Dealing with Predictive-but-Unpredictable Attributes in Noisy Data Sources

Ying Yang, Xindong Wu, and Xingquan Zhu

Department of Computer Science, University of Vermont, Burlington VT 05405, USA
{yyang,xwu,xqzhu}@cs.uvm.edu

Abstract. Attribute noise can affect classification learning. Previous work in handling attribute noise has focused on those predictable attributes that can be predicted by the class and other attributes. However, attributes can often be predictive but unpredictable. Being predictive, they are essential to classification learning and it is important to handle their noise. Being unpredictable, they require strategies different from those of predictable attributes. This paper presents a study on identifying, cleansing and measuring noise for predictive-but-unpredictable attributes. New strategies are accordingly proposed. Both theoretical analysis and empirical evidence suggest that these strategies are more effective and more efficient than previous alternatives.

1 Introduction

Real-world data are seldom as perfect as we would like them to be. Except in the most structured environment, it is almost inevitable that data contain errors from a variety of corrupting processes, such as acquisition, transmission and transcription [11, 13]. The corrupted data, namely noise, usually have adverse impact on interpretations of the data, models created from the data, and decisions made based on the data [7]. Since a manual process is laborious, time consuming and itself prone to errors, effective and efficient approaches that automate noise handling are necessary [8].

This paper handles noise in the context of classification learning, which plays an active role in machine learning. In classification learning, data are composed of *instances*. Each instance is expressed by a vector of *attribute* values and a *class* label. We further differentiate attributes into three types: unpredictive, predictive-and-predictable and predictive-but-unpredictable. *Unpredictive* attributes are futile for or irrelevant to predicting the class. They can be discarded by feature selection methods prior to the learning. *Predictive* attributes are useful for predicting the class, among which *predictable* ones can be predicted by the class and other attributes while *unpredictable* ones cannot. We argue that since predictive attributes contribute to classification learning, it is important to handle their noise. However, because predictable attributes and unpredictable attributes have different natures, they require different strategies for noise handling. For convenience of expression, an 'attribute' throughout this paper implies a predictive attribute unless otherwise mentioned.

Generally, there are two types of noise, attribute noise and class noise. Most previous efforts are engaged in handling class noise, such as for contradictory instances (the same instances with different class labels) or misclassifications (instances labeled with a wrong class). Various achievements have been reported [2–6, 14, 15]. In comparison, much less attention has been paid to attribute noise. Ironically, attribute noise tends to happen more often in the real world. For example, if noise comes from entry mistakes, it is very likely that the class has fewer errors since the people involved know that it is the 'important' value and pay more attention to it [11].

Among few publications on handling attribute noise, an important contribution is LENS [7]. It aims at presenting a 'complete' understanding that helps identify, correct or prevent noise by modelling the generation of clean data, the generation of noise and the process of corruption, under the assumption that the noise generation and the corruption process have learnable underlying structures. If the assumption does not hold, the authors indicate that LENS has limited utility. Different from LENS, another often-cited achievement, polishing [12], deals with attribute noise without knowing its underlying structures. As we will detail later, polishing excels in handling noise for predictable attributes. A predictable attribute is one that can be predicted by the class and other attributes. We suggest that attributes in real-world applications can frequently be otherwise. It is because normally attributes are collected as long as they contribute to predicting the class. Whether or not one attribute itself can be predicted is often not a concern.

Hence, with all due respect to previous achievements, we suggest that the picture of handling attribute noise is incomplete. It is an open question how to address unpredictable attributes when their underlying noise structures are unavailable. These understandings motivate us to explore appropriate strategies that identify, cleanse and measure noise for predictive-but-unpredictable attributes. In particular, Section 2 introduces the background knowledge. Section 3 proposes *sifting* to identify noise. Section 4 discusses *deletion*, *uni-substitution* and *multi-substitution* to cleanse the identified noise. Section 5 studies *literal equivalence* and *conceptual equivalence* to measure noise. Section 6 empirically evaluates our new strategies. Section 7 gives a conclusion and suggests further research topics.

