Applied Artificial Intelligence

ISSN: 0883-9514 (Print) 1087-6545 (Online) Journal homepage: http://china.tandfonline.com/loi/uaai20

Noise detection and elimination in data preprocessing: Experiments in medical domains

Dragan Gamberger , Nada Lavrac & Saso Dzeroski

To cite this article: Dragan Gamberger , Nada Lavrac & Saso Dzeroski (2000) Noise detection and elimination in data preprocessing: Experiments in medical domains, Applied Artificial Intelligence, 14:2, 205-223, DOI: 10.1080/088395100117124

To link to this article: http://dx.doi.org/10.1080/088395100117124

Applied Artificial Intelligence, 14:205–223, 2000

0883–9514/00 $12.00 + .00

NOISE DETECTION AND ELIMINATION IN DATA PREPROCESSING: EXPERIMENTS IN MEDICAL DOMAINS

DRAGAN GAMBERGER
Rudjer Bošković Institute, Zagreb, Croatia

NADA LAVRAČ and SAŠO DŽEROSKI
Jožef Stefan Institute, Ljubljana, Slovenia

Compression measures used in inductive learners, such as measures based on the minimum description length principle, can be used as a basis for grading candidate hypotheses. Compression-based induction is suited also for handling noisy data. This paper shows that a simple compression measure can be used to detect noisy training examples, where noise is due to random classification errors. A technique is proposed in which noisy examples are detected and eliminated from the training set, and a hypothesis is then built from the set of remaining examples. This noise elimination method was applied to preprocess data for four machine-learning algorithms, and evaluated on selected medical domains.

INTRODUCTION

In an ideal inductive learning problem, the induced hypothesis H "agrees" with the classifications of training examples E and performs as a perfect classifier on yet unseen instances. In practice, however, it frequently happens that data given to the learner contain various kinds of errors, either random or systematic. Random errors are usually referred to as *noise*. Therefore, in most real-life problems the success of machine learning very much depends on the learner's noise-handling capability, i.e., its ability of appropriately dealing with noisy data. Although noise in training examples may be due to erroneous attribute values and erroneous class labels, machine-learning algorithms usually treat noisy examples as being mis-labeled.

It should be noted that the term noise used in this work does not refer only to errors in the data; as opposed to the standard terminology, noise in

Address correspondence to Dragan Gamberger, Rudjer Boškovic Institute, Bijenicka 54, 10000 Zagreb, Croatia.

this work refers to errors (incorrect examples), as well as outliers (correct examples representing some relatively rare subconcept of the target theory).

In this work, a target theory is defined as the source of correct examples. A learning task is to find, from a given set of training examples E, a model for the target theory. In an ideal situation, E consists of correct examples only, and the induced model, called a target hypothesis, is the representation of the target theory in the selected hypothesis language. In real-life problems, however, the training set E may include erroneous data.

The problem of noise handling has been extensively studied in attribute-value learning. This problem has been approached in different ways. Noise-handling mechanisms can be incorporated in search heuristics (e.g., Mingers, 1989) and in stopping criteria (e.g., Clark & Boswell, 1991) used in hypothesis construction. Hypotheses fulfilling stopping criteria may further be evaluated according to some quality measure, giving a preferential order of hypotheses. In addition, the induced hypotheses can be subjected to some form of postprocessing, such as postpruning and simplifying of decision trees (e.g., Mingers, 1989; Quinlan, 1987; Cestnik & Bratko, 1991). Compression measures (Muggleton et al., 1992), based on the minimum description length (MDL) principle (Rissanen, 1978), evaluate candidate hypotheses by a heuristic, integrating a measure of complexity (simplicity or understandability) and correctness (expected accuracy). Compression measures can also be used for noise handling.

Systems employing any of the above techniques can be called *noise-tolerant* systems since they try to avoid overfitting the possibly noisy training set. A noise-handling technique proposed in this paper is different: it detects and eliminates noisy examples in preprocessing of the training set. The result of noise elimination in preprocessing is a reduced training set which is used as input to a machine-learning algorithm. The separation of noise detection and hypothesis formation has the advantage that noisy examples do not influence hypothesis construction (Gamberger et al., 1996). The explicit detection of potentially noisy examples allows us to show the examples to the expert, who can distinguish outliers from errors. A general hypothesis can then be built from error-free data (therefore better capturing the regularities of the domain), and outliers can be added to the hypothesis as exceptions to the general rule. Sometimes the separated set of potentially noisy examples itself can be interesting for inspection by domain experts and users of machine-learning algorithms.

