A Survey of Mislabeled Training Data Detection Techniques for Pattern Classification

Donghai Guan[1,2] and Weiwei Yuan[1,3]

[1]College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, [2]College of Automation, [3]College of Computer Science and Technology, Harbin Engineering University, Harbin, China

Abstract

Pattern classification is an important part of machine learning. To use it, a classifier is trained on the training data and then predicts the label for the future unseen data. To obtain a classifier with good performance, the quality of the training data plays an important role. Unfortunately in many areas, it is difficult to provide absolutely clean data. This paper focuses on mislabeled data, which is one of the main types of noisy data. A number of mislabeled data detection techniques have been proposed; however, there is no survey work to summarize those techniques. This paper reviews the existing studies and classifies them into three types: Local learning-based, ensemble learning-based, and single learning-based methods. The technical details, advantages, and disadvantages of these methods are discussed.

Keywords

Ensemble learning, Local learning, Mislabeled data detection.

1. Introduction

Classification is an important type of machine learning technique. In classification, training examples are required in order to train the classifier, and these examples aim to maximize the classification accuracy for unobserved instances. When the algorithm is given, the quality of the classifier will depend heavily on the quality of the training data. This is because the trained classifier corresponds to one of the hypotheses in the hypothesis space. The training data are used to search for the optimal hypothesis in the space. If the quality of the training data is poor, then the search for optimal hypothesis tends to fail.

The quality of the training data is affected by many factors, among which the data source is one of the most crucial. Unfortunately, due to the nature of data sources, noise is commonly included in the data, e.g., in the data entry and acquisition phases [1,2].

Noisy training data can arise due to two reasons: Noisy features and noisy labels. Noisy features means that the values of the features of some training examples are incorrect. Noisy labels means that some of the training examples are mislabeled. Many studies have shown that both these types of noise tend to degrade the learning performance.

In this work, we mainly focus on the noisy label case. One existing study [3] has shown that even a small amount of mislabeled data could dramatically degrade the performance of the obtained classifier. Therefore, the appropriate handling of mislabeled training examples is important.

According to Brodley and Friedl [4,5], mislabeling exists due to various reasons, including the subjective nature of the labeling task, data entry error, and the lack of information to determine the true label of a given example. A subjective labeling error may arise if some experts disagree with the general consensus (e.g., a disease severity ranking). Data entry is another potential cause for mislabeling. A third cause of labeling errors arises when there is a shortage of information to determine the label. For example, in the medical domain, a physician might fail to make a 100% accurate diagnosis if some expensive medical procedures are not available.

The studies that are related to mislabeled data have covered many various domains. For instance, Umaa *et al*. [6] proposed the filtering of mislabeled training examples to improve the onboard analysis of Hyperion images. Malossini *et al*. [7] proposed the filtering of labeling errors in microarrays. Berthelsen *et al*. [8] detected mislabeled data in PoS-tagged corpora. Esuli *et al*. [9] proposed a technique to clean training data for text classification. Khoshgoftaar *et al*. [10] filtered mislabeled data to enhance the software quality estimation. Sculley *et al*. [11] filtered mislabeled spam emails to improve the performance of a spam filtering system.

There are two main methods for handling mislabeled data. The first method enhances the existing learning

algorithms to make them noise tolerant. The second method detects and filters the potential mislabeled data before learning by using different strategies. Then the processed training examples are used to train the classifiers. Although each of these methods has its own advantages, most researchers favor the second method. As pointed out in [12], when the noise detection and hypothesis formation are separated, the induced hypothesis tends to be less complex and more accurate. For this reason, in this work, we focus on the second method, mislabeled training data detection.