2 Background Knowledge: Polishing

Polishing [12] is an effective approach to handling noise for predictable attributes. When identifying noise for an attribute A, polishing swaps A with the original class, and uses cross validation to predict A's value for each instance. Those values that are mis-predicted are identified as noise candidates. It then replaces each noise candidate by its predicted value. If the modified instance is supported by a group of classifiers learned from the original data, the modification is retained. Otherwise, it is discarded. Polishing assumes that since one can predict the class by attribute values, one can turn the process around and use the class

Table 1. Prediction accuracies of classes and attributes in polishing's data. For instance, mushroom has 100.0% in the 'Class' column indicating that the prediction accuracy for its class is 100.0%. Mushroom has 10 in the column [90%,100%] of 'Attributes' indicating that it has 10 attributes whose prediction accuracies fall into [90%,100%].

Dataset	Class	Attributes			
		[90%,100%]	[70%,90%)	[50%,70%)	[0%,50%)
mushroom	100.0%	10	4	5	3
soybean	91.5%	27	3	3	2
led-24	100.0%	7	0	17	0
vote	96.3%	3	10	3	0
audiology	77.9%	58	9	0	2
promoters	81.1%	0	0	3	54

together with some attributes to predict another attribute's values as long as this attribute is predictive.

However, this 'turn around' can be less justifiable when unpredictable attributes are involved. For example, the often-cited monk's problems from the UCI data repository [1] have 6 attributes $A_1, \ldots, A_6$ and a binary class. One underlying concept is ($A_4 = 1$ and $A_5 = 3$) or ($A_2 \neq 3$ and $A_5 \neq 4$) $\Rightarrow C = 1$; otherwise $C = 0$. Accordingly A_5 is a predictive attribute. Although the class can be predicted with 100% accuracy by C4.5trees [9] using 10-fold cross validation, the prediction accuracy for A_5 is only 35% because it has multiple values mapped to a single class. Directed by such a low accuracy, the noise identification has a strong potential to be unreliable. More datasets involving predictive-but-unpredictable attributes will be shown in Section 6.

Polishing has reported favorable experimental results. Nonetheless, they apply to predictable attributes. For each of polishing's datasets, Table 1 summarizes prediction accuracies for its class and attributes[1]. In all datasets except 'promoters', highly predictable attributes dominate the data[2]. In many cases, attributes are even more predictable than the original class. The 'promoters' data, which are not dominated by predictable attributes, produce a less favorable result for polishing.

These observations by no means devalue polishing's key contribution to handling predictable attributes. Nonetheless they raise the concern of polishing's suitability for unpredictable attributes and inspire our further study.

3 Identifying Noise

We propose a *sifting* approach to identifying noise for unpredictable attributes. For a predictable attribute, there exists a value whose probability given an in-

[1] We summarize them because many datasets have too many attributes to be listed individually. For instance, audiology has 69 attributes. Each accuracy results from C4.5trees using 10-fold cross validation, as polishing did.

[2] For the 'vote' dataset, although 17 attributes fall in [50%,70%], these attributes are randomly-added unpredictive attributes.

stance is high enough to dominate alternative values. For an unpredictable attribute, there is often no such dominating value. Instead, multiple values may be valid given an instance.

To identify noise for an attribute, polishing predicts what the clean value is and identify other values suspicious. This strategy is less appropriate if the attribute is unpredictable. For example, we have instances to represent apples and berries. One attribute is color. Apples can be green, yellow or red. Berries can be blue, black or red. If green apples happen to have a slightly higher occurrence, polishing will identify valid instances like $< \cdots, yellow, apple, \cdots >$ and $< \cdots, red, apple, \cdots >$ suspicious since the predicted color of an apple is green.

Neither is it adequate to *individually* identify noise for each attribute, where an attribute is separated from the others, swaped with the class and predicted by other possibly noisy attributes. Suppose an instance to be $< \cdots, black, apple, \cdots >$. When the attribute 'color' is under identification, its value 'black' will be identified suspicious since it should be 'green' given 'apple'. Meanwhile, when the attribute 'fruit' is under identification, its value 'apple' will be identified suspicious since it should be 'berry' given 'black'. Thus both values are identified. However, it is very likely that only one is real noise. The original instance can be a 'black berry' or a 'green apple'. This *compounded suspicion* is caused by identifying noise according to noisy evidence.

