Editing Training Data for kNN Classifiers with Neural Network Ensemble

Yuan Jiang and Zhi-Hua Zhou

National Laboratory for Novel Software Technology,
Nanjing University, Nanjing 210093, China
{jiangyuan,zhouzh}@nju.edu.cn

Abstract. Since kNN classifiers are sensitive to outliers and noise contained in the training data set, many approaches have been proposed to edit the training data so that the performance of the classifiers can be improved. In this paper, through detaching the two schemes adopted by the Depuration algorithm, two new editing approaches are derived. Moreover, this paper proposes to use neural network ensemble to edit the training data for kNN classifiers. Experiments show that such an approach is better than the approaches derived from Depuration, while these approaches are better than or comparable to Depuration.

1 Introduction

KNN [5] is one of the most widely used lazy learning approach [1]. Given a set of n training examples, upon receiving a new instance to predict, the kNN classifier will identify k nearest neighboring training examples of the new instance and then assign the class label holding by the most number of neighbors to the new instance.

The asymptotic classification error of kNN tends to the optimal Bayes error rate as $k \to \infty$ and $k/n \to 0$ when n grows to infinity, and the error is bounded by approximately twice the Bayes error if $k = 1$ [6]. This behavior in asymptotic classification performance combining with the simplicity in concept and implementation, makes kNN a powerful classification approach capable of dealing with arbitrarily complex problems, provided there is a large training data set. However, the theoretical behavior can hardly be obtained because kNN is sensitive to outliers and noise contained in the training data set, which usually occurs in real-world applications. Therefore, it is important to eliminate outliers in the training data set and make other necessary cleaning. The approaches devoting to this purpose are referred to as *editing* approaches [6].

During the past years, many editing approaches have been proposed for kNN classifiers [7]. In this paper, the Depuration algorithm [2] is examined and two schemes adopted by it are separated so that two new editing approaches are derived. Experiments show that the effect of Depuration is very close to one new approach will worse than the other, which suggests that the Depuration algorithm has not fully exploit the schemes it adopted. Moreover, this paper proposes to use neural network ensemble to edit the training examples and obtains some success.

The rest of this paper is organized as follows. Section 2 introduces the Depuration algorithm and proposes several new editing approaches. Section 3 presents the experimental results. Section 4 concludes.

2 Editing Approaches

The Depuration algorithm was regarded as the first *prototype selection* approach [9], which consists of removing some "suspicious" training examples while changing the class labels of some other examples. Its purpose is to deal with all types of dirts in the training data set, including outliers, noise and mislabeled examples. The algorithm is based on the *generalised editing* scheme [8] where two parameters, i.e. k and k', have to be set according to $(k+1)/2 \leq k' \leq k$. When k and k' were set to 3 and 2 respectively, Sánchez et al. [9] reported that the Depuration algorithm achieved the best effect among some other editing approaches they compared.

The Depuration algorithm is summarized in Table 1, where X is the original training data set and S is the edited training data set to be returned.

Table 1. The Depuration algorithm

```
Let $S = X$
For each $x_i \in X$ do
    Find $k$ nearest neighbors of $x_i$ in $(X - \{x_i\})$
    If a class label, say $c$, is held by at least $k'$ neighbors
    Then set the label of $x_i$ in $S$ to $c$
    Else remove $x_i$ from $S$
```

Through examining Table 1, it can be found that Depuration implicitly adopts two schemes to edit the training data set. The first scheme is that if there are k' neighbors holding the same class label, then change the class label of the concerned example to the commonly agreed label; otherwise the concerned example is kept as it was. The second scheme is that if there are k' neighbors holding the same class label, then keep the concerned example as it was; otherwise the concerned example is removed. Through separating these two schemes, two new editing approaches can be derived, as shown in Tables 2 and 3. Note that similar to Depuration, both the RelabelOnly approach and the RemoveOnly approach have two parameters, i.e. k and k', have to be set according to $(k+1)/2 \leq k' \leq k$.