Existing studies use different strategies to determine the mislabeled example. According to the employed strategies, we divide the existing studies into three categories:

1. Local learning-based methods: In these methods, if the class label of one example is different from the class labels of its surrounding neighbors, this example is regarded as potentially mislabeled data. The main research issues in this method are how to determine the neighborhood and how to define the searching strategy.
2. Ensemble learning-based methods: In these methods, multiple classifiers are employed to detect mislabeled data. They assume that multiple classifiers tend to generate conflicting class labels for the mislabeled examples. The main research issue of this method is in determining how to construct different classifiers.
3. Single learning-based methods: Instead of using local learning (mainly the nearest neighbor algorithm) and ensemble learning, there are some methods that are based on a single classifier, like a decision tree or neural network.

Although the importance and necessity of detecting mislabeled training examples have been recognized, there is no paper that reviews all the different types of methods mentioned above. To promote the development of this topic, a comprehensive review is necessary. In our review, each of these different types of methods will be discussed. In addition, the methods are compared and discussed.

2. Local Learning-based Methods

Local learning-based methods assume that the examples that are close to each other tend to have the same class label. If the class label of one instance differs from those of its neighborhood instances, this instance is regarded as mislabeled.

Existing local learning-based methods differ from the methods used to detect mislabeled data. In this section, some representative studies will be introduced. In essence, most local learning-based methods follow the generalized scheme shown in Table 1.

1. Edited nearest neighbor (ENN): This algorithm [Table 2] was proposed by Wilson [13,14]. It removes all instances that have been misclassified by the k-NN rule from the training set. This method is the basis for many other related methods.
2. Repeated-ENN (RENN): RENN [13] applies the ENN algorithm repeatedly until all remaining instances have a majority of their neighbors with the same class, which continues to widen the gap between the classes and to smooth the decision boundary of ENN. The RENN algorithm is shown in Table 3.
3. All k-NN (ANN): The ANN algorithm [15] is similar to the iterative ENN except that the value of k is increased after each iteration. Its algorithm is given in Table 4.
4. Multiedit [16]: This method [Table 5] was proposed because Wilson's ENN is inconsistent with its assumption of statistical independence. This is because

Table 1: Generalized scheme to detect mislabeled training data

Let T be the initial training examples. T_n is the training set after filtering mislabeled data. δ, ε, and σ are the classification rule, error estimator, and stopping criterion, respectively

1. Using the (error-counting) estimator ε, obtain an estimate of the classification error for the rule δ training with the set T. Let S be the set of prototypes misclassified in this process
2. $T_n = T$
3. $T_n = T_n \backslash S$
4. If σ then STOP, else go to 1

Table 2: The algorithm of edited nearest neighbor

1. Let $T_n = T$ // T is the original training set, and T_n is the edited set after filtering noises
2. For each $x_i \in T_n$, do
   Discard x_i from T_n if it is misclassified using the k-NN rule with prototypes in $T_n / \{x_i\}$

Table 3: The algorithm of repeated edited nearest neighbor

1. Let $T_n = T$
   REPEAT
2. At iteration t, for each $x_i \in T_e^t$, do
   Discard x_i from $x_i \in T_e^t$ if it is misclassified using the k-NN rule with prototypes in $T_e^t / \{x_i\}$;
   UNTIL $T_e^t = T_e^{t-1}$ // T_e^t and T_e^{t-1} denote the edited data set of T at iterations t and $t-1$, respectively

Table 4: The algorithm of all k-NN

1. Let $T_n = T$
2. For each $x_i \in T_n$, do:
   2.1 Set $m=1$
   2.2 While $m < k+1$ do:
   2.2.1 Discard x_i from T_n if it is misclassified using the m-NN rule with prototypes in $T_n / \{x_i\}$ go to step 2
   2.2.2 Set $m = m+1$

the training examples are alternatively used as both testing and training data.
To achieve statistical independence, the examination of the training examples can be performed in a Holdout manner. In the ideal case, we need to obtain the testing and training data independently. Although the ideal case is hard to satisfy in reality, it has been proven that the use the idea of Holdout is asymptotically optimal in the Bayes sense.