Accordingly sifting evaluates whether the pattern presented by a *whole* instance is suspicious instead of judging isolated values. It identifies an instance suspicious *only when* this instance does not satisfy *any* pattern that is learned with certain confidence from the data. We name this strategy 'sifting' because it takes the set of learned patterns as a sifter to sift instances. Instances that match any pattern may go through while the remaining are identified suspicious. By this means, an unpredictable attribute is not forced to comply with a single value. Instead, its multiple valid values can be allowed. For instance, colors of green, yellow and red are all allowable for an apple since we can learn those patterns. A black apple will be identified suspicious since its pattern is very unlikely, and this anomaly is efficiently identified in one go.

Algorithm 1 shows an implementation of *sifting*. Please be noted that we are dealing with situations where the underlying structure of noise (if there is any at all) is not available. This implies that if there is any learnable pattern, it reflects the knowledge of clean data. Furthermore, the learned patterns are in terms of classification rules (with the class, but not any attribute, on the right hand side).

We expect sifting to obtain a subset of instances with a high concentration of noise. We then forward this subset to the next process: cleansing.

4 Cleansing Noise

One should be very cautious when coming to cleanse an instance. For example, the data can be completely noise-free but an inappropriate rule learner is employed. Hence, we suggest that noise cleansing is conducted only when the data's

Algorithm 1: Identifying noise for unpredictable attributes

```
Input: possibly noisy dataset $D$; a rule learner $L$;
Output: a set of suspicious instances $IS$;
Begin
  $RS$ = a set of rules that $L$ learns from $D$;
  foreach Instance $I_i \in D$
    $flag = 0$;
    foreach Rule $R_j \in RS$
      if $I_i$ satisfies $R_j$   {$flag = 1$; break;}
    if $flag == 0$   {push $I_i$ into $IS$;}
End
```

genuine concept is learnable; practically, only when a high prediction accuracy is achievable.

For predictive attributes, it is sensible to only permit changes towards the predicted values, as polishing does. For unpredictable attributes, usually no value can be predicted with such a high accuracy as to exclude alternative values. Accordingly, we study three cleansing approaches, *deletion*, *uni-substitution* and *multi-substitution*.

As delineated in Algorithm 2, deletion simply deletes all identified noisy instances. At first sight, it most likely causes information loss. Nevertheless, this intuition needs to be verified. Uni-substitution cleanses an instance by the rule that minimizes the number of value changes. If there are multiple such rules, uni-substitution filters them by their quality indicators[3] and chooses a single one to cleanse the suspicious instance. Hence uni-substitution maintains the original data amount. Multi-substitution is the same as uni-substitution except at the final stage. If finally several rules are still equally qualified, multi-substitution produces multiple cleansed instances, each corresponding to a rule, and substitutes all of them for the suspicious instance. In this way, it may increase the data amount. But it has a merit that retrieves all valid values.

5 Measuring Noise

It is important to measure how noisy the corrupted data are if compared against its clean version. Otherwise, it is difficult to evaluate the effectiveness of a noise handling mechanism. A common measurement is *literal equivalence* [12, 15]. We believe that it is often of limited utility and propose *conceptual equivalence* instead.

5.1 Literal Equivalence

Suppose CD is a clean dataset and is corrupted into a noisy dataset ND. When measuring noisy instances in ND, literal equivalence conducts a literal comparison. Two common mechanisms are *match-corresponding* and *match-anyone*.

[3] Normally the rule learner attaches quality indicators to each rule. For example, we employ C4.5rules that attaches each rule with *confidence*, *coverage* and *advantage*.

Algorithm 2: Cleansing noise for unpredictable attributes

```
Input: a set of suspicious instances $IS$ identified by a rule set $RS$ in a possibly noisy dataset $D$;
Output: a cleansed dataset $D$;
Begin
  if $cleanse ==$ deletion
    $D = D - IS$;
    return $D$;
  foreach Instance $I_i \in IS$
    foreach Rule $R_j \in RS$
      $changes$ = numbers of attribute values to be changed to make $I_i$ satisfy $R_j$;
    $candidates$ = set of rules with the smallest $changes$;
    // Further filter by rules' quality
    if $|candidates| > 1$  {$candidates$ = rules in $candidates$ with highest $confidence$;}
    if $|candidates| > 1$  {$candidates$ = rules in $candidates$ with highest $coverage$;}
    if $|candidates| > 1$  {$candidates$ = rules in $candidates$ with highest $advantage$;}
    if $cleanse ==$ uni-substitution
      $I'$ = change $I_i$ to satisfy the first rule[a] in $candidates$;
      $D = D - \{I_i\}$;
      $D = D + \{I'\}$;
    if $cleanse ==$ multi-substitution
      $D = D - \{I_i\}$;
      foreach Rule $R_c$ in $candidates$
        $I'$ = change $I_i$ to satisfy $R_c$;
        $D = D + \{I'\}$;
  return $D$;
End
```

[a] Or a randomly-chosen rule in *candidates*. Since normally the order of the rules implies some overall ranking made by the learner, we here have always chosen the first one.