The elimination of noisy examples has been proposed also by other authors. Noise and outlier detection and elimination have been extensively studied in statistics and in the research on nearest neighbor classifiers. Srinivasan et al. (1992) studied the problem of distinguishing exceptions from noise in inductive logic programming. Noise detection was addressed also in decision tree learning. An early decision tree learning algorithm assistant

(Cestnik et al., 1987) enabled learning from "good instances only, performed by eliminating from the training set the incorrect" examples misclassified by the incorporated naive Bayesian classifier. More recently, removal of outliers from data was also studied by John (1995). In the recent work of Brodley and Friedl (1996) the usefulness of the elimination of noisy examples in preprocessing has been undoubtly demonstrated. In their approach, one or more learning algorithms are used to create classifiers that serve as noise filters for training data, performing n-fold cross-validation on training examples. A training example is detected as noisy if it is misclassified by one or more classifiers when tested as a member of a test set.

The approach implemented in this work assumes a similar learning setting consisting of two steps: preprocessing of the training set that includes noise filtering, and a separate hypothesis formation step. In this work, hypothesis formation is performed using the ILLM algorithm (Gamberger, 1995) and the well-known algorithms C4.5 (Quinlan, 1993), *k*-NN (Wettschereck, 1994), and CN2 (Clark & Niblett, 1989; Clark & Boswell, 1991. In contrast to the work by Brodley and Friedl, our work proposes a compression-based approach to noise filtering: the use of a simple compression measure enables us to detect potentially noisy examples without actually constructing a hypothesis from the training set. The proposed approach to noise filtering is described in the next section. The section entitled experimental evaluation on UCI Medical Datasets presents the experimental setting and the results achieved by applying the proposed noise-detection method in medical diagnostic problems selected from the data repository at the University of Irvine. The final section evaluates the performance of the noise detection method on the problem of early diagnosis of rheumatic diseases.

NOISE DETECTION AND ELIMINATION

Representing Examples by Literals

We first consider a two-class learning problem where training set E consists of positive and negative examples of a concept ($E = P \cup N$) and examples e are tuples of truth-values of terms in the hypothesis language. The set of all terms, called *literals*, is denoted by L.

Let us represent the training set E as a table where rows correspond to training examples and columns correspond to literals. An element in the table has the value *true* when the example satisfies the condition (literal) in the column of the table, otherwise its value is *false*.

If the training set does not have the form of tuples of truth-values, a transformation to this form is performed in preprocessing of the training set. The transformation procedure is based on the analysis of values of training

examples. For each attribute A_i, let v_{ix} ($x = 1 \ldots k_{ip}$) be the k_{ip} different values of the attribute that appear in the positive examples and let w_{iy} ($y = 1 \ldots k_{in}$) be the k_{in} different values appearing in the negative examples. The transformation results in a set of literals L:

- For discrete attributes A_i, literals of the form $A_i = v_{ix}$ and $A_i \neq w_{iy}$ are generated.
- For continuous attributes A_i, literals of the form $A_i \leq (v_{ix} + w_{iy})/2$ are created for all neighboring value pairs (v_{ix}, w_{iy}) with the property $v_{ix} < w_{iy}$, and literals $A_i > (v_{ix} + w_{iy})/2$ for all neighboring pairs (w_{iy}, v_{ix}) with the property $w_{iy} < v_{ix}$. The motivation is similar to one suggested in (Fayyad & Irani, 1992).
- For integer-valued attributes A_i, literals are generated as if A_i were both discrete and continuous, resulting in literals of four different forms: $A_i \leq (v_{ix} + w_{iy})/2$, $A_i > (v_{ix} + w_{iy})/2$, $A_i = v_{ix}$, and $A_i \neq w_{iy}$.

The above procedure applies to propositional learning problems. On the other hand, when learning logical definitions of relations, a different procedure for generating literals is applied; see (Lavrač & Džeroski, 1994; Lavrač et al., 1995) presenting the LINUS literal generation procedure applicable to a class of relational learning problems.

p/n Pairs of Examples

For noise detection and elimination, we need to investigate the properties of literals that hold on individual pairs of training examples, each pair consisting of a positive and negative example.

Definition 1. A p/n pair denoted by (e_i, e_j) is a pair of two examples e_i and e_j such that $e_i \in P$ and $e_j \in N$.

Definition 2. A literal $l \in L$ covers a p/n pair (e_i, e_j) if the literal has value true for e_i and value false for e_j.[1]

The notion of p/n pairs can be used to prove important properties of literals for building complete and consistent concept descriptions (Gamberger & Lavrač, 1996; Gamberger & Lavrač, 1997).

