Semi-supervised Nearest Neighbor Editing

Donghai Guan, Weiwei Yuan, Young-Koo Lee and Sungyoung Lee

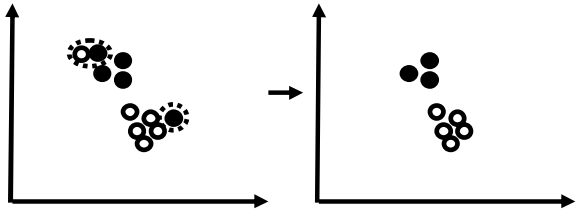
Abstract—**This paper proposes a novel method for data editing. The goal of data editing in instance-based learning is to remove instances from a training set in order to increase the accuracy of a classifier. To the best of our knowledge, although many diverse data editing methods have been proposed, this is the first work which uses semi-supervised learning for data editing. Wilson editing is a popular data editing technique and we implement our approach based on it. Our approach is termed semi-supervised nearest neighbor editing (SSNNE). Our empirical evaluation using 12 UCI datasets shows that SSNNE outperforms KNN and Wilson editing in terms of generalization ability.**

I. Introduction

K-nearest neighbor (KNN) classifier has received considerable attention by the research community. The nearest-neighbor (NN) algorithm and its derivatives have been shown to perform well for pattern recognition in many domains. This classifier consists of finding the k nearest neighbors to each target instance according to a certain dissimilarity measure and making a decision according to the known classification of these k neighbors, usually by assigning the label of the most voted class among these neighbors [1]. When $k = 1$, each instance is assigned to the same class as its nearest neighbor.

Compared with other classifiers, such as multi-layer perceptron and support vector machine, KNN learns more quickly because it need only read in the training set without much further processing. However, since the basic nearest neighbor algorithm stores all of the training instances, it has relatively large memory requirements. It must search through all available instances to classify a new input vector, so it is slow during classification. To produce more time-efficient KNN, much work has been done. For example, several condensing techniques have been proposed that replace the set of training examples T by a smaller set $T_c \subset T$ such that all instances in T are still classified correctly by a KNN classifier that uses T_c.

On the other hand, data editing techniques aim at replacing training set T with a smaller dataset T_e with the goal of improving the accuracy of a KNN classifier. A popular technique in this category is Wilson editing [2]; it removes all instances that have been misclassified by the KNN rule from a training set. The idea of Wilson editing relies on the fact that one can optimally eliminate outliers and possible overlap among classes from a given training set, so that the training of the corresponding classifier becomes easier in practice. It has been shown by Penrod and Wagner [3] that the accuracy of a Wilson edited nearest neighbor classifier converges to Bayes' error as the number of instances approaches infinity. Figure 1.a shows a hypothetical dataset where instances that are misclassified using the 1-NN-rule are marked with circles around them. Figure 1.b shows the reduced dataset after applying Wilson editing.



a. Hypothetical Dataset b. Edited Dataset

Fig. 1. Wison editing for a 1-NN classifier.

Wilson editing edits each instance $x_i \in T$ based on the label of its neighbor instances $N_T \subset T$. Now suppose we have another labeled data set T_{new} ($T_{new} \cap T = \varnothing$). We edit the instances $x_i \in T$ again based on its neighbors $N_{T \cup T_{new}} \subset (T_{new} \cup T)$. Usually $N_{T \cup T_{new}}$ is closer (more similar) to x_i compared with N_T and consequently it edits x_i more correctly. Let T_e and $T_{e(T \cup T_{new})}$ denote the edited datasets of T that based on T and $T_{new} \cup T$ respectively. Our experiment results show that $T_{e(T \cup T_{new})}$ could achieve better generalization ability than T_e.

However, in many cases, the additional training set T_{new} is not available, and instead of it, many unlabeled data U exist. In machine learning, the learning with both labeled instances and unlabeled instances is called semi-supervised learning. Existing semi-supervised learning only focuses on semi-supervised classification. To the best of our knowledge, there is no work done for semi-supervised data editing.