Match-corresponding. For the ith instance in ND, I_i, match-corresponding compares it with the ith instance I'_i in CD. If I_i matches I'_i, I_i is clean. Otherwise I_i is noisy. Although it is straightforward, match-corresponding has very limited function. The reason is that it is sensitive to the order of instances. As illustrated in Figure 1, (a) is a clean dataset with n instances I_1 to I_n. Suppose that there are no identical instances. Now we shift as in (b) each instance one location left so that I_i takes the location of I_{i-1} for any $i \in [2, n]$ and I_1 takes the location of I_n. The datasets (a) and (b) are the same. But match-corresponding will judge (b) as 100% noise since no instance matches its corresponding one in (a).

(a)
I_1 I_2 • • • I_{n-1} I_n
(b)
I_2 I_3 • • • I_n I_1

Fig. 1. Match-corresponding is sensitive to the order.

Match-anyone. For an instance I_i in ND, as long as it can match anyone of CD's instances, match-anyone deems it clean. Only if it does not appear in CD at all will I_i be judged noisy. Although it is less censorious, match-anyone can be insensitive to information loss. An extreme example is that an algorithm obtains a rule complying with a single clean instance I_1 and cleanses all other instances into I_1 as in (b) of Figure 2. Although (b) has significantly lost information of (a), match-anyone still judges the cleansing superb with 100% success.

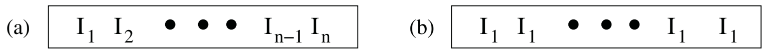

(a)
I_1 I_2 • • • I_{n-1} I_n
(b)
I_1 I_1 • • • I_1 I_1

Fig. 2. Match-anyone is insensitive to information loss.

5.2 Conceptual Equivalence

Other mechanisms of literal equivalence may well exist. However, all measurements involving literal comparison suffer from a problem that they confine the data's legitimacy to the clean dataset at hand. Usually a dataset is only a sample of the whole population. Instances can legitimately exist in the whole population but do not appear in certain samples. This motivates us to propose *conceptual equivalence*, which we expect to be more elegant and capable in measuring noise. Suppose a clean dataset CD is corrupted and is then cleansed into a dataset CD'. Suppose the target concepts learned from CD and CD' are $Concept_{CD}$ and $Concept_{CD'}$ respectively. In order to evaluate how well CD' resembles CD, conceptual equivalence will cross-exam how well CD's data support $Concept_{CD'}$ and how well CD''s data support $Concept_{CD}$. The better both data support each other's concept, the less noise in CD' and the higher the conceptual equivalence. The process is illustrated in Figure 3.

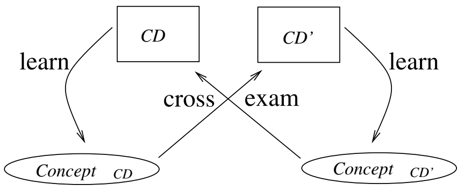


Fig. 3. Conceptual equivalence conducts cross-exam.

6 Experiments

Experiments are conducted to test three hypotheses for unpredictable attributes: (1) our new strategy identifies noise more accurately and efficiently than polishing; (2) if the data's genuine concept is learnable and hence cleansing is allowed, our new strategy cleanses noise more effectively than polishing; and (3) conceptual equivalence is more appropriate than literal equivalence to measure noise. Please be noted that we do not claim that our new strategies outperform polishing for predictable attributes. Instead, the niche of our methodology lies within unpredictable attributes.

