Conservative Noise Filters

Mona M. Jamjoom
Department of Computer Sciences
College of Computer & Information Sciences
King Saud University, Riyadh, Saudi Arabia

Khalil El Hindi
Department of Computer Sciences
College of Computer & Information Sciences
King Saud University, Riyadh, Saudi Arabia

***Abstract*—Noisy training data have a huge negative impact on machine learning algorithms. Noise-filtering algorithms have been proposed to eliminate such noisy instances. In this work, we empirically show that the most popular noise-filtering algorithms have a large False Positive (FP) error rate. In other words, these noise filters mistakenly identify genuine instances as outliers and eliminate them. Therefore, we propose more conservative outlier identification criteria that improve the FP error rate and, thus, the performance of the noise filters. With the new filter, an instance is eliminated if and only if it is misclassified by a mutual decision of Naïve Bayesian (NB) classifier and the original filtering criteria being used. The number of genuine instances that are incorrectly eliminated is reduced as a result, thereby improving the classification accuracy.**

Keywords—component; Instance Reduction Techniques; Instance-Based Learning; Class noise; Noise Filter; Naive Bayesian; Outlier; False Positive

I. INTRODUCTION

In Machine Learning (ML), the quality of the training data can have a huge impact on the induced classifier. Noise can cause a learning algorithm to overfit the training data [1], which harms the classifier's performance. ML algorithms use different methods to mitigate the effect of noise. For example, decision-tree learning algorithms use pruning techniques [2] whereas neural networks use validation datasets to determine when to stop the training process [1]. In Instance-Based Learning (IBL), the effect of noise is mitigated by using a large number of several similar instances instead of just one as done by the k Nearest Neighbor (kNN) algorithm, where k (the number of neighbors) is usually set to 3. Another general approach that can mitigate the effect of noise is to use a noise-filtering algorithm that determines and eliminates the outlier instances [3], [4], [5], [6], [7], [8]. Although, most of these methods were designed for IBL methods, they can also be used to preprocess the training data before using them with other ML approaches, such as decision trees [9] and neural networks [10], [11]. The efficacy of ML methods, specifically kNN, is highly influenced by the quality of training data. This is most obvious when the number of neighbors, k, is set to one [12]; the problem is less severe when k is set to larger values (e.g., 3). Some Instance Reduction (IR) techniques [13], [14], [15] have been developed as noise filters specifically to tackle the noise problem.

In this work, we empirically show that these filters may eliminate some genuine instances because they mistakenly identify them as outliers due to the noise effect [13]. In other words, their FP (i.e., incorrectly eliminated noisy instances) error rate in identifying outliers is relatively high. This is because they use a relaxed outlier identification criterion, which is especially bad when the available training data are limited in size. This work proposes more conservative identification criteria to replace the outlier identification criteria of ENN [14], RENN, and All-kNN [15] noise filters. The proposed method uses the decision of NB classifier and the decision of the noise filter being used to determine whether to discard the instance or keep it.

The empirical results using 50 benchmark datasets obtained from UCI machine learning repository [16] show that the new method improves the performance of these noise filters at different noise ratios. The proposed conservative methods proved to be effective at minimizing the FP error rate. In other words, the methods managed to save more genuine instances and improved the classification accuracy. We present a comprehensive comparison between the methods in terms of the average classification accuracy, number of datasets in which the methods achieve better results and significantly better results, the average percentage of eliminated FP instances, the average ratio of data reduction, and the average percentage of True Positive (TP) instances (i.e., correctly eliminated noisy instances) eliminated by each algorithm.

The paper is organized as follows: Section 2 presents an overview of noise-filtering techniques used in the paper and NB classifier, section 3 presents the conservative criteria and empirically discusses and analyses the results, and section 4 concludes the paper.

II. RELATED WORK AND BACKGROUND MATERIAL

In this section, we review the noise-filtering material that we modify as well as the Naïve Bayesian algorithm.

A. IR Techniques for Noise Filtering

The problem of noise in classification has been an active research area for many decades, and most machine learning algorithms focus on this issue. The problem has continued to be a major challenge due to the uncertainty property of the noise [17]. Various approaches to dealing with the problem of noise have been integrated with the learning algorithms to mitigate its effect and improve the learning capabilities. [18] categorized the techniques that handle noise into three main groups: robust, polishing, and filtering.