Neural network ensemble [12] is a learning technique where multiple neural networks are trained to solve the same problem. Since the generalization ability of a neural network ensemble is usually significantly better than that of a single

Table 2. The RelabelOnly algorithm

```
Let $S = X$
For each $x_i \in X$ do
    Find $k$ nearest neighbors of $x_i$ in $(X - x_i)$
    If a class label, say $c$, is held by at least $k'$ neighbors
    Then set the label of $x_i$ in $S$ to $c$
```

Table 3. The RemoveOnly algorithm

```
Let $S = X$
For each $x_i \in X$ do
    Find $k$ nearest neighbors of $x_i$ in $(X - x_i)$
    If no class label is held by at least $k'$ neighbors
    Then remove $x_i$ from $S$
```

neural network, it has become a hot topic during the past years. Recently, Zhou and Jiang [10] proposed a strong rule learning algorithm through using a neural network ensemble as the preprocess of a rule inducer, and later they showed that using a neural network ensemble to preprocess the training data set could be beneficial when the training data set contains noise and has not captured the whole target distribution [11].

Inspired by these works, here a new editing approach based on neural network ensemble is proposed for kNN classifiers. As shown in Table 4, the NNEE (Neural Network Ensemble Editing) algorithm uses a popular ensemble learning algorithm, i.e. Bagging [4], to construct a neural network ensemble from the original training data set. This trained neural network ensemble is then used to classify the training examples, and the classification is used to replace the original class label of the concerned training example. The NNEE approach has two parameters to set, i.e. the number of neural networks contained in the neural network ensemble and the number of hidden units in the networks, if single-hidden-layered feedforward neural networks are used. Fortunately, our experiments show that the NNEE approach is not sensitive to the setting of these parameters.

Table 4. The NNEE algorithm

```
Let $S = X$
Let NNE = Bagging($X$)
For each $x_i \in X$ do
    change the label of $x_i$ in $S$ to the label predicted by NNE
```

3 Experiments

Ten data sets from the UCI machine learning repository [3] are used in the experiments. Information on these data sets are shown in Table 5.

Table 5. Experimental data sets

Data set	Attribute		Size	Class
	Categorical	Continuous		
annealing	33	5	798	6
credit	9	6	690	2
glass	0	9	214	7
hayes-roth	4	0	132	3
iris	0	4	150	3
liver	0	6	345	2
pima	0	8	768	2
soybean	35	0	683	19
wine	0	13	178	3
zoo	16	0	101	7

On each data set, 10 runs of 10-fold cross validation is performed with random partitions. The effects of the editing approaches described in Section 2 are compared through coupling them with a 3NN classifier. The predictive accuracy of the 3NN classifiers trained on the training data sets edited by different approaches are shown in Table 6, where the values following ± are standard deviations. The parameters of Depuration, RelabelOnly and RemoveOnly are set as $k = 3$ and $k' = 2$. Therefore these algorithms are denoted as Depuration(3,2), RelabelOnly(3,2) and RemoveOnly(3,2), respectively. Five BP networks are contained in the neural network ensemble used by NNEE, and each network has one hidden layer consisting of five hidden units. Therefore here the approach is denoted as NNEE(5,5).

Table 6 shows that the NNEE approach achieves the best editing effect. In detail, it obtains the best performance on seven data sets, i.e. *annealing*, *credit*, *liver*, *pima*, *soybean*, *wine* and *zoo*. RemoveOnly obtains the best performance on three data sets, i.e. *glass*, *hayes-roth* and *wine*. It is surprising that Depuration obtains the best performance on only one data set, i.e. *iris*, as RelabelOnly does. These observations indicate that NNEE is a better editing approach than Depuration. Moreover, since the effect of Depuration is only comparable to that of RelabelOnly, it is obvious that Depuration has not exploited well the power of the two schemes it adopted, especially the scheme used by RemoveOnly. In fact, in some cases such as on *glass*, *hayes-roth*, *soybean* and *zoo*, simultaneously adopting the schemes used by RelabelOnly and RemoveOnly is even worse than adopting any of these schemes. The reason why the phenomenon appearing remains to be explored in the future.