6.1 Data and Design

We choose a dataset from the UCI data repository [1] if it satisfies the following conditions. First, its genuine concept is documented. We do not use this information during any stage of identifying or cleansing. It only helps us check the types of attributes and verify our analysis. Second, its genuine concept is learnable, that is, the prediction accuracy is high. Otherwise, we will not be able to differentiate the clean from the noisy. Third, the attributes are predictive but unpredictable. Here we deem an attribute unpredictable if its prediction accuracy is significantly lower than the class. Fourth, there are no numeric attributes. Our approaches can apply to numeric attributes if discretization is employed. However, discretization may introduce extra intractable noise and compromises our understanding of the experimental results. The resulting 5 (3 natural, 2 synthetic) datasets' statistics and prediction accuracies for the class and attributes are in Table 2.

We use C4.5rules [9] as the rule learner for both noise identification and cleansing. Since originally polishing employs C4.5trees, we re-implement polishing using C4.5rules to ensure a fair comparison. Each attribute is randomly corrupted, where each value other than the original value is equally likely to be chosen. A noise level of $x\%$ indicates that $x\%$ instances are corrupted. Each original dataset is corrupted into four levels: 10%, 20%, 30% and 40% respectively.

Table 2. Experimental datasets.

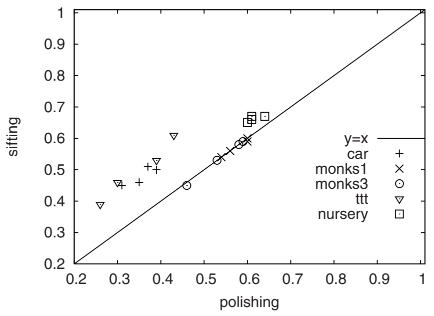
Data	Size	Class	Attribute	Prediction accuracy (%)	
set		No.	No.	Class	Attributes
car	1728	4	6	92.4	29.3, 30.7, 23.2, 46.8, 36.6, 52.4
monks1	432	2	6	96.5	51.2, 48.1, 41.0, 23.1, 35.9, 44.7
monks3	432	2	6	100.0	23.1, 51.2, 41.2, 28.5, 35.4, 44.2
ttt[a]	958	2	9	85.1	51.0, 48.9, 50.8, 49.6, 62.7, 49.1, 49.8, 49.3, 51.0
nursery	12960	5	8	97.1	42.8, 27.7, 24.2, 25.5, 37.5, 50.8, 34.4, 75.9

[a] The ttt dataset represents the tic-tac-toe dataset.

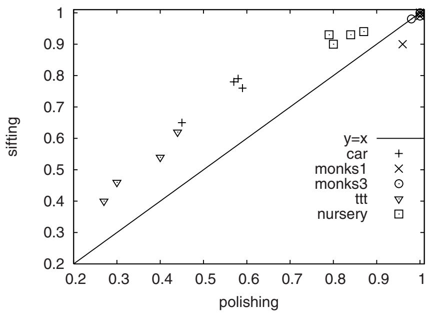
6.2 Identification

The identification performance is measured by $F1$ *measure* [10], a popular measure used in information retrieval that evenly combines precision and recall of the identification. Precision p reflects the purity of identification. It equals the number of truly noisy instances identified divided by the total number of identified instances. Recall r reflects the completeness of identification. It equals to the number of truly noisy instances identified divided by the total number of truly noisy instances. $F1(p, r)$ equals to $\frac{2pr}{p+r}$, which falls in the range $[0, 1]$. The higher a method's $F1$ *measure*, the better this method simultaneously maximizes both precision and recall. The results for all 20 cases (5 datasets, each with 4 levels of corruption) are depicted in Figure 4. No matter whether one uses match-corresponding or uses match-anyone[4], sifting achieves a higher $F1$ *measure* in almost all cases. This indicates that sifting outperforms polishing in identifying noise for unpredictable attributes.

[4] Conceptual equivalence does not apply here since no data have been cleansed yet.



(a) Match-corresponding



(b) Match-anyone

Fig. 4. The $F1$ *measure* measures sifting and polishing's identification. Each coordinate represents an experimental case. Its x component represents polishing's $F1$ *measure* and its y component represents sifting's $F1$ *measure*. Hence, any coordinate above the $y = x$ line indicates a case where sifting achieves a higher $F1$ *measure* than polishing does.