5. Modified ENN (MENN) [17]: This algorithm is an extension of ENN. It removes the example if its label differs from the predicted label of its k + l nearest neighbors. Here, the additional l neighbors are the instances in the training set that have the same distance as the last neighbor of x_i. The algorithm of MENN is shown in Table 6.
6. Nearest centroid neighbor edition [18]: In the nearest centroid neighborhood (NCN) concept [19,20], the neighborhood is defined by two factors: The proximity of instances to a given training example and the symmetrical distribution around it. The NCN neighborhood makes use of the information of the geometrical distribution to obtain better classification. The algorithm is shown in Table 7. In this algorithm, the centroid of a set of points $X = \{x_1, x_2, ...x_r\}$ can be defined as $x_r^c = \frac{1}{r}\sum_{i=1}^{r} x_i$.
7. Relative neighborhood graph edition [21]: Although there are many mislabeled data detection methods, most existing studies suffer from certain limitations. Especially when the number of training examples is not large enough, many of the existing methods are no longer optimal. Therefore, some alternative schemes are needed. This method uses the concept of a proximity graph to detect mislabeled examples.

Let $T = \{x_1, x_2, ..., x_n\}$ be a set of training examples, where n is the number of training data. A proximity graph (PG), G = (V, E), is an undirected graph with a set of vertices, $V = T$, and a set of edges, E, such that $(x_i, x_j) \in E$ if and only if x_i and x_j satisfy some neighborhood relation. In this case, x_i and x_j are regarded as graph neighbors. The graph neighbors of a given training example constitute its graph neighborhood. The graph neighborhood of a subset, $S \subseteq V$, consists of the union of all of the graph neighbors of every node in S. The proposed scheme can be expressed in the following way: (a) Constructing the corresponding PG and (b) discarding those examples that are misclassified by their graph neighbors.

In addition to the above methods, some other related methods can be found in [22-27]. In essence, they are similar to the methods discussed above.

3. Ensemble Learning-based Methods

In recent years, more researchers have recognized the importance of ensemble learning-based approaches for mislabeled data detection [5,28,29]. In machine learning, it has been proven that the use of ensemble learning tends to provide superior performance if the following two conditions are satisfied: (1) The errors made by the base models are independent of each other and (2) the error rates of the base models are less than 50% [30].

Ensemble learning-based methods define an example as mislabeled if its label differs from its predicted label by multiple independent classifiers. Algorithms that belong to this category vary in terms of how the different classifiers are constructed. We divide these algorithms into two main classes: Classification algorithms variation and data variation.

To apply ensemble-based methods for noise detection, it is necessary to generate variations among various classifiers. For this purpose, classification algorithms variation means that different kinds of algorithms are used. For example, with the same training examples, the decision tree, neural network, and k-nearest neighbor algorithms can generate three different classifiers due to the algorithm variation. On the other hand, data variation will manipulate the training data to generate different training sets for each classifier. In this case, even with the same

Table 5: The algorithm of multiedit

1. Let $T_n = \varnothing$ and randomly split T into B blocks, $T_1, T_2, ..., T_B (B > 2)$
2. For $i = 1, 2, ..., B$
   Add to T_n the prototypes from T_i that are misclassified using the k-NN rule with $T_{(i+1) \bmod B}$
3. If $T_n = T$ during the last i iterations, STOP
4. Let $T = T_n$, go to 1

Table 6: The algorithm of modified edited nearest neighbor

1. Let $T_n = T$
2. For each $x_i \in T_n$, do:
   2.1 Let $V(k, x_i)$ be the set of the k-nearest neighbors of x_i, $d_j(x_i)$ be the distance from x_i to its j^{th} nearest neighbor and assume $d_1(x_i) \le d_2(x_i) \le ... \le d_k(x_i)$
   2.2 If l different labeled samples in T satisfy the condition that the distance from x_i to each of them is equal to $d_k(x_i)$, then include these l labeled examples in $V(k, x_i)$
   2.3 Discard x_i from T_n if it is misclassified using the k-NN rule with prototypes in $V(k, x_i)$