Robust techniques leave the noise in the dataset and use an embedded pruning phase to mitigate its effect. Such techniques are used in decision trees and rule learning. In decision trees some branches are pruned C4.5 [19], while in rule learning some preconditions of rules are pruned CN2 [20]. However, the classifier built from noisy a dataset may be less

representative and less predictive if the noise ratio is very high. [21] preferred to handle the noise as a pre-processing phase, so that the constructed classifier is not affected by the noise.

The other two techniques (i.e., polishing and filtering) use more pure training data as the noise is preprocessed before any classifier is constructed; therefore, most studies tend to use such techniques [17]. Polishing techniques try to repair noisy instances by replacing the suspected attribute values with other appropriate values [22], [23]. The new values are determined based on the class of an instance and some additional attribute values. Correcting or relabeling an instance is a risky process because it can replace a noisy value with another [24], [18], [25]. Meanwhile, filtering techniques use an independent noise-filtering stage in which noisy instances that meet certain criteria are determined and discarded. Noise filtering has been implemented in different forms with different types of classifiers [26], [6], [27], [28], [4], [9], [10], [3], [5], [8], [29] and has been proven to be effective in improving the classification accuracy [25].

IR techniques were developed to speed up and reduce the storage requirements for IBL while preserving the classification accuracy [13]. Some such techniques are designed specifically to work as noise filters (e.g., ENN [14], RENN, and All-kNN [15]). The retained set of instances is purer and better represents the underlying instance space. The filtering techniques used in this paper include the following:

- The **ENN** [14] is a decremental algorithm that starts with the complete training set and eliminates an instance if it is misclassified by its k nearest neighbor(s). We set k to 3 in this work. The algorithm smooths the decision boundaries by removing the noisy instances, which are typically closer to the border. The pseudo code for the ENN algorithm is shown in Figure 1.
- The **RENN** [15] is a repeated form of ENN until no more instances can be removed. This will increase the gap between classes.
- **All-kNN** [15] is a batch algorithm that starts with a complete training set. It marks all instances misclassified by its i neighbors for all i = 1 to k. The elimination is done once, after all the instances in the training set are examined. Internal noisy instances within a class as well as odd instances on the border will be removed. The pseudo code for the All-kNN algorithm is shown in Figure 2.

```
T is a training set contains all training instances

For each instance (i)
If (instance(i).class<> majority class of k neighbors)
        Remove instance(i) from T
```

Fig. 1. Pseudo code for ENN algorithm

```
T is a training set contains all training instances
oldk=k
For each instance (i)
 For k=1 to oldk
  If (instance(i).class<> majority class of k neighbors)
        mark instance(i) to be removed
Remove all marked instances
```

Fig. 2. Pseudo code for All-kNN algorithm

B. Naïve Bayesian (NB)

The NB classifier is a simple form of Bayesian Network (BN) with one parent and several children [30]. It is a probabilistic classifier based on Bayes' theorem with strong (naïve) independence assumptions between the features, given the class [31]. To classify an instance, NB calculates the conditional probability for each instance class value and considers the class with the maximum probability as the predicted class, as shown in Equation 1.

$$\text{Class}_{predicted} = \underset{c \in C}{\operatorname{argmax}}\, p(c) \cdot \prod_j p(a_j|c) \quad (1)$$

where, C is a vector of all class attribute values, $p(c)$ is the probability of class c, and $\prod_j p(a_j|c)$ is the naïve assumption that all the attribute values are conditionally independent given the class value. Domingos et al. [32] found that NB performance is competitive with more sophisticated ML methods, such as DT, IBL, and rule induction, even if the features' dependency is very strong. Moreover, NB is a strongly noise-tolerant algorithm [33], [34]. Nettleton et al. [33] performed a systematic analysis of robustness of many ML algorithms to noise—namely, NB, C4.5, IBk, and SMO. They determined that NB is the most noise-robust ML algorithm. An extended NB structure obtained from noisy data was presented in [35], in which the NB is constructed from noisy data and is incorporated with the NB model constructed from real data using linear equations and optimization methods. The proposed method was effective in handling noisy data and achieving classification accuracy. El Hindi [36] enhanced the performance of NB classification by using a fine-tuning stage to improve the probabilities estimation, but this degraded the sensitivity of NB toward noise. Therefore, several modifications for the fine-tuning process were proposed by [34]. They simply assign smaller weights for noisy instances during the fine-tuning process instead of eliminating or correcting them.

III. Conservative Noise Filters

In this section, we study and compare the performance of the reviewed noise filters and suggest more conservative criteria to identify outliers. To study their robustness to noise, we performed several experiments with different noise ratios. We paid special attention to the FP error rate [37] of each algorithm, because we believe that these noise filters mistakenly classify many genuine instances as outliers and, consequently, eliminate them.