6.3 Cleansing

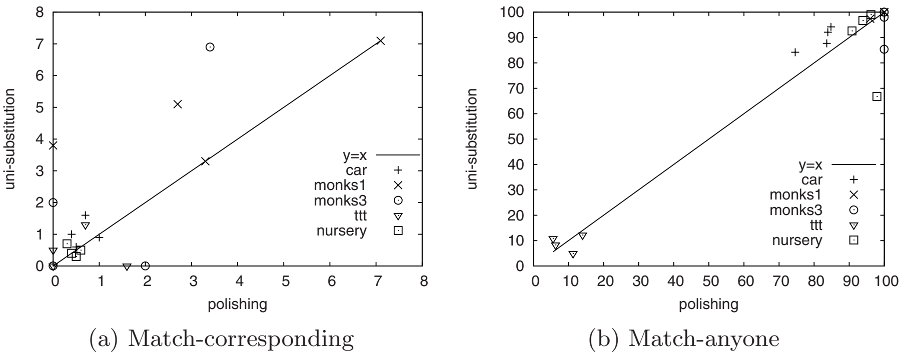
The cleansing performance under literal and conceptual equivalence is studied here.

Under literal equivalence. Figure 5 records *correction accuracy* of uni-substitution and polishing[5] in each of 20 cases, which equals to the number of correctly cleansed instances divided by the total number of identified instances. Although uni-substitution achieves higher correction accuracies more often than not, the results also confirm our belief that literal equivalence tends to improperly reflect the cleansing performance. Correction accuracies under match-corresponding are always low (below 10%) since match-corresponding is sensitive to the instances' order and is over censorious. Correction accuracies under match-anyone are far higher since match-anyone can be insensitive to information loss and has a potential to exaggerate the cleansing efficacy.

However, ttt is an exception that obtains a low accuracy even under match-anyone. Its clean data encode the complete set of legal 3x3 board configurations at the end of tic-tac-toe games and contain 958 instances. But the corruption does not have the legitimacy in mind and has $3^9 = 19683$ configurations at choice. Few can be restored to completely match a clean instance. This raises another type of situation where literal equivalence can not manage well.

Under conceptual equivalence. There can be different ways to calculate conceptual equivalence. Table 3 reports the results of our implementation. We first learn classification rules from clean data and use them to classify

[5] Correction accuracy here does not apply to deletion or multi-substitution since they do not maintain the original data amount.



(a) Match-corresponding (b) Match-anyone

Fig. 5. Under literal equivalence, the *correction accuracy*(%) measures polishing and uni-substitution's cleansing. Each coordinate represents an experimental case. Its x component represents polishing's *correction accuracy* and its y component represents uni-substitution's *correction accuracy*. Hence, any coordinate above the $y = x$ line indicates a case where uni-substitution achieves a higher *correction accuracy* than polishing does. The axis value ranges of (a) and (b) are different. The correction accuracies under match-corresponding are far lower than those under match-anyone.

cleansed data, obtaining a classification accuracy acc_1. We then learn classification rules from cleansed data and use them to classify clean data, obtaining another classification accuracy acc_2. Because acc_1 and acc_2 can be associated with different data sizes ($size_1$ and $size_2$ respectively)[6], their *weighted mean* $= \sum_{i=1}^{2}(acc_i \times size_i)/\sum_{i=1}^{2} size_i$ is used to indicate the degree of conceptual equivalence as in (a). Graphs are drawn in (b) and (c) to better reflect the trend of the values. The pattern in (b) corresponds to 'car', which is representative of other three datasets, monks1, monks2 and nursery. The pattern in (c) corresponds to 'ttt', which is a little different from others because of ttt's special nature as we have explained in the previous section.

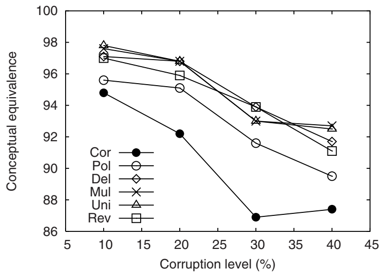
Compared with corrupted data, all of deletion, uni-substitution and multi-substitution achieve higher conceptual equivalence across all datasets at all noise levels. This suggests that our handling helps improve the quality of the corrupted data. Compared among themselves, there is no significant difference. An interesting observation is that, despite the risk of losing information, deletion works surprisingly well. A closer look reveals that deletion gains this advantage mainly through large datasets like nursery. This suggests that when the available data well exceed the amount that is needed to learn the underlying concept, appropriately deleting suspicious noise may not harm the genuine reflection of the concept while may effectively eliminate the chance of introducing new noise. As for multi-substitution, we have observed that multiple candidates do not often happen. It is because our process of filtering is very fine as given in Algorithm 2. One can make it coarser and may observe more differences between multi-substitution and uni-substitution.