Table 7: The algorithm of nearest centroid neighbor edition

1. Let $T_n = T$
2. For each $x_i \in T_n$, do
   2.1 Search for the k nearest centroid neighbors of x_i. Represent them by $NCN_1(x_i)$, $NCN_2(x_i)$, ..., $NCN_k(x_i)$. Note that $NCN_1(x_i) = NN(x_i)$; $NCN_j(x_i)$ is the centroid of this and the previously selected NCN
   2.2 Discard x_i from T_n if it does not agree with the label defined by the k nearest centroid neighbors

type of classification algorithm, due to the training data variation, multiple various classifiers can be generated.

3.1 Classification Algorithms Variation

Majority filtering (MF) and consensus filtering (CF) [4,5] are the most popular classification algorithms variation techniques.

The general idea of MF and CF is that they train a set of classifiers based on different learning algorithms. Then these multiple classifiers will be used to classify the training examples. For each example, if its predicted label is different from its given label, the example is tagged as mislabeled. For MF, if more than half of the total classifiers misclassify one instance, then this instance is regarded as mislabeled data. CF is more conservative, so the instance is tagged as mislabeled only when all of the classifiers misclassify the instance. The reason for employing ensemble classifiers in MF and CF is that the ensemble classifier's decision is generally more reliable than that of each individual classifier.

As shown in Figure 1, MF initially divides the training set E into n equal-sized disjoint subsets (step 1). The main loop (steps 3-16) 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 is then used to train a classifier H_j based on learning algorithm A_j (step 6). As shown in steps 5 and 6, due to the variations of the learning algorithms, multiple various classifiers will be trained. Then these classifiers will predict the labels for the instances in E_i. If the majority of these classifiers do not agree with an instance's given label, then this instance is regarded as mislabeled data (step 14).

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 subsets of E_i, where $\bigcup_i E_i = E$
(2) $A \leftarrow \varnothing$
(3) for i=1,…,n do
(4) form $E_t \leftarrow E \setminus E_i$
(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

Figure 1: The majority filtering algorithm.

The CF algorithm differs from MF at step 14. In CF, the example in E_i is regarded as a noisy example only when all the classifiers do not agree with its class label.

3.2 Data Variation

Data variation does not need to use multiple various learning algorithms. Instead, it only utilizes one type of learning algorithm. To generate the diversity among the classifiers, it will manipulate the training set and provide different training data to each learning algorithm. The data variation methods are divided into four types: Cross-validated committees, bagging, boosting, and subspace change. The general scheme of these four types is shown in Table 8. The main difference is how each one provides different sets of training data (step 3.3).

1. Cross-validated committees [31]: In this method, the training set T will be randomly divided into n equal-sized disjoint subsets. For example, in step 3.3 of Table 8, suppose that there are 12 training examples (from $T1$ to $T12$). If we plan to use three classifiers, then the training data set for each classifier might be: Training set 1 (T1, T5, T8, T12); Training set 2 (T2, T3, T7, T11); Training set 3 (T4, T6, T9, T10).
2. Bagging: This method is based on bootstrapping [32,33] and generates several training sets. Each training set is established from the original training set by sampling with replacement. For example, the three training sets might be: Training set 1 (T1, T2, T3, T3, T5, T6, T6, T8, T9, T10, T11, T12); Training set 2 (T1, T2, T3, T4, T5, T6, T7, T7, T9, T10, T11, T11); Training set 3 (T1, T2, T2, T4, T5, T5, T7, T8, T9, T10, T11, T12).
3. Boosting: Boosting was proposed by Freund and Schapire [34]. In boosting, the training sets are obtained in a deterministic way. The training sets are obtained sequentially in the algorithm. This is different from bagging, where the training sets are obtained randomly. At each step of boosting, the training data are reweighted so that the incorrectly classified ex-

Table 8: Ensemble-based mislabeled data detection by data variation

1. Let $T_n = \varnothing$
2. Randomly divide training set T into n equal-sized disjoint crosses
3. For i=1:n
3.1 Take T_i as the data to be examined
3.2 Take T/T_i as the training data
3.3 Train k different classifiers by generating k different sets of training data based on T/T_i
3.4 For each instance x in T_i, if its label differs from its predicted label of k classifiers (usually by voting), then $T_n = T_n \cup x$

amples get large weights in a new modified training set. AdaBoost is one of the most popular boosting techniques. A noise detection based AdaBoost algorithm has been proposed in [35].