TABLE I. CHARACTERISTICS OF DATA SETS USED IN THE EXPERIMENTS

Data set	# of Classes	# of Attributes	# of Instances
anneal	6	39	798
anneal.ORIG	6	39	798
arrhythmia	16	280	452
audiology	22	70	226
autos	7	26	205
Breast-cancer	2	10	286
Breast-w	2	10	699
bridges_version1	6	13	108
bridges_version2	6	13	108
car	4	7	1728
colic	2	23	368
colic.ORIG	2	28	368
congress-voting-1984	2	17	435
credit-a	2	16	690
credit-g	2	21	1000
cylinder-bands	2	40	512
dermatology	6	35	366
diabetes	2	9	768
ecoli	8	8	336
flags	8	30	194
glass	7	10	214
heart-c	5	14	303
heart-h	5	14	294
heart-statlog	2	14	270
hepatitis	2	20	155
hypothyroid	4	30	3772
ionosphere	2	35	351
iris	3	5	150
labor	2	17	57
lung-cancer	2	57	32
lymph	4	19	148
monk2	2	7	169
musk1	2	167	476
postoperative-patient-data	3	9	90
primary-tumor	22	18	339
segment	7	20	2310
sick	2	30	3772
solar-flare_1	2	13	323
solar-flare_2	3	13	1066
sonar	2	61	208
Soybean	19	36	683
spect_train	2	23	80
splice	3	62	3190
sponge	3	46	72
tennis	2	5	14
trains	2	33	10
Vehicle	4	19	846
Vote	2	17	435
Vowel	11	14	990
Zoo	7	18	101

In this work, the FP rate refers to the rate of mistakenly classifying genuine instances as outliers. Of course, a noise filter with a low FP rate is better than a noise filter with a high FP rate because the former helps retain more instances that are genuine. We must also take into account the False Negative (FN) rate, which refers to the rate of mistakenly classifying instances as genuine instances when they are in fact outliers. A noise filter with a high FN rate is bad because it would fail to eliminate all outliers.

We used 50 benchmark datasets obtained from the UCI Repository for Machine Learning [16] to experimentally test the FP and FN rates of the noise filters. We deliberately inserted class noise in the training sets by replacing the class values of some randomly selected instances with other random class values. We kept the class values in the test datasets unchanged and used different noise ratios of 0%, 5%, 10%, 15%, and 20%. Each noisy experiment was repeated five times. We used the kNN algorithm, with k = 3, and the discretized VDM (DVDM) as a distance function [38]. Ten-fold cross-validation and a paired t-test with a confidence level of 95% were used in all experiments. We compared the different methods with respect to the following criteria: average classification accuracy, the number of datasets in which each method achieved better, and significantly better results. We also calculated the average reduction size and the FP rate for each noise filter. The FP rate was calculated according to Equation 2 [39],

$$\text{The False Positive Rate (FPR)} = \frac{FP}{FP+TN} \quad (2)$$

where TN is the number of correctly retained instances (i.e., True Negative). We implemented kNN and the classical IR noise filters (i.e., ENN, RENN, and All-kNN) in the Weka work frame [40]. Table 1 lists the main characteristics of these datasets in terms of the number of class values, the number of attributes, and the number of instances.

A. The Prerforamnce of the IR Noise Filters

We simply applied the selected noise-filtering algorithms to the datasets and compared the results of the kNN before and after filtering. Table 2 summarizes the results. As shown in Table 2, at 0% noise, kNN outperforms all noise filters. In addition, All-kNN outperforms RENN, which is consistent with the results reported by [13] and [15]. However, the advantage of using noise filters is obvious when used with noisy training sets, especially when the noise ratio increases. At 5% noise, the noise filters start to influence the results in a positive way.

Although, at 5%, noise ENN shows better results than the rest of the filters, at 10%, 15%, and 20% noise, RENN emerges as the best noise filter; ENN is more conservative and thus more suitable when we have a low ratio of noise, yet when we have a large ratio of noise, we need a more aggressive algorithm, such as RENN. In general, applying a noise-filtering algorithm to noisy data before classification improves the classification accuracy, which is consistent with the findings of [13].

Table 3 shows the average reduction in the size of datasets as a result of applying each noise-filtering algorithm. RENN has the largest reduction in size rate among all algorithms due to its natural repeated elimination process.