Theorem 1. Assume a training set E and a set of literals L such that a complete and consistent hypothesis H can be found. Let $L' \subseteq L$. A complete and consistent concept description H can be found using only literals from the set L' if and only if for each possible p/n pair from the training set E there exists at least one literal $l \in L'$ that covers the p/n pair.

Proof. Proof of necessity: Suppose that the negation of the conclusion holds, i.e., that a p/n pair exists that is not covered by any literal $l \in L'$. Then no rule built of literals from L' will be able to distinguish between these two

examples. Consequently, a description which is both complete and consistent cannot be found.

Proof of sufficiency: Take a positive example p_i. Select from L' the subset of all literals L_i that cover p_i. A constructive proof of sufficiency can now be presented, based on k runs of a covering algorithm, where k is the cardinality of the set of positive examples, $k = |P|$. In the ith run, the algorithm learns a conjunctive description h_i, $h_i = l_{i\,1} \wedge \ldots \wedge l_{i\,m}$ for all $l_{i\,1}, \ldots l_{i\,m} \in L_i$ that are true for p_i. Each h_i will thus be true for p_i (h_i covers p_i), and false for all $n \in N$. After having formed all the k descriptions h_i, a resulting complete and consistent hypothesis can be constructed: $H = h_1 \vee \ldots \vee h_k$.

This theorem plays a central role in our noise elimination approach. Namely, if the set L is sufficient for constructing a complete and consistent hypothesis H from E, several such hypotheses may be found. Among these, one may prefer the simplest according to some complexity measure. Given the sets E and L, the minimal complexity of a hypothesis that uses literals from L and is complete and consistent w.r.t. E, denoted by $g(E, L)$ represents the so-called CLCH value (complexity of the least complex hypothesis, correct for all the examples in E). Because the set of literals L is assumed to be fixed in this presentation, $g(E)$ will be used in the rest of this paper as the function used for computing the CLCH value. One possible measure of complexity of a hypothesis H is the number of different literals that appear in it. This measure is very interesting because, as Theorem 1 suggests, there is a possibility to compute $g(E)$ without actually constructing a hypothesis H; in this case, the corresponding $g(E)$ value can be defined as the minimal number of literals $|L'|$ that are necessary to build a hypothesis that is correct for all the examples in E: $g(E) = |L'|$. This fact enables that the $g(E)$ value can be computed by any minimal-covering algorithm over the set of example pairs. In this work, the ILLM heuristic minimal covering algorithm is used (Gamberger, 1995).[2] The advantages of this approach are: $g(E)$ computation does not require the actual construction of a hypothesis and the $g(E)$ value can be determined relatively fast. The algorithm is presented in Figure 1.

The algorithm starts with the empty set of selected literals L' (step 1) and the set U' of yet uncovered p/n pairs equal to all possible pairs of one positive and one negative example from the training set (step 3), for which in (step 2) weights $v(e_i, e_j)$ have been computed. The weight of a pair is high if the pair is covered by a small number of distinct literals from L. The meaning of this measure is that for a pair with a high weight, it will be more difficult to find an appropriate literal that will cover this p/n pair than for a p/n pair with a small weight.

Each iteration of the main algorithm loop (steps 4–11) adds one literal to the minimal set L' (step 9). At the same time, all p/n pairs covered by the selected literal are eliminated from U' (step 10). The algorithm terminates

Algorithm 1: MINIMAL COVER
Input: U (set of p/n pairs), L (set of literals)
Output: L' (minimal set of literals)
(1) $L' \leftarrow \emptyset$
(2) **for** every $(e_i, e_j) \in U$ compute weights $v(e_i, e_j) = 1/z$,
where z is the number of literals $l \in L$ that cover (e_i, e_j)
(3) $U' \leftarrow U$
(4) **while** $U' \neq \emptyset$ **do**
(5) select $(e_a, e_b) \in U'$: $(e_a, e_b) = arg\ max\ v(e_i, e_j)$,
where max is over all $(e_i, e_j) \in U'$
(6) $L_{ab} \leftarrow \{l \mid l \in L$ covering $(e_a, e_b)\}$
(7) **for** every $l \in L_{ab}$ compute $w(l) = \sum v(e_i, e_j)$,
where sum is over all $(e_i, e_j) \in U'$ covered by l
(8) select literal l_s: $l_s = arg\ max\ w(l)$,
where max is over all $l \in L_{ab}$
(9) $L' \leftarrow L' \cup \{l_s\}$
(10) $U' \leftarrow U' \setminus \{$all (e_i, e_j) covered by $l_s\}$
(11) **end while**

FIGURE 1. Heuristic minimal covering algorithm.

when U' remains empty. In each iteration we try to select the literal that covers a maximal number of "heavy" p/n pairs (pairs with high weight). This is achieved so that a p/n pair (e_a, e_b) is detected which is covered by the least number of literals (step 5). At least one of the literals from the set L_{ab} with literals that cover this pair (step 6), must be included into the minimal set L'. To determine this literal, for each of them weight $w(l)$ is computed (step 7) and the literal with the maximal weight is selected (step 8). The weight of a literal is the sum of the weights of p/n pairs that are covered by the literal.