4. Subspace change [36]: In this method, instead of training classifiers on different subsets of the data (like in bagging), multiple classifiers will be trained on the different parts of the data. For example, each training example might include nine features. We can randomly pick three features to train each classifier. To guarantee both diversity and discrimination, some techniques, such as random space, can be used.

4. Single Learning-based Methods

In this section, the single learning-based methods are discussed. Since local learning is based on the nearest neighbor, the nearest neighbor is not covered here. Based on the popularity of the algorithms, we discuss two methods: Decision trees and neural networks.

4.1 Decision Tree

A decision tree provides a graphical representation of a classification model. The internal node is the attribute, and the leaf node is the predicted class label. A decision tree usually consists of an iterative top-down procedure of selecting the best attribute to label an internal node of the tree. To search for the best attribute, different decision tree algorithms have used various metrics.

C4.5 [37] is one of the most popular decision tree algorithms. It employs an entropy-based criterion to select the best attribute to create a node. After inducing a tree, we need to prune the tree to make it simpler. We need a procedure for choosing one of the many subtrees to prune. After repeatedly pruning or not pruning nodes, we obtain the final tree.

The robust-C4.5 algorithm can detect mislabeled training data based on the conventional C4.5 algorithm. Its algorithm is shown in Table 9.

4.2 Neural Network

Automatic noise reduction (ANR) has been proposed to identify and remove noisy training examples, the classes of which have been mislabeled [38]. The underlying mechanism behind ANR is based on a framework of multi-layer artificial neural networks.

ANR assigns each example a class probability vector to represent its soft class label. Initially the vector equals the original class label and can be modified during training. When the noise level is not very high (less than 30%), the clean data can contribute more to determine the architecture of the network. Thus, its output can be referred to correct mislabeled data by aligning the class probability vector with the network output.

The class probability will be updated based on its difference from the network output. For a mislabeled class, the probability will become smaller, while it will become larger for the correct class. Eventually, the mislabeled data's label will be changed to the correct label after enough training. After training, those data whose classes have been relabeled are then treated as noisy data and are removed from the data set.

We have summarized existing mislabeled data detection methods into three types: Local learning-based method, ensemble learning-based method, and single learning-based method.

As shown in Sections 2-4, different types of methods have their own advantages and disadvantages. There is no best method since it depends on the characteristics of data and the user's focus. To provide a reference for the researchers, in Table 10, we summarize the advantages/disadvantages for each method.

Although in this review we have tried to cover as many algorithms as we can, there are some methods that are

Table 9: The robust-C4.5 algorithm

1. Let $T_n = T$ // T is the original training set, T_n is the set after filtering noisy examples

   REPEAT

2. At iteration t, build a tree based on T_n^t
3. Prune the tree
4. For each example x in T_n^t, if it is misclassified by the pruned tree, remove it ($T_n^t = T_n^t - x$)

   UNTIL $T_n^t = T_n^{t-1}$ // T_n^t and T_n^{t-1} denote the edited data set of T at iterations t and $t-1$, respectively

Table 10: The advantages and disadvantages of each mislabeled data detection method