It is obvious that the number of instances eliminated by the original noise-filtering algorithms is much greater than the number of outliers. For example, at 15% noise, the ENN, RENN, and All-kNN eliminate 27.4%, 30.65%, and 21.26% of the datasets, respectively, which is much higher than the noise ratio. Thus, these noise filters eliminate some genuine instances that are incorrectly classified by the filter as an outlier, resulting in the average percentage of eliminated FPs listed in Table 6 for these filters being considered high.

TABLE II. THE RESULTS OF THE KNN ALGORITHM BEFORE AND AFTER APPLYING EACH NOISE FILTER

Class Noise %	Criteria	kNN	ENN	kNN	RENN	kNN	All_kNN
0	Average Accuracy	**78.79**	78.07	**78.79**	77.10	**78.79**	78.06
	# Better datasets	**30**	16	**30**	16	**20**	12
	# Sign. better	**3**	1	**3**	1	**3**	2
5	Average Accuracy	76.66	**77.60**	**76.66**	76.50	**76.66**	76.32
	# Better datasets	9	**40**	16	**33**	14	**35**
	# Sign. better	2	**22**	7	**23**	8	**22**
10	Average Accuracy	75.68	**76.01**	**75.68**	75.52	**75.68**	75.64
	# Better datasets	14	**33**	15	**34**	16	**34**
	# Sign. better	2	**18**	9	**24**	12	**23**
15	Average Accuracy	73.61	**74.36**	73.61	**73.93**	73.61	**73.80**
	# Better datasets	14	**36**	16	**34**	19	**31**
	# Sign. better	2	**18**	6	**24**	8	**18**
20	Average Accuracy	71.84	**72.55**	71.84	**72.86**	71.84	**71.86**
	# Better datasets	14	**36**	14	**35**	22	**27**
	# Sign. better	2	**14**	5	**25**	9	**19**

TABLE III. THE AVERAGE REDUCTION IN SIZE OF THE NOISE FILTER AT DIFFERENT NOISE RATIOS

Class Noise %	ENN%	RENN%	All_kNN%
0	13.25	15.74	9.26
5	18.26	20.97	13.81
10	23.37	25.92	17.89
15	27.40	30.65	21.26
20	31.49	34.92	24.30

B. Hybrid Outlier-Identification Criteria

The fact that these noise filters have a high FP rate suggests that the noise filters are too relaxed or loose in determining the outliers. To improve their performance, we believe that their criteria for determining outliers need to be more conservative and restricted. As the NB learning algorithm is robust to noise, we use it to make the noise identification criteria more conservative. We consider an instance an outlier if the NB classifier misclassifies it; the condition is added to the original noise identification criteria of ENN, RENN, and All-kNN. As a result, we ultimately double check if an instance is really an outlier before discarding it. The new hybrid outlier-identification criteria aim to sift through the suspicious instances to determine the actual outliers. We expect that this approach will reduce the number of mistakenly eliminated FP instances. Mutual decision of more than one condition for elimination can be helpful in this case.

Mutual decision has been used with instance reduction techniques in [9]. The authors applied the DROP5 algorithm on instances marked by the All-kNN algorithm (i.e., AllKnnDROP5) as a pre-pruning phase before applying the rule induction on the reduced training set. The hybrid reduction techniques gave better results in terms of average accuracy when compared to ENN, AllKnn, and DROP5 techniques individually. Mutual decision looks relatively similar to the ensembles' voting concept [25], in which considering different methods can lead to better decisions and thus better performance, as shown in [9].

To evaluate the performance of the conservative noise filters, we re-performed the previously described experiments using these algorithms. We prefixed the name of the noise filters with NB (e.g., "NB_ENN") to distinguish the conservative algorithms from the original algorithms. Table 4 shows the results of proposed conservative filters and their original counterparts. When noise-free, the conservative filters achieved better than all other filters used. This is expected as they eliminate fewer genuine instances.

In the presence of noise, the conservative filter with ENN and All-kNN gave better results than their corresponding regular ENN and All-kNN filters. The improved accuracy during noise increases is not less on average than 0.4% and 0.7% for NB_ENN and NB_All-kNN, respectively. Combining NB with RENN does not enhance the original RENN; this will be explained shortly.

Of course, the conservative filters eliminate fewer instances than the original filters. This is obvious and has been confirmed by the average reduction size of each algorithm reported in Table 5. The average reduction in size of the noise filter algorithms is larger than the noise ratio.