In this work, we propose to utilize those unlabeled data U for Wilson editing. Let $T_{e(T \cup U)}$ denotes the edited dataset of T based on $T \cup U$. Although $T_{e(T \cup U)}$ might be not good as $T_{e(T \cup T_{new})}$, we argue that it should provide better generalization ability compared with T_e.

Donghai Guan is with the Computer Engineering Department, Kyung Hee University, Korea (e-mail: donghai@oslab.khu.ac.kr).

Weiwei Yuan is with the Computer Engineering Department, Kyung Hee University, Korea (e-mail: weiwei@oslab.khu.ac.kr).

Young-Koo Lee (Corresponding author) is with the Computer Engineering Department, Kyung Hee University, Korea (e-mail: yklee@khu.ac.kr).

Sungyoung Lee is with the Computer Engineering Department, Kyung Hee University, Korea (e-mail: sylee@oslab.khu.ac.kr).

978-1-4244-1821-3/08/$25.00©2008 IEEE

We term our proposed approach semi-supervised nearest neighbor editing (SSNNE). We compared it with k-nearest neighbor and Wilson editing (NNE-nearest neighbor editing) using 12 UCI datasets. Experiment results show that in terms of classification accuracy, Wilson editing is worse than KNN; while, SSNNE outperforms KNN and Wilson editing.

II. DATA EDITING WHEN AN ADDITIONAL TRAINING SET EXITS

The Wilson editing rule is shown in Table I:

TABLE I
WILSON EDITING

Let $T_e = T$. (T is the original training set, and T_e will be the edited set) For each $x_i \in T$, do: -- Discard x_i from T if it is misclassified using the k-NN rule with prototypes in $T/\{x_i\}$

Given the training set T , the similarity between instance x_i and its neighbor instances, denoted by $d(x_i, N_T)$, plays an important role in editing. With $d(x_i, N_T)$ smaller, the neighbor instances N_T will be more similar with x_i and their editing on x_i will be more persuasive and correct.

One intuitive idea of reducing $d(x_i, N_T)$ is to adding the size of training set. Suppose we get another training set T_{new} and $T_{new} \cup T = T'$. In T' , the neighbor instances of x_i is $N_{T'}$. Since $T \subset T'$, $d(x_i, N_{T'}) \le d(x_i, N_T)$. In this case, the editing performance is usually improved. It should be noted that we will not edit any instance of T_{new} . The existence of T_{new} is only used for editing dataset T . T 's Wilson editing based on $T_{new} \cup T$ is shown in Table II.

TABLE II
WILSON EDITING WITH ANOTHER TRAINING SET

Given another training set T_{new} ($T_{new} \cap T = \varnothing$) Let $T_e = T$. (T is the original training set, and T_e will be the edited set) For each $x_i \in T$, do: -- Discard x_i from T if it is misclassified using the k-NN rule with prototypes in $(T/\{x_i\}) \cup T_{new}$

The function of T_{new} can also be explained by Figure 2. In Fig. 2, the black points and white points belong to different classes.

When we edit T using Wilson editing (1-NN), Instance x_1 will not be removed since its nearest neighbor among T has the same label with it. Instance x_2 will be removed since its nearest neighbor among T has different label with it. However, when T_{new} exist, the editing result of x_1 and x_2 will be changed. Now x_1 will be removed and x_2 will be kept.

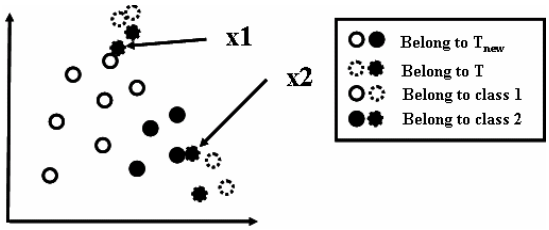


Fig. 2. The function of additional training set for data editing

We further compare the performance of data editing with and without the additional training set. Twelve data sets from the UCI Machine Learning Repository [4] are used in this experiment, where missing values on continuous attributes are set to the average value while those on binary or nominal attributes are set to the majority value. Information on these data sets is tabulated in Table III.