Method	Advantages	Disadvantages
Local learning-based method	Mostly based on nearest neighbor algorithm; easy to understand and implement	Assume that the examples that are close to each other tend to have the same label; this assumption is not valid for all the data sets
Ensemble learning-based method	The accuracy to detect mislabeled data is usually better than the other two methods as multiple learning algorithms can complement with each other	More time consuming because multiple classifiers need to be trained
Single learning-based method	In terms of efficiency, usually this method is better; it is more suitable for the highly dynamic noisy data	As single learning method is used, the performance has more variance

not included, as they do not belong to the three types we discussed above or they are not general purpose methods. For example, Guan *et al*. proposed the semi-supervised method for mislabeled detection [39,40]. Qiu *et al*. proposed the spectral analysis based method [41]. Umaa *et al*. proposed the instance weighting scheme [42]. Saez *et al*. proposed the rule set based noise detection scheme [43]. Zhang *et al*. proposed the data perturbation based method [44].

Moreover, in recent years, the study of mislabeled training data has been combined with other research issues, like multi-class and imbalanced data classification. For example, Bootkrajang and Saze have studied multi-class classification in the presence of labeling errors [45,46]. Hulse and Khoshgoftaar have studied learning from imbalanced data with labeling errors [47-49].

5. Conclusions

The performance of many classifiers suffers due to mislabeled training data. To solve this problem, one effective solution is to detect and remove the mislabeled training data, after which the filtered data will be used for training.

In this paper, we survey existing works and summarize them into three types: Local learning-based methods, ensemble learning-based methods, and single learning-based methods. For each method, the detailed algorithm and an analysis are presented.

6. Acknowledgment

Dr. Weiwei Yuan is the corresponding author of this paper. This research was supported by National Natural Science Foundation of China (Grant No. 61100007, 61100081). It was also supported by the collaborative research project under NSFC-NRF cooperative Program (Grant No. 613111015).