[6] For example, if multi-substitution is used, the size of cleansed data can be bigger than that of clean data.

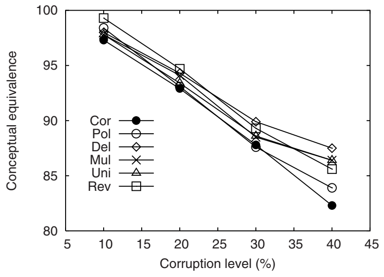
Table 3. Cleansing performance under conceptual equivalence (%).

Noise level	Data set	Cor	Del	Mul	Uni	Pol	Rev
10%	car	94.8	97.1	97.6	97.8	95.6	97.0
	monks1	98.4	100.0	100.0	100.0	100.0	100.0
	monks3	98.3	100.0	100.0	100.0	100.0	100.0
	ttt	97.3	98.0	97.8	97.8	98.4	99.3
	nursery	96.6	99.3	99.2	99.2	99.0	99.1
20%	car	92.2	96.8	96.8	96.8	95.1	95.9
	monks1	96.6	100.0	100.0	100.0	100.0	100.0
	monks3	96.7	100.0	100.0	100.0	100.0	100.0
	ttt	92.9	94.3	94.1	93.4	93.1	94.7
	nursery	93.7	99.0	98.8	98.8	97.9	98.8
30%	car	86.9	93.9	93.0	93.0	91.6	93.9
	monks1	93.8	100.0	100.0	100.0	100.0	100.0
	monks3	93.9	99.5	99.4	99.4	99.5	99.5
	ttt	87.8	89.9	88.5	88.6	87.6	89.3
	nursery	90.4	98.5	95.5	95.6	96.9	98.0
40%	car	87.4	91.7	92.7	92.5	89.5	91.1
	monks1	91.3	99.2	98.8	98.8	99.7	97.8
	monks3	94.4	99.9	98.4	98.4	100.0	99.9
	ttt	82.3	87.5	86.4	86.4	83.9	85.6
	nursery	87.3	97.3	96.5	96.6	94.3	96.6
Mean		92.6	97.1	96.7	96.6	96.1	96.8
Geomean		92.5	97.0	96.6	96.6	96.0	96.7

(a) summary



(b) car



(c) ttt

Note: Each method's conceptual equivalence is calculated between the clean data and the data processed by this method. 'Cor' is corruption; 'Del' is deletion; 'Mul' is multi-substitution; 'Uni' is uni-substitution; 'Pol' is polishing; and 'Rev' is the revised version of polishing that is supplied with sifting's identification. The 'Mean' row and 'Geomean' row record its arithmetic mean and geometric mean across different datasets.

Compared with polishing, all of our cleansing methods achieve higher conceptual equivalence more often than not. Their arithmetic and geometric means are also higher than polishing's. Nevertheless, we do not jump to the conclusion that polishing's cleansing, which is sophisticated, is inferior. It is possible that the disadvantage of its identification is passed on to its cleansing. Hence, we implement a revised version of polishing whose cleansing is supplied with sifting's identification. Thus we can have a pure comparison between the cleansing performances. The experimental results show that revised polishing either improves on polishing or maintains polishing's high conceptual equivalence (like 100%) in 18 out of 20 cases. This from another perspective verifies that sifting is effective.

The revised polishing thus can obtain competitive conceptual equivalence. However, in terms of efficiency, our new strategies are superior to polishing. Suppose the number of instances and attributes to be I and A respectively. Suppose the rule learning algorithm's time complexity and the number of learned rules to be $O(L)$ and R respectively. For each attribute, polishing conducts cross validation to predict its value for each instance. It then recursively tries out different combinations of attribute changes for each instance. It reaches a prohibitive time complexity of $O(ALI) + O(I2^A)$. In comparison, our cleansing needs to learn a rule set for once and match each instance against this rule set. Thus it has a time complexity of $O(L) + O(IR)$. Hence our methods are far more efficient than polishing, which has been verified by the experimental running time and is important in the real world where large data are routinely involved.