TABLE III
UCI DATA SETS

Data set	Attribute	Size	Class	Class distribution
iris	4	150	3	50/50/50
bupa	6	345	2	145/200
breast	9	1000	2	700/300
glass	9	214	6	70/76/17/13/9/30
australian	14	690	2	383/307
diabetes	8	768	2	500/268
echo	7	131	2	88/43
german	24	1000	2	700/300
heart1	13	303	2	164/139
heart2	13	294	2	188/106
horse	15	368	2	232/136
wine	13	178	3	59/71/48

Each data set D will be randomly divided into two parts: L and U . In this part, the ratio of L is 50% ($|L|/|D| = 0.5$). Ten-fold cross-validation on L was used to evaluate the performance of Wilson editing. At each trial, 90% of L is used as training set (T). The remaining 10% of L was used as test set which evaluates the generalization ability of T 's edited sets through classification accuracy. Let T_e and $T_{e(T \cup T_{new})}$ denote the edited datasets of T that based on T and $T_{new} \cup T$ respectively. T_{new} is the additional training set which is randomly selected from U with $|T_{new}| = |T|$. The average accuracy over 10 such trials is calculated. To get a fair result, above experiment is repeated for 5 times. In each time, the division of L and U is different. Finally we will get 5 accuracies. The average accuracy over them is reported in Table IV. To make a clearer view of the relative performance between KNN (T), Wilson editing T_e and $T_{e(T \cup T_{new})}$, we use a scoring mechanism.

Initially the scores of them are all zero. Then for each data set, the performances between this three methods are

compared and consequently the score of them (T , T_e and $T_{e(T\cup T_{new})}$) which gives the best result is incremented by one. Conversely, the score of it is decreased by one if it gives the worst result. It is noted that moderate result (neither best nor worst) does not change the score.

TABLE IV
PERFORMANCE COMPARISON BETWEEN T_e AND $T_{e(T\cup T_{new})}$

Data set	KNN	Wilson Editing (T)	Wilson Editing ($T+T_{new}$)
iris	0.9269	0.9212	0.9343
bupa	0.5932	0.6214	0.6385
breast	0.9695	0.9725	0.9725
glass	0.6453	0.6076	0.6217
australian	0.8319	0.8548	0.86
diabetes	0.7265	0.7354	0.748
echo	0.67	0.72	0.72
german	0.6932	0.7088	0.7192
heart1	0.8062	0.8259	0.8284
heart2	0.7952	0.7943	0.8067
horse	0.8081	0.8341	0.833
wine	0.954	0.9465	0.9477

For above Table, $SCORE_1 = -8$, $SCORE_2 = -1$ and $SCORE_3 = 9$. $SCORE_1$ denotes the score of KNN without editing. $SCORE_2$ denotes the score of data editing based on T and $SCORE_3$ denotes the score of data editing based on $T \cup T_{new}$.

This experiment results verify that when another training set T_{new} is used for T 's Wilson editing, the edited data set $T_e \subset T$ will achieve better generalization ability.

III. SEMI-SUPERVISED LEARNING

Above section shows that when editing T , better edited result could be achieved if an additional training set T_{new} is available. However, in most cases, the additional training set T_{new} is not available, and instead of it, many unlabeled data U exist. The reason is that labeled instances are often difficult, expensive, or time consuming to obtain, as they require the efforts of experienced human annotators. Meanwhile unlabeled instances may be relatively easy to collect. In this paper, we propose to utilize these unlabeled data to boost the performance of data editing.

How to use unlabeled data U for data editing is the main issue we should address. Through the experiment in Section 2, an intuitive idea of solving this issue is to find a way to convert these unlabeled data to labeled data. Much work on semi-supervised learning has been done. Although most of them are used for semi-supervised classification, some ideas of them can also be used to solve our problem.