References

1. X. Wu, "Knowledge Acquisition from Databases," Norwood, New Jersey: Alex Publishing Corp, 1995.
2. K. Orr, "Data quality and systems theory", Communications of the ACM, Vol. 41, no. 2, pp. 66-71, 1998.
3. A. Malossini, E. Blanzieri, and R. Ng, "Assessment of SVM reliability for microarray data analysis," in Proc. 14th Dutch–Belgian Conference on Machine Learning, pp. 1-10, 2005.
4. C.E. Brodley, and M.A. Friedl, "Improving automated land cover mapping by identifying and eliminating mislabeled observations from training data," In Proc. Geoscience and Remote Sensing Symposium, pp. 1379-81, 1996.
5. C.E. Brodley, and M.A. Friedl, "Identifying mislabeled training data," Journal of Artificial Intelligence Research, Vol. 11, pp. 131-67, 1999.
6. U. Rebbapragada, "Improving onboard analysis of Hyperion images by filtering mislabeled training data examples," in Proc. IEEE Aerospace Conference, pp. 1-9, 2009.
7. A. Malossini, E. Blanzieri, and R. Ng, "Detecting potential labeling errors in microarrays by data perturbation," Bioinformatics, Vol. 17, pp. 2114-21, 2006.
8. H. Berthelsen, and B. Megyesi, "Ensemble of Classifiers for Noise Detection in PoS Tagged Corpora," In Proc. of the Third International Workshop on Text, Speech and Dialogue, pp. 27-32, 2000.
9. A. Esuli, and F. Sebastiani, "Training Data Cleaning for Text Classification," In Proc. of the 2nd International Conference on Theory of Information Retrieval: Advances in Information Retrieval Theory, pp. 29-41, 2009.
10. T. Khoshgoftaar, S. Zhong, and V. Joshi, "Enhancing software quality estimation using ensemble-classifier based noise filtering," Intelligent Data Analysis, Vol. 9, no. 1, pp. 3-27, 2005.
11. D. Sculley, and G. V. Cormack, Filtering Spam in the Presence of Noisy User Feedback. Tufts University, 2008.
12. D. Gamberger, N. Lavrac, and S. Dzeroski, "Noise detection and elimination in data preprocessing: Experiments in medical domains," Applied Artificial Intelligence, Vol. 14, pp. 205-23, 2000.
13. D.L. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," IEEE Transactions on Systems, Man and Cybernetics, Vol. 2, no. 3, pp. 408-21, 1972.
14. C. Penrod, and T. Wagner, "Another look at the edited nearest neighbor rule," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 7, pp. 92-4, 1977.
15. I. Tomek, "An experiment with the edited nearest-neighbor rule," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 6, no. 6, pp. 448-52, 1976.
16. F.J. Ferri, J.V. Albert, and E. Vidal, "Consideration about sample-size sensitivity of a family of edited nearest-neighbor rules," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 29, no. 4, pp. 667-72, 1999.
17. K. Hattori, and M. Takahashi, "A new edited k-nearest neighbor rule in the pattern classification problem," Pattern Recognition, Vol. 33, no. 3, pp. 521-8, 2000.
18. J. Sanchez, R. Barandela, A. Marques, R. Alejo, and J. Badenas, "Analysis of new techniques to obtain quality training sets," Pattern Recognition Letters, Vol. 24, no. 7, pp. 1015-22, 2003.
19. B.B. Chaudhuri, "A new definition of neighbourhood of a point in multi-dimensional space," Pattern Recognition Letters, Vol. 17, pp. 11-7, 1996.
20. J.S. Sanchez, F. Pla, and F.J. Ferri, "On the use of neighbourhood based non-parametric classifiers," Pattern Recognition Letters, Vol. 18, pp. 1179-86, 1997.
21. J.S. Sanchez, F. Pla, and F.J. Ferri, "Prototype selection for the nearest neighbour rule through proximity graphs," Pattern Recognition Letters, Vol. 18, pp. 507-13, 1997.
22. N. Jankowski, and M. Grochowski, "Comparison of instances selection algorithms survey," Lecture Notes in Computer Science, Vol. 3070, pp. 598-603, 2004.
23. F. Vazquez, J. Sanchez, and F. Pla, "A stochastic approach to Wilson's editing algorithm," In Proc. of the Second Iberian Conference on Pattern Recognition and Image Analysis, pp. 35-42, 2005.
24. C.E. Brodley, "Addressing the selective superiority problem: Automatic algorithm/model class selection," In Proc. of the Tenth International Machine Learning Conference. MA, pp. 17-24, 1993.
25. D.W. Aha, D. Kibler, and M.K. Albert, "Instance-based learning algorithms," Machine Learning, Vol. 6, pp. 37-66, 1991.
26. D.W. Aha, "Tolerating noisy, irrelevant and novel attributes in instance-based learning algorithms," International Journal of Man-Machine Studies, Vol. 36, pp. 267-87, 1992.
27. D.R. Wilson, and T.R. Martınez, "Instance pruning techniques," Proceedings of the Fourteenth International Conference, Morgan Kaufmann, pp. 403-11, 1997.
28. Z. Zhou, and Y. Jiang, "Editing training data for knn classifiers with neural network ensemble," Lecture Notes in Computer Science, Vol. 3173, pp. 356-61, 2004.