Noise Elimination Algorithm for Two-Class Problems

The approach to noise detection has its theoretical foundation in the saturation property of training data (Gamberger & Lavrač, 1997). It has been shown that if E is noiseless and saturated (containing enough training examples to find a correct target hypothesis), then $g(E) < g(E_n)$, where $E_n = E \cup \{e_n\}$ and e_n is a noisy example for which the target theory is not correct. The property $g(E) < g(E_n)$ means that noisy examples can be detected as those that enable CLCH value reduction. The approach in an iterative form is applicable also when E_n includes more than one noisy example.

The greatest practical problem in this approach is the computation of the CLCH value $g(E)$ for a training set E; this problem is solved by applying Theorem 1 and the presented heuristic minimal covering algorithm (Figure 1). The algorithm shown in Figure 2 presents the complete procedure for noise detection and elimination called the *saturation filter* since it is theoreti-

Algorithm 2: SATURATION FILTER
Input: $E = P \cup N$ (training set), L (set of literals)
Parameter: ε_h (noise sensitivity parameter)
Output: A (detected noisy subset of E), E' (noiseless subset of E)

(1) $E' \leftarrow E$
(2) $A \leftarrow \emptyset$
(3) **while** $E' \neq \emptyset$ **do**
(4) initialize $w(e) \leftarrow 0$ for all $e \in E'$
(5) find U, set of all possible p/n pairs for examples in E'
(6) call **Algorithm 1** to find minimal L', $L' \subseteq L$ such that
$\forall (e_i, e_j) \in U \ \exists \ l \in L'$ such that l covers (e_i, e_j)
(7) **for** every $l \in L'$ **do**
(8) $P^* \leftarrow \emptyset$, $N^* \leftarrow \emptyset$
(9) **for** every $(e_i, e_j) \in U$
(10) **if** (e_i, e_j) covered by l and no other literal from L' **then**
(11) $P^* \leftarrow P^* \cup \{e_i\}$, $N^* \leftarrow N^* \cup \{e_j\}$
(12) **end for**
(13) **if** $P^* = \emptyset$ **then** $L' \leftarrow L' \setminus \{l\}$ and **goto** step 7
(14) **for** every $e \in P^*$ **do** $w(e) \leftarrow w(e) + \frac{1}{|P^*|}$
(15) **for** every $e \in N^*$ **do** $w(e) \leftarrow w(e) + \frac{1}{|N^*|}$
(16) **end for**
(17) select example e_s: $e_s = arg\ max\ w(e)$, where max is computed over all $e \in E'$
(18) **if** $w(e_s) > \varepsilon_h$ **then**
(19) $A \leftarrow A \cup \{e_s\}$
(20) $E' \leftarrow E' \setminus \{e_s\}$
(21) **else** exit with generated sets A and E'
(22) **end while**

FIGURE 2. Saturation filter.

cally based on the saturation property of the training set (Gamberger & Lavrač, 1997). This algorithm, in its step (6), makes use of the heuristic minimal-covering algorithm.

Algorithm 2 begins with the reduced training set E' equal the input training set E (step 1) and an empty set of detected noisy examples A (step 2). The algorithm supposes that the set of all appropriate literals L for the domain is defined. U represents a set of all possible example pairs, where the first example in the pair is from the set of all positive training examples P' in the reduced set E', and the second example is from the set N' of all negative examples in the reduced training set E'. The algorithm detects one noisy example per iteration. The base for noise detection are weights $w(e)$ which are computed for each example e from E'. Initially all $w(e)$ values are initialized to 0 (step 4). At the end, the example with maximum weight $w(e)$ is selected (step 17). If the maximum $w(e)$ value is greater than the parameter ε_h predefined value, then the corresponding training example is included into

the set A (step 19) and eliminated from the reduced training set E' (step 20). The new iteration of noise detection begins with this reduced training set (steps 3–22). The algorithm terminates when in the last iteration no example has $w(e)$ greater than ε_h. Noisy examples in A and the noiseless E' are the output of the algorithm.