Table 6. Predictive accuracy (%) of 3NN coupled with different editing approaches

Data set	Depuration(3,2)	RelabelOnly(3,2)	RemoveOnly(3,2)	NNEE(5,5)
annealing	89.95 ± 2.81	89.95 ± 2.81	92.76 ± 1.80	**92.81** ± 1.77
credit	84.75 ± 3.95	84.75 ± 3.95	85.33 ± 2.50	**86.20** ± 1.97
glass	59.90 ± 9.29	60.27 ± 9.21	**68.23** ± 5.27	67.94 ± 6.60
hayes-roth	47.81 ± 9.09	48.34 ± 9.23	**54.31** ± 7.89	50.50 ± 9.06
iris	**95.67** ± 4.75	**95.67** ± 4.75	95.20 ± 5.08	95.47 ± 3.25
liver	57.28 ± 7.25	57.28 ± 7.25	61.22 ± 5.11	**64.06** ± 5.27
pima	72.42 ± 4.79	72.42 ± 4.79	74.35 ± 2.77	**75.57** ± 3.04
soybean	87.76 ± 3.40	89.28 ± 3.38	89.58 ± 2.57	**90.87** ± 2.53
wine	94.94 ± 4.26	94.94 ± 4.26	**96.05** ± 2.89	**96.05** ± 2.89
zoo	90.75 ± 6.79	90.95 ± 6.91	93.49 ± 3.87	**94.48** ± 4.47

4 Conclusion

This paper proposes to use neural network ensemble to edit the training data set for kNN classifiers. In detail, a neural network ensemble is trained from the original training data set. Then, the class labels of the training examples are replaced by the labels generated by the neural network ensemble. Experiments show that such an approach could achieve better editing effect than the Depuration algorithm does.

This paper also examines the Depuration algorithm and identifies the two editing schemes it adopted. Through detaching these two schemes, this paper derives two new editing approaches from Depuration, i.e. RelabelOnly and RemoveOnly. Experiments show that the editing effect of Depuration is only comparable to that of RelabelOnly while worse than that of RemoveOnly. This discloses that the scheme of RemoveOnly does not function in the Depuration algorithm. Moreover, in some cases simultaneously using the scheme of RelabelOnly and the scheme of RemoveOnly is even worse than using either of them. Exploring the reason behind these observations is an interesting issue for future work.

Acknowledgement

This work was supported by the National Outstanding Youth Foundation of China under the Grant No. 60325207, the Fok Ying Tung Education Foundation under the Grant No. 91067, the Excellent Young Teachers Program of MOE of China, the Jiangsu Science Foundation Key Project, and the National 973 Fundamental Research Program of China under the Grant No. 2002CB312002.

References

1. Aha, D.W.: Lazy learning: special issue editorial. Artificial Intelligence Review **11** (1997) 7–10

2. Barandela, R., Gasca, E.: Decontamination of training samples for supervised pattern recognition methods. In: Ferri, F.J., Iñesta Quereda, J.M., Amin, A., Paudil, P. (eds.): Lecture Notes in Computer Science, Vol. 1876. Springer, Berlin (2000) 621–630
3. Blake, C., Keogh, E., Merz, C.J.: UCI repository of machine learning databases [http://www.ics.uci.edu/∼mlearn/MLRepository.html], Department of Information and Computer Science, University of California, Irvine, CA (1998)
4. Breiman, L.: Bagging predictors. Machine Learning **24** (1996) 123–140
5. Dasarathy, B.V.: Nearest Neighbor Norms: NN Pattern Classification Techniques. IEEE Computer Society Press, Los Alamitos, CA (1991)
6. Devijver, P.A., Kittler, J.: Pattern Recognition: A Statistical Approach. Prentice Hall, Englewood Cliffs, NJ (1982)
7. Ferri, F.J., Albert, J.V., Vidal, E.: Considerations about sample-size sensitivity of a family of edited nearest-neighbor rules. IEEE Transactions on Systems, Man, and Cybernetics - Part B **29** (1999) 667–672
8. Koplowitz, J., Brown, T.A.: On the relation of performance to editing in nearest neighbor rules. Pattern Recognition **13** (1981) 251–255
9. Sánchez, J.S., Barandela, R., Marqués, A.I., Alejo, R., Badenas, J.: Analysis of new techniques to obtain quality training sets. Pattern Recognition Letters **24** (2003) 1015–1022
10. Zhou, Z.-H., Jiang, Y.: Medical diagnosis with C4.5 rule preceded by artificial neural network ensemble. IEEE Transactions on Information Technology in Biomedicine **7** (2003) 37–42
11. Zhou, Z.-H., Jiang, Y.: NeC4.5: neural ensemble based C4.5. IEEE Transactions on Knowledge and Data Engineering **16** (2004)
12. Zhou, Z.-H., Wu, J., Tang, W.: Ensembling neural networks: many could be better than all. Artificial Intelligence **137** (2002) 239–263