29. X. Wu, X. Zhu, and Q. Chen, "Eliminating class noise in large datasets," In Proc. of International Conference on Machine Learning, pp. 920-7, 2003.
30. L.K. Hansen, and P. Salamon, "Neural Network Ensembles," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 12, no. 10, pp. 993-1001, 1990.
31. S. Verbaeten, and A. Assche, "Ensemble methods for noise elimination in classification problems," In Proc. of the 4th international conference on Multiple classifier systems, pp. 317-25, 2003.
32. L. Breiman, "Bagging predictors," Machine Learning, Vol. 24, no. 2, pp. 123-40, 1996.
33. J. Quinlan, "Bagging, boosting, and C4.5," In Proc. of the Thirteenth National Conference on Artificial Intelligence, pp. 725-30, 1996.
34. Y. Freund, and R.E. Schapire, "Experiments with a new boosting algorithm," In Proc. of International Conference on Machine Learning, pp. 148-56, 1996.
35. J. Cao, S. Kwong, and R. Wang, "A noise-detection based AdaBoost algorithm for mislabeled data," Pattern Recognition, Vol. 45, no. 12, pp. 4451-65, 2012.
36. T. Ho, "The Random Subspace Method for Constructing Decision Forests," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 20, no. 8, pp. 832-44, 1998.
37. J.R. Quinlan, "Programs for Machine Learning," Norwood, New Jersey: Morgan Kaufmann Publishers; 1993.
38. X. Zeng, and T. Martinez, "A noise filtering method using neural networks," In Proc. of International Workshop on Soft Computing Techniques in Instrumentation, Measurement and Related Applications, pp. 26-31, 2003.
39. D. Guan, W. Yuan, Y. Lee, and S. Lee, "Nearest neighbor editing aided by unlabeled data," Information Sciences, Vol. 179, no. 13, pp. 2273-82, 2009.
40. D. Guan, W. Yuan, Y. Lee, and S. Lee, "Identifying mislabeled training data with the aid of unlabeled data," Applied Intelligence, Vol. 35, pp. 345-58, 2011.
41. P. Qiu, and S. Plevritis, "Simultaneous class discovery and classification of microarray data using spectral analysis," Journal of Computational Biology, Vol. 16, no. 7, pp. 935-44, 2009.
42. U. Rebbapragada, and C. Brodley, "Class Noise Mitigation Through Instance Weighting," In Proc. of the 18th European conference on Machine Learning, pp. 708-15, 2007.
43. J. Saez, J. Luengo, and F. Herrera, "Predicting noise filtering efficacy with data complexity measures for nearest neighbor classification," Pattern Recognition, Vol. 46, no. 1, pp. 355-64, 2012.
44. C. Zhang, "Methods for labeling error detection in microarrays based on the effect of data perturbation on the regression model," Bioinformatics, Vol. 25, no. 20, pp. 2708-14, 2009.
45. J. Bootkrajang, and A. Kaban, "Multi-class classification in the presence of labelling errors," In Proc. of the 19th European Symposium on Artificial Neural Networks, 2011.
46. J. Saez, "Analyzing the presence of noise in multi-class problems: Alleviating its influence with the One-vs-One decomposition," Knowledge and Information Systems, DOI 10.1007/s10115-012-0570-1, 2012.
47. J. Hulse, and T. Khoshgoftaar, "Knowledge discovery from imbalanced and noisy data," Data and Knowledge Engineering, Vol. 68, no. 12, pp. 1513-42, 2009.
48. T. Khoshgoftaar, "Supervised Neural Network Modeling: An Empirical Investigation Into Learning From Imbalanced Data With Labeling Errors," IEEE Transactions on Neural Networks, Vol. 21, no. 5, pp. 813-30, 2010.
49. T. Khoshgoftaar, "Comparing Boosting and Bagging Techniques With Noisy and Imbalanced Data," IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, Vol. 41, no. 3, pp. 552-68, 2011.

AUTHORS

Donghai Guan received his Ph. D. degree in Computer Science from Kyung Hee University (KHU), South Korea in 2009. From 2009, he was a postdoctoral student and research professor at KHU. Since February 2011, he has been an assistant professor at Harbin Engineering University, China. Since March 2012, he has been an assistant professor at Kyung Hee University, Korea. His research interests are machine learning and data mining.

E-mail: dhguan@gmail.com

Weiwei Yuan received her Ph. D. degree in Computer Science from Kyung Hee University, South Korea in 2010. Since September 2010, she has been an assistant professor at Harbin Engineering University, China. Since September 2012, she has been a postdoctoral fellow in the Computer Engineering Department at Kyung Hee University. Her research interests are social computing, information security, and machine learning.

E-mail: yuanweiwei00@gmail.com

DOI: 10.4103/0256-4602.125689; Paper No. TR 184_13; Copyright © 2013 by the IETE