On the other hand, this average is close to the noise ratio when we used the conservative noise filters. Comparing Tables 3 and 5 shows a big difference in reduction between the conservative filter and the original one.

Table 6 shows the average percentage of eliminated FP instances by each algorithm. As we can see, the All-kNN filter eliminated more FPs among other techniques' reach by an average of 12.08%, followed by RENN with an average elimination of 9.43% and ENN with an average of 5.57%.

Clearly, the number of FPs eliminated by the noise filters is high when compared with the number eliminated by the conservative filters, and this number increases as the noise ratio increases in most cases. For example, at the 20% noise ratio, ENN, RENN, and All-kNN eliminated 6.24%, 10.59%, and 15.43% of FP instances, respectively, compared to 5.41%, 7.3%, and 4.96% eliminated by the corresponding conservative filters. NB_All-kNN had the lowest average percentage of eliminated FPs among all conservative filters. The conservative filters save more FPs than the regular filter, thus the performance of NB_ENN and NB_All-kNN improved accordingly, as demonstrated in Table 4. This means that these retained instances are important and contribute positively in the classification as they increase the number of correctly retained instances (i.e., TN instances).

TABLE IV. SUMMARY OF RESULTS COMPARING THE CONSERVATIVE FILTER WITH THEIR COUNTERPARTS

Class Noise %	Criteria	NB_ ENN	ENN	NB_R ENN	RE NN	NB_All _kNN	All_ kNN
0	Average Accuracy	**79.14**	78.07	**78.17**	77. 10	**78.96**	78.06
	# Better datasets	**27**	19	**28**	14	**26**	19
	# Sign. better	**6**	3	**3**	2	**2**	**2**
5	Average Accuracy	77.58	**77.60**	75.77	**76. 50**	**77.08**	76.32
	# Better datasets	**28**	17	16	**31**	**26**	15
	# Sign. better	**9**	8	10	**16**	**15**	4
10	Average Accuracy	**76.42**	76.01	74.06	**75. 52**	**76.38**	75.64
	# Better datasets	**27**	17	19	**29**	**30**	15
	# Sign. better	**12**	3	8	**18**	**12**	2
15	Average Accuracy	**74.78**	74.36	72.91	**73. 93**	**74.78**	73.80
	# Better datasets	**29**	14	18	**30**	**30**	14
	# Sign. better	**17**	3	5	**19**	**15**	2
20	Average Accuracy	**73.07**	72.55	71.49	**72. 86**	**72.60**	71.86
	# Better datasets	**29**	13	15	**32**	**30**	13
	# Sign. better	**16**	2	3	**21**	**22**	1

TABLE V. THE AVERAGE REDUCTION IN SIZE OF CONSERVATIVE FILTERS AT DIFFERENT NOISE RATIOS

Class Noise %	NB_ENN%	NB_RENN%	NB_ All_kNN%
0	5.23	8.35	6.29
5	7.63	11.03	8.33
10	9.79	13.51	10.37
15	11.89	15.88	12.35
20	13.68	18.02	14.09

TABLE VI. THE FPR FOR EACH ALGORITHM

Class Noise%	The average percentage of eliminated FPs					
	NB_ENN %	ENN %	NB_RENN %	RENN %	NB_ All_kNN %	All_kNN %
5	4.28	5.07	6.59	8.43	4.46	8.76
10	4.66	5.29	6.84	8.98	4.58	10.93
15	5.1	5.68	7.03	9.7	4.82	13.19
20	5.41	6.24	7.3	10.59	4.96	15.43
Average	**4.86**	**5.57**	**6.94**	**9.43**	**4.71**	**12.08**

TABLE VII. THE FNR OF THE NOISE FILTERS

Class Noise%	The False Negative Rate					
	NB_ENN %	ENN %	NB_RENN %	RENN %	NB_ All_kN N %	All_kN N %
5	22.32	19.67	27.11	16.31	34.62	32.19
10	25.2	22.23	27.82	16.81	35.89	33.38
15	28.15	24.63	29	17.87	38.31	35.34
20	31.11	27.38	30.46	18.98	40.03	36.84
Average	**26.7**	**23.48**	**28.6**	**17.49**	**37.21**	**34.44**

As Table 4 shows, NB_RENN does not improve the regular filter despite the number of FPs eliminated by RENN being high compared to the conservative filter. The reason must be related to the FN rate of the algorithms. In other words, NB_RENN must have a higher FN rate than RENN, which simply means it is less effective at identifying and eliminating all outliers. Therefore, we calculated the FN rate, which is defined as Equation 3 [39],

$$FNR=1-TPR \quad (3)$$

where TPR is defined as Equation 4 [39],

$$\text{The True Positive Rate (TPR)} = \frac{TP}{TP+FN} \quad (4)$$

where FN is the number of incorrectly retained instances (i.e., False Negative).