7 Conclusion

This paper handles predictive-but-unpredictable attributes in noisy data sources. To identify noise, we have proposed *sifting*. To cleanse noise, we have suggested that unless the genuine concept can be reliably learned, one should be very cautious to modify an instance. When the genuine concept is learnable, we have studied three cleansing approaches, *deletion*, *uni-substitution* and *multi-substitution*. To measure noise, we have argued that *literal equivalence* is often inadvisable and proposed *conceptual equivalence*. Both theoretical analysis and empirical evidence have demonstrated that our strategies achieve better efficacy and efficiency than previous alternatives.

The knowledge acquired by our study, although preliminary, is informative. We expect it to contribute to completing the picture of attribute noise handling. However, a single study seldom settles an issue once and for all. More efforts are needed to further advance this research field. We name three topics here.

First, whether an attribute is predictable or unpredictable is a matter of degree. In our current research, we deem an attribute unpredictable when its prediction accuracy is significantly lower than the class. Further research to work out more sophisticated thresholds or heuristics would be interesting. Second, although we take polishing as straw man, sifting does not claim to outperform polishing for predictable attributes. Instead, sifting and polishing are parallel, each having its own niche to work. Hence it is sensible to combine them. A work frame might be: (1) use feature selection to discard unpredictive attributes; (2) decide whether a predictive attribute is predictable; (3) if it is predictable, use polishing to handle its noise; and if it is unpredictable, use sifting to handle its noise. Lastly, it would be enchanting to extend our research beyond classification learning, such as to association learning where patterns exist but attributes are seldom predictable.

Acknowledgement

This research has been supported by the U.S. Army Research Laboratory and the U.S. Army Research Office under the grant number DAAD19-02-1-0178.

References

1. Blake, C. L., and Merz, C. J. UCI repository of machine learning databases, Department of Information and Computer Science, University of California, Irvine, 1998.
2. Brodley, C. E., and Friedl, M. A. Identifying and eliminating mislabeled training instances. In *Proc. of the 13th National Conf. on Artificial Intelligence* (1996), pp. 799–805.
3. Brodley, C. E., and Friedl, M. A. Identifying mislabeled training data. *Journal of Artificial Intelligence Research 11* (1999), 131–167.
4. Gamberger, D., Lavrac, N., and Dzeroski, S. Noise detection and elimination in data preprocessing: experiments in medical domains. *Applied Artificial Intelligence 14* (2000), 205–223.
5. Gamberger, D., Lavrac, N., and Groselj, C. Experiments with noise filtering in a medical domain. In *Proc. of the 16th International Conf. on Machine Learning* (1999), pp. 143–151.
6. Guyon, I., Matic, N., and Vapnik, V. *Discovering Informative Patterns and Data Cleaning.* AAAI/MIT Press, 1996, pp. 181–203.
7. Kubica, J., and Moore, A. Probabilistic noise identification and data cleaning. In *Proc. of the 3rd IEEE International Conf. on Data Mining* (2003), pp. 131–138.
8. Maletic, J. I., and Marcus, A. Data cleansing: Beyond integrity analysis. In *Proc. of the 5th Conf. on Information Quality* (2000), pp. 200–209.
9. Quinlan, J. R. *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers, 1993.
10. Rijsbergen, C. J. v. *Information Retrieval, second edition.* Butterworths, 1979.
11. Schwarm, S., and Wolfman, S. Cleaning data with Bayesian methods, 2000. Final project report for CSE574, University of Washington.
12. Teng, C. M. Correcting noisy data. In *Proc. of the 16th International Conf. on Machine Learning* (1999), pp. 239–248.
13. Teng, C. M. Applying noise handling techniques to genomic data: A case study. In *Proc. of the 3rd IEEE International Conf. on Data Mining* (2003), pp. 743–746.
14. Verbaeten, S. Identifying mislabeled training examples in ILP classification problems. In *Proc. of the 12th Belgian-Dutch Conf. on Machine Learning* (2002), pp. 1–8.
15. Zhu, X., Wu, X., and Chen, Q. Eliminating class noise in large datasets. In *Proc. of the 20th International Conf. on Machine Learning* (2003), pp. 920–927.