Semi-supervised classification includes many methods such as generative models [5][6][7], self-training[8], co-training[9], graph-based methods[10] and so on. Among them, self-training and co-training are good candidates to solve our problem. Both of them convert some data from unlabeled dataset to label dataset somehow. This labeled set can be used as T_{new} in Section 2.

In self-training a classifier is first trained with the small amount of labeled data. The classifier is then used to classify the unlabeled data. Typically the most confident unlabeled points, together with their predicted labels, are added to the training set. The classifier is re-trained and the procedure repeated. Note the classifier uses its own predictions to teach itself.

For co-training, it has strict requirement on the data. It requires that features can be split into two sets; each sub-set is sufficient to train a good classifier; The two sets are conditionally independent given the class. Initially two separate classifiers are trained with the labeled data, on the two sub-feature sets respectively. Each classifier then classifiers the unlabeled data, and 'teaches' the other classifier with the few unlabeled examples (and the predicted labels) they feel most confident. Each classifier is retrained with the additional training examples given by the other classifier, and the process repeats.

As explained above, both of self-training and co-training have their own limitations. For self-training, on one hand, the classifier requires some measures to evaluate the "confidence" of unlabeled data. Actually most classifier can not give this measure easily. On the other hand, even the classifier could measure the confidence, its own prediction on the unlabeled data might not be correct. Co-training is lack of generality since it only works for the datasets which can be represented by two views.

Through combing self-training and co-training, we propose to use ensemble-based co training (En-Co-training) methods. As shown in Table V, three classifiers (algorithms) are used instead of individual classifier of self-training and two classifiers of co-training. En-Co-training overcome the limitations of self-training and co-training. Through majority voting of these three classifiers, explicit measure of confidence is not required. In addition, in co-training, the diversity of each classifier is achieved by using different sets of features which requires two views of features. In En-co-training, the diversity of each classifier is achieved by using different algorithms. Hence, En-co-training has not any requirement on the data set. At the beginning of En-Co-training, small number of examples U' is randomly selected from U . Then at the end of each iteration, more examples will be replenished into U' . Many related work show that this kind of setting is better than dealing with the whole U directly.

The data editing process of Wilson, Wilson with additional training set T_{new} and Wilson with semi-supervised learning can be compared using Fig. 3.

The result that $T_{e(b)}$ is better than $T_{e(a)}$ has been shown in Section 2, however, it should be noted that $T_{e(c)} \neq T_{e(b)}$. The

reason is that $T_{new(c)}$ is generated by semi-supervised learning which might include mislabeled data. While, $T_{new(b)}$ is the real data without noise. We argue that although there might be some noise in $T_{new(c)}$, when its number is not big, $T_{new(c)}$ will still be useful for Wilson editing.

TABLE V
ENSEMBLE-BASED CO-TRAINING ALGORITHM

Given:
- L, A set of labeled training examples x, consists of M classes
- U, A set of unlabeled examples

Create a pool U' of examples by choosing u examples at random from U

Loop for k iterations:

(1) Use L to train a classifier h_1, h_2 and h_3 respectively

(2) For each class C, pick the n_c unlabeled data which classifier h_1, h_2 and h_3 agree with that its class label is C and add it to the collection of labeled examples

(3) Randomly choose $\sum_{c=1}^{M} n_c$ examples from U to replenish U'

- Output: The new labeled data generated from step (2).

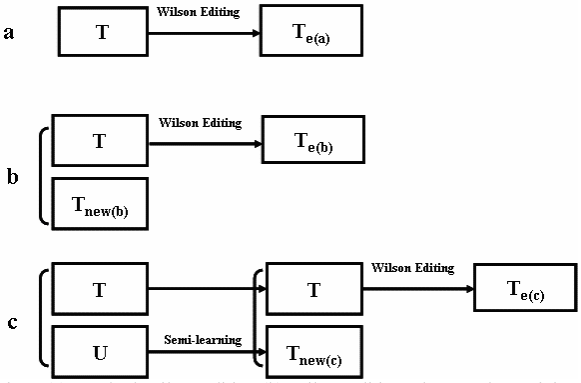


Fig. 3 (a) standard Wilson editing (b) Wilson editing when another training set is available (c) semi-supervised Wilson editing