Table 6 and Table 7 show the FPR and FNR for each algorithm at different noise ratio. As expected, Table 7 shows that RENN has the lowest FNR at all noise ratios; making it more conservative by combining it with NB substantially increases the FNR. For example, at 0% noise, RENN has an FNR of 16.31% while NB_RENN has 27.11%. Meanwhile, at 20% noise, RENN has an FNR of 18.98% while NB_RENN has 30.46%. Thus, although making RENN more conservative reduces its FPR (see Table 6), it also substantially reduces its ability to identify and eliminate all outliers. The problem is less severe for the other two algorithms because making them more conservative slightly increases their FNR.

Table 8 shows the results of the kNN before and after applying the conservative noise-filtering algorithm. Table 2 and Table 8 indicate that the best conservative algorithm is the NB_ENN whereas the best non-conservative algorithm is RENN. Therefore, it is logical to compare the effect of each compared to the kNN algorithm. Table 9 shows the result of this comparison, demonstrating that the NB_ENN's performance is much better than or equal to that of RENN on all noise for all ratios using all comparison criteria. The only exception is at 20% noise, where RENN achieves significantly better results than kNN for 20 datasets whereas NB_ENN achieves significantly better results than kNN for 13 datasets. However, at the same noise ratio, NB_ENN still achieves better average accuracy than RENN and achieves better results (but not significantly better) than kNN for 28 datasets whereas RENN achieves better results for 21 datasets. Thus, using the conservative ENN (i.e., NB_ENN) is probably better than using RENN, especially for small noise ratios ranging from 0% to 15%.

TABLE VIII. THE RESULTS OF KNN BEFORE AND AFTER APPLYING THE CONSERVATIVE NOISE FILTERS

Class Noise %	Criteria	kNN	NB_ENN	kNN	NB_RENN	kNN	NB_All_kNN
0	Average Accuracy	78.79	**79.14**	**78.79**	78.17	78.79	**78.96**
	# Better datasets	21	**25**	**26**	21	20	**23**
	# Sign. better	**4**	1	**3**	1	**3**	2
5	Average Accuracy	76.66	**77.58**	**76.66**	75.77	76.66	**77.08**
	# Better datasets	7	**42**	18	**31**	11	**38**
	# Sign. better	2	**27**	7	**20**	4	**23**
10	Average Accuracy	75.68	**76.42**	**75.68**	74.06	75.68	**76.38**
	# Better datasets	11	**38**	18	**32**	14	**34**
	# Sign. better	3	**22**	13	**19**	9	**24**
15	Average Accuracy	73.61	**74.78**	**73.61**	72.91	73.61	**74.78**
	# Better datasets	15	**35**	20	**30**	17	**33**
	# Sign. better	1	**19**	8	**19**	8	**24**
20	Average Accuracy	71.84	**73.07**	**71.84**	71.49	71.84	**72.56**
	# Better datasets	11	**39**	15	**35**	22	**27**
	# Sign. better	2	**15**	9	**21**	9	**21**

TABLE IX. COMPARING THE BEST CONSERVATIVE (NB_ENN) AND THE BEST NON-CONSERVATIVE (RENN) ALGORITHMS

Class Noise %	Criteria	kNN	NB_ENN	Difference= NB_ENN-KNN	kNN	RENN	Difference= RENN-KNN
0	Average Accuracy	78.79	**79.14**	**0.35**	**78.79**	77.1	**-1.69**
	# Better datasets	21	**25**	**4**	**30**	16	**-14**
	# Sign. better	**4**	1	**-3**	**3**	1	**-2**
5	Average Accuracy	76.66	**77.58**	**0.92**	**76.66**	76.5	**-0.16**
	# Better datasets	7	**42**	**35**	16	**33**	**17**
	# Sign. better	2	**27**	**25**	7	**23**	**16**
10	Average Accuracy	75.68	**76.42**	**0.74**	**75.68**	75.52	**-0.16**
	# Better datasets	11	**38**	**27**	15	**34**	**19**
	# Sign. better	3	**22**	**19**	9	**24**	**15**
15	Average Accuracy	73.61	**74.78**	**1.17**	73.61	**73.93**	**0.32**
	# Better datasets	15	**35**	**20**	16	**34**	**18**
	# Sign. better	1	**19**	**18**	6	**24**	**18**
20	Average Accuracy	71.84	**73.07**	**1.23**	71.84	**72.86**	**1.02**
	# Better datasets	11	**39**	**28**	14	**35**	**21**
	# Sign. better	2	**15**	**13**	5	**25**	**20**

