713
16. Bootkrajang J, Kaban A (2013) Classification of mislabelled microarrays using robust sparse logistic regression. Bioinformatics 29(7):870–877
17. Saez J, Galar M, Luengo J, Herrera F (2012) A first study on decomposition strategies with data with class noise using decision trees. Hybrid Artif Intell Syst (Lect Notes Comput Sci) 7209:25–35
18. Beigman E, Klebanov BB (2009) Learning with annotation noise. In: Proceedings of the joint conference of the 47th annual meeting of the ACL and the 4th international joint conference on natural language processing, pp 280–287
19. Sastry PS, Nagendra GD, Manwani N (2010) A team of continuous action learning automata for noise-tolerant learning of half-spaces. IEEE Trans Syst Man Cybern B Cybern 40(1):19–28
20. Manwani N, Sastry PS (2013) Noise tolerance under risk minimization. IEEE Trans Cybern 43(3):1146–1151
21. Abellan J, Masegosa AR (2010) Bagging decision trees on data sets with classification noise. In: Proceedings of the 6th international conference on foundations of information and knowledge systems, pp 248–265
22. Abellan J, Moral S (2003) Building classification trees using the total uncertainty criterion. Int J Intell Syst 18(12):1215–1225
23. Brodley CE, Friedl MA (1996) Improving automated land cover mapping by identifying and eliminating mislabeled observations from training data. In: Proceedings of geoscience and remote sensing symposium, pp 1379–1381
24. Brodley CE, Friedl MA (1999) Identifying mislabeled training data. J Artif Intell Res 11:131–167
25. Chaudhuri BB (1996) A new definition of neighborhood of a point in multi-dimensional space. Pattern Recognit Lett 17:11–17
26. Guan D, Yuan W et al (2011) Identifying mislabeled training data with the aid of unlabeled data. Appl Intell 35(3):345–358
27. John GH (1995) Robust decision trees: removing outliers from databases. In: Proceeding of international conference on knowledge discovery and data mining, pp 174–179
28. Marques AI et al (1876) Decontamination of training data for supevised pattern recognition. Adv Pattern Recognit Lect Notes Comput Sci 2000:621–630
29. Marques AI et al (2003) Analysis of new techniques to obtain quality training sets. Pattern Recognit Lett 24:1015–1022
30. Metxas et al (2004) Distinguishing mislabeled data from correctly labeled data in classifier design. In: Proceedings of 16th IEEE international conference on tools with artificial intelligence, pp 668–672
31. Verbaeten S, Assche, AV (2003) Ensemble methods for noise elimination in classification problems. In: Proceeding of 4th international workshop on multiple classifier systems, pp 317–325
32. Wilson DL (1992) Asymptotic properties of nearest neighbor rules using edited data. IEEE Trans Syst Man Cybern 2(3):431–433
33. Wu X, Zhu X, Chen Q (2003) Eliminating class noise in large datasets. In: Proceeding of international conference on machine learning, pp 920–927
34. Young J, Ashburner J, Ourselin S (2013) Wrapper methods to correct mislabeled training data. In: Proceedings of the 3rd international workshop on pattern recognition in neuroimaging, pp 170–173
35. Zhou ZH, Jiang Y (2004) Editing training data for kNN classifiers with neural network ensemble. Lect Notes Comput Sci 3173:356–361
36. Gu B, Sheng VS, Tay KY et al (2015) Incremental support vector learning for ordinal regression. IEEE Trans Neural Netw Learn Syst 26(7):1403–1416
37. Gu B, Sheng VS (2016) A robust regularization path algorithm for-support vector classification. IEEE Trans Neural Netw Learn Syst. doi:10.1109/TNNLS.2016.2527796
38. Gu B, Sun XM, Sheng VS (2016) Structural Minimax Probability Machine. IEEE Trans Neural Netw Learn Syst. doi:10.1109/TNNLS.2016.2544779
39. Gu B, Sheng VS, Wang Z et al (2015) Incremental learning for-support vector regression. Neural Netw 67:140–150
40. Wen X, Shao L, Xue Y et al (2015) A rapid learning algorithm for vehicle classification. Inf Sci 295:395–406
41. Yuan W, Guan D, Shen L et al (2014) An empirical study of filter-based feature selection algorithms using noisy training data. In: Proceedings of the 4th IEEE international conference on information science and technology, pp 209–212
42. Guan D et al (2014) Detecting potential labeling errors for bioinformatics by multiple voting. Knowl Based Syst 66:28–35
43. Nicholson B, Zhang J, Sheng VS (2015) Label noise correction methods. In: Proceedings of 2015 IEEE international conference on data science and advanced analytics, pp 1–9
44. Frenay B, Verleysen M (2014) Classification in the presence of label noise: a survey. IEEE Trans Neural Netw Learn Syst 25(5):845–869
45. Triguero I, Saez JA, Luengo J (2014) On the characterization of noise filters for self-training semi-supervised in nearest neighbor classification. Neurocomputing 132:30–41