IV. EXPERIMENTAL RESULTS

In this section the experiment is similar with that done in Section 2. Each data set D will be randomly divided into two parts: L and U. The only difference is the labels of T_{new} here are generated from semi-supervised learning as shown in Fig. 3(c). In addition, considering that the ratio of L, $|L|/|D|$, might influence the experiment results, different labeled ratios are tested including 50%, 40%, 30% and 20%. As shown in the following tables, each column gives the classification accuracy. Let T and T_e represent the original data set and edited data set respectively. Finally the scoring mechanism is same with that used in Section 2. NNE denotes Wilson editing (nearest neighbor editing). SSNEE denotes semi-supervised nearest neighbor editing.

TABLE VI
PERFORMANCE COMPARISON WHEN LABELED RATIO IS 50%

Labeled data ratio=50%			
Data set	KNN	NNE	SSNNE
iris	0.9269	0.9212	0.9343
bupa	0.5932	0.6214	0.6385
breast	0.9695	0.9725	0.9725
glass	0.6453	0.6076	0.6217
australian	0.8319	0.8548	0.86
diabetes	0.7265	0.7354	0.748
echo	0.67	0.72	0.72
german	0.6932	0.7088	0.7192
heart1	0.8062	0.8259	0.8284
heart2	0.7952	0.7943	0.8067
horse	0.8081	0.8341	0.833
wine	0.954	0.9465	0.9477
SCORES	**-8**	**-1**	**9**

TABLE VII
PERFORMANCE COMPARISON WHEN LABELED RATIO IS 40%

Labeled data ratio=40%			
Data set	KNN	NNE	SSNNE
iris	0.9167	0.9133	0.9333
bupa	0.5528	0.5798	0.6015
breast	0.9624	0.9661	0.9668
glass	0.5346	0.5214	0.533
australian	0.8189	0.8271	0.8358
diabetes	0.7137	0.7147	0.7309
echo	0.615	0.654	0.659
german	0.6935	0.7035	0.704
heart1	0.795	0.8202	0.8152
heart2	0.7807	0.8152	0.8207
horse	0.8107	0.8196	0.8287
wine	0.9505	0.9139	0.9283
SCORES	**-7**	**-2**	**9**

Table VI shows that when label ratio is 50%, the score of KNN is -8; the score of NNE is -1; while the score of SSNNE is 9. Hence SSNEE is best in term of classification accuracy (generalization ability). Table VII gives the comparison when label ratio is 40%. At this ratio, SSNEE's score is 9, which is much better than KNN (-7) and NNE (-2). Table VIII shows the comparison result when label ratio is 30%. The result is: SSNEE's classification accuracy is best (score is 9); while NNE is the second best (score is -1); KNN is worst (score is

-8). In Table IX, label ratio is 20%. The result shows that SSNEE is best in term of classification accuracy. And it should be noted that KNN is better than NNE in this Table. Hence it shows that NNE cannot provide consistent improvement over KNN.

TABLE VIII
PERFORMANCE COMPARISON WHEN LABELED RATIO IS 30%

Labeled data ratio=30%			
Data set	KNN	NNE	SSNNE
iris	0.9269	0.9212	0.9343
bupa	0.5932	0.6214	0.6385
breast	0.9695	0.9725	0.9725
glass	0.6453	0.6076	0.6217
australian	0.8319	0.8548	0.86
diabetes	0.7265	0.7354	0.748
echo	0.67	0.72	0.72
german	0.6932	0.7088	0.7192
heart1	0.8062	0.8259	0.8284
heart2	0.7952	0.7943	0.8067
horse	0.8081	0.8341	0.833
wine	0.954	0.9465	0.9477
SCORES	**-8**	**-1**	**9**