IV. CONCLUSION

The problem of classical noise filters is that they eliminate large numbers of good instances when such instances are incorrectly identified as outliers and consequently eliminated. These good instances are useful for improving the classification accuracy of the induced classifier. Therefore, this work proposed a simple modification for these algorithms to make them more conservative. We proposed using hybrid outlier-identification criteria by combining an NB classifier with the original filtering criteria used by the algorithm. This work empirically shows that the conservative filters outperform the original filters because they have a smaller false positive rate (i.e., eliminate fewer genuine instances). The only exception is the conservative RENN (i.e., NB_RENN), which performs poorly compared to RENN despite the fact that NB_RENN has a smaller false positive rate, but a much higher false negative rate than RENN. Consequently, NB_RENN performs poorly compared to RENN, especially at large noise ratios. However, the conservative ENN (i.e., NB_ENN) outperforms RENN, especially at small noise ratios (e.g., from 0% to 15%). Future work should develop and investigate more hybrid noise-filtering criteria.

REFERENCES

[1] T. Mitchell, Machine Learning, vol. 4, no. 1. McGraw Hill, 1997.

[2] J. Quinlan, "C4. 5: programs for machine learning," Mach. Learn., vol. 240, p. 302, 1993.

[3] H. Liu and S. Zhang, "Noisy data elimination using mutual k-nearest neighbor for classification mining," J. Syst. Softw., vol. 85, no. 5, pp. 1067–1074, May 2012.

[4] F. Muhlenbach, S. T. Ephane, and D. A. Zighed, "Identifying and Handling Mislabelled Instances," J. Intell. Inf. Syst., vol. 22, no. 1, pp. 89–109, 2004.

[5] F. Pasquier, S. Delany, and P. Cunningham, "Blame-based noise reduction: An alternative perspective on noise reduction for lazy learning," Dublin, Trinity Coll. Dublin, Dep. Comput. Sci., pp. 1–17, 2005.

[6] N. Segata and E. Blanzieri, "Noise reduction for instance-based learning with a local maximal margin approach," J. Intell. Inf. Syst. 35, no. October, 2010.

[7] N. Segata, E. Blanzieri, and P. Cunningham, "A scalable noise reduction technique for large case-based systems," 8th Int. Conf. Case-based Reason. (ICCBR 09), vol. volume 565, pp. 328–342, 2009.

[8] X. Zeng and T. Martinez, "A Noise Filtering Method Using Neural Networks," in Soft Computing Techniques in Instrumentation, Measurement and Related Applications, 2003, pp. 26–31.

[9] O. Othman and C. H. Bryant, "Preceding Rule Induction with Instance Reduction Methods," Proc. 9th Int. Conf. Mach. Learn. Data Min.

Pattern Recognition. Lect. Notes Comput. Sci. , Springer-Verlag, Berlin, pp. 209–218, 2013.

[10] M. El Hindi, KAL-Akhras, "Smoothing decision boundaries to avoid overfitting in neural network training," Neural Netw. World, vol. 21, no. 4, pp. 311–326, 2011.

[11] K. El Hindi and M. Alakhras, "Eliminating border instance to avoid overfitting," Antonio Palma dos Reis (Ed.). Proceeding Intell. Syst. Agents, pp. 93–99 . IADIS press Algarve, Portugal, 2009.

[12] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," Mach. Learn., vol. 6, no. 1, pp. 37–66, Jan. 1991.

[13] D. R. Wilson and T. R. Martinez, "Reduction techniques for instance-based learning algorithms," Mach. Learn., vol. 38, no. 3, pp. 257–286, 2000.

[14] D. L. Wilson, "Asymptotic Properties of Nearest Neighbor Rules Using Edited Data," IEEE Trans. Syst. Man. Cybern., vol. 2, no. 3, pp. 408–421, 1972.

[15] I. Tomek, "An experiment with the edited nearest-neighbor rule," IEEE Trans. Syst. Man. Cybern., vol. 6, no. 6, pp. 448–452, 1976.

[16] "UCI Machine Learning Repository." [Online]. Available: http://archive.ics.uci.edu/ml/machine-learningdatabases/.

[17] H. Yin and H. Dong, "The Problem of Noise in Classification :Past,Current and Future work 1 2 1," in Communication Software and Networks (ICCSN), 2011, pp. 412–416.

[18] C. M. Teng, "A Comparison of Noise Handling Techniques," Proc. Fourteenth Int. Florida Artif. Intell. Res. Soc. Conf., pp. 269–273, 2001.

[19] J. R. Quinlan, "Induction of Decision Trees," Mach. Learn., pp. 81–106, 1986.

[20] P. Clark and T. Niblett, "The CN2 rule induction algorithm," Mach. Learn., vol. 3, no. 4, pp. 261–284, 1989.

[21] D. Gamberger, N. Lavrac, and S. Dzeroski, "Noise detection and elimination in data preprocessing: Experiments in medical domains," Appl. Artif. Intell. An Int. J., vol. 14, no. 2, pp. 205–223, 2000.

[22] C. M. Teng, "Correcting noisy data," Proc.\ Intl.\ Conf.\ Mach. Learn., 1999.

[23] C. M. Teng, "Polishing blemishes: issues in data correction," IEEE Intell. Syst., vol. 19, no. 2, pp. 34–39, 2004.

[24] Y. Yang, X. Wu, and X. Zhu, "Dealing with predictive-but-unpredictable attributes in noisy data sources," Proc. 8th Eur. Conf. Princ. Pract. Knowl. Discov. databases, Pisa, Italy, 2004.

[25] C. E. Brodley and M. A. Friedl, "Identifying Mislabeled Training Data," J. Artif. Intell. Res., vol. 11, pp. 131–167, 1999.

[26] S. Massie, S. Craw, and N. Wiratunga, "When Similar Problems Don ' t Have Similar Solutions," Proc. 7th Int. Conf. Case-Based Reason. (ICCBR 07), Springer-Verlag, Berlin, Heidelb., pp. 92–106, 2007.

[27] S. J. Delany, N. Segata, and B. Mac Namee, "Profiling instances in noise reduction," Knowledge-Based Syst., vol. 31, pp. 28–40, Jul. 2012.

[28] B. Frénay and M. Verleysen, "Classification in the Presence of Label Noise : a Survey," Neural Networks Learn. Syst. IEEE Trans., vol. 25, no. 5, pp. 845–869, 2014.

[29] S. Verbaeten and A. Van Assche, "Ensemble Methods for Noise Elimination in Classification Problems," Proc. 4th Int. Conf. Mult. Classif. Syst., vol. 2709, pp. 317–325, 2003.

[30] S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas, "Machine learning: a review of classification and combining techniques," Artif. Intell. Rev., vol. 26, no. 3, pp. 159–190, Nov. 2007.

[31] G. H. G. John and P. Langley, "Estimating Continuous Distributions in Bayesian Classifiers," in In Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence. Montreal, Quebec, Canada, 1995, vol. 1, pp. 338–345.

[32] P. Domingos and M. Pazzani, "On the Optimality of the Simple Bayesian Classifier under Zero-One Los," Mach. Learn., vol. 29, no. 2–3, pp. 103–130, 1997.

[33] D. F. Nettleton, A. Orriols-Puig, and A. Fornells, "A study of the effect of different types of noise on the precision of supervised learning techniques," Artif. Intell. Rev., vol. 33, no. 4, pp. 275–306, Jan. 2010.

[34] K. El Hindi, "A noise tolerant fine tuning algorithm for the Naïve Bayesian learning algorithm," J. King Saud Univ., vol. 26, pp. 237–246, 2014.

[35] Y. Yang, Y. Xia, Y. Chi, and R. Muntz, "Learning naive Bayes classifier from noisy data," … Calif. Los Angeles, Dep. …, no. 030056, pp. 1–19, 2003.

[36] K. El Hindi, "Fine tuning the Naïve Bayesian learning algorithm," AI Communcations, vol. 27, pp. 133–141, 2014.

[37] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," Inf. Process. Manag., vol. 45, no. 4, pp. 427–437, 2009.

[38] D. R. Wilson and T. R. Martinez, "Improved Heterogeneous Distance Functions," J. Artif. Intell. Res., vol. 6, pp. 1–34, 1997.

[39] T. Fawcett, "ROC Graphs: Notes and Practical Considerations for Data Mining Researchers," 2003.

[40] U. of Waikato, "WEKA: The Waikato Environment for Knowledge Acquisition." [Online]. Available: http://www.cs.waikato.ac.nz/ml/weka/.