TABLE IX
PERFORMANCE COMPARISON WHEN LABELED RATIO IS 20%

Labeled data ratio=20%			
Data set	KNN	NNE	SSNNE
iris	0.9	0.86	0.8933
bupa	0.598	0.574	0.568
breast	0.9436	0.9539	0.9539
glass	0.6621	0.5086	0.5421
australian	0.8648	0.8737	0.8719
diabetes	0.6985	0.6934	0.7185
echo	0.67	0.6522	0.66
german	0.687	0.691	0.694
heart1	0.7567	0.7624	0.7862
heart2	0.796	0.7828	0.795
horse	0.7597	0.7857	0.7961
wine	0.9333	0.8044	0.8222
SCORES	**0**	**-4**	**4**

Table VI-IX show that for most data sets used in our experiment, data editing could generate improvement over KNN. And SSNNE could make more improvement than NNE. For other data set, such as wine, data editing will decrease the classification accuracy. In these cases, the bad influence generated by SSNEE is less than NNE.

In summary, when data editing is useful, SSNEE, using unlabeled data, could generate more improvement in terms of classification accuracy. When data editing is harmful, SSNEE could decrease the bad influence through using unlabeled data.

V. CONCLUSIONS AND FUTURE WORK

In this work, we propose a novel data editing approach. The core idea is to boost the generalization ability of data editing through using unlabeled data. To the best of our knowledge, it is the first work in data editing which utilizes unlabeled data.

The experimental results show that the classification accuracy of our proposed semi-supervised nearest neighbor editing outperforms Wilson editing and k-nearest neighbor. Although we present our approach based on Wilson editing, it can also be used by other data editing methods.

Our future work mainly includes two parts: 1) testing the performance of our approach in the real applications developing by our lab, such as activity recognition and location recognition. 2) testing our approach using other data editing methods.

ACKNOWLEDGMENT

This research was supported 1) by the MKE (Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Advancement) (IITA-2008-(C1090-0801-0002)). 2) by the MIC (Ministry of Information and Communication), Korea, Under the ITFSIP (IT Foreign Specialist Inviting Program) supervised by the IITA (Institute of Information Technology Advancement, Project Management Number: C1012-0801-0003.

REFERENCES

[1] B.V. Dasarathy, *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*. Los Alamitos, Calif: IEEE CS Press, 1991.

[2] D.L. Wilson, "Asymptotic Properties of Nearest Neighbor Rules Using Edited Data," *IEEE Transactions on Systems, Man, and Cybernetics*, 2: 408-420, 1972.

[3] C. Penrod and T. Wagner, "Another look at the edited nearest neighbor rule," *IEEE Transactions on Systems, Man, and Cybernetics*, 7: pp. 92-94, 1977.

[4] UCI Machine Learning Repository, http://www.ics.uci.edu/~mlearn/MLRepository.html.

[5] K. Nigam, A. K. McCallum, S. Thrun and T. Mitchell, "Text classification from labeled and unlabeled documents using EM," *Machine Learning*, 39, pp: 103-134, 2000.

[6] S. Balujia, "Probabilistic modeling for face orientation discrimination: Learning from labeled and unlabeled data," *Neural Information Processing Systems*, pp: 854-860, 1998.

[7] A. Fujino, N. Ueda and K. Saito, "A hybrid generative/discriminative approach to semi-supervised classifier design," *The twentieth National Conference on Artificial Intelligence*, pp: 764-769, 2005.

[8] E. Riloff, J. Wiebe and T. Wilson, "Learning subjective nouns using extraction pattern bootstrapping," *Proceedings of the Seventh Conference on Natural Language Learning*, pp: 25-32, 2003.

[9] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," *COLT: Proceedings of the Workshop on Computational Learning Theory*, pp: 92-100, 1998.

[10] A. Blum and S. Chawla, "Learning from labeled and unlabeled data using graph mincuts," *Proceeding of 18th International Conference on Machine Learning*, pp: 19-26, 2001.