Computations in each iteration begin with the search for the minimal set of literals L' that cover all example pairs in U (calling Algorithm 1 in step 6). A pair of examples is covered by a literal l if the literal is evaluated *true* for the positive example and evaluated *false* for the negative example in the pair. This step represents the computation of the $g(E)'$ value. Next, a heuristic approach is used to compute weights $w(e)$ that measure the possibility that the elimination of an example e would enable $g(E')$ reduction. Weights $w(e)$ are computed so that for every literal l from L', minimal sets of positive ($P*$) and negative examples ($N*$) are determined, such that if $P*$ or $N*$ are eliminated from E' then l becomes unnecessary in L'. This is done in a loop (steps 9–12) in which every example pair is tested if it is covered by a single literal l. If such a pair is detected (step 10), then its positive example is included into the set $P*$ and its negative example into the set $N*$ (step 11). Literal elimination from L' presents the reduction of the $g(E')$ value. If a literal can be made unnecessary by the elimination of a very small subset of training examples, then this indicates that these examples might be noisy. In steps 14 and 15, the $w(e)$ weights are incremented only for the examples which are the members of the $P*$ and $N*$ sets. The weights are incremented by the inverse of the total number of examples in these sets. Weights are summed over all literals in L'. Step 13 is necessary because of the imperfectness of the heuristic minimal cover algorithm. Namely, if some $l \in L'$ exists for which there is no example pair that is covered only by this literal (i.e., for which either $P* = \emptyset$ or $N* = \emptyset$), this means that L' is actually not the minimal set because $L' \backslash \{l\}$ also covers all example pairs in U. In such case L' is substituted by $L' \backslash \{l\}$.

The presented saturation filter uses the parameter ε_h that determines noise sensitivity of the algorithm. The parameter can be adjusted by the user in order to tune the algorithm to the domain characteristics. Reasonable values are between 0.25 and 2. For instance, the value 1.0 guarantees the elimination of every such example by whose elimination the set L' will be reduced for at least one literal. Lower ε_h values mean greater sensitivity of the algorithm (i.e., elimination of more examples): lower ε_h values should be used when the domain noise is not completely random, and when dealing with large training sets (since statistical properties of noise distribution in large training sets can have similar effects). In ILLM (Gamberger, 1995) the default values of ε_h are between 0.5 and 1.5, depending on the number of training examples in the smaller of the two subsets of E: the set of positive examples P or the set of negative examples N. Default values for the satura–

tion filter's noise sensitivity parameter ε_h are: 1.5 for training sets with 2–50 examples, 1.0 for 51–100 examples, 0.75 for 101–200 examples, and 0.5 for more than 200 examples.

An alternative to using the described heuristic algorithm for noise filtering is the use of an exhaustive noise filtering algorithm (Gamberger & Lavrač, 1996). The exhaustive noise filtering algorithm presented in (Gamberger & Lavrač, 1997) would require the generation of a large number of E_x subsets of E' such that from 1 or up to 3 training examples are eliminated. For each of the subsets E_x, the value of $g(E'x)$ is computed and when the condition $g(E_x) < g(E')$ is fulfilled, then the difference $E' \setminus (E_x)$ represents the potentially noisy example(s). This approach is impractical because of the large number of required $g(E_x)$ computations.

Noise Elimination Algorithm for Multiclass Problems

The saturation filter works for two-class learning problems, where positive and negative examples of a single concept are described by literals (binary features). This section describes the multiclass saturation filter, performing noise elimination for disjoint multiclass problems.

Given an example set E of a multiclass learning problem, the elimination of noisy examples is performed as follows.

1. For each of the M classes c_j, create a two-class learning problem: examples that belong to class c_j become the positive examples for learning the concept c_j and all other examples become the negative examples of this concept. Each pair $(e_i, c_i) \in E$ is thus mapped into a new pair (c_i, c_{ij}), where $c_{ij} = 1$ if $c_i = c_j$ and $c_{ij} = 0$, otherwise.
2. Transform each pair (e_i, c_{ij}), where e_i is described by attribute values, into a pair $(f(e_i), c_{ij})$ where $f(e_i)$ is a tuple of truth values of literals (see the Section entitled Representing Examples by Literals describing this transformation). This results in new example sets $E_j, j = 1 \ldots M$.
3. For each of the M two-class learning problems, a set of noisy examples $A_j \subseteq E_j$ is detected by applying the saturation filter. Let $A'_j = \{e_i \mid f(e_i) \in A_j\}$.
4. Finally, the noisy examples A'_j of each E_j are eliminated from the original multiclass training set E. This results in a pruned training set

$$E' = E \setminus \bigcup_{j/\ 1 \ldots M} A'_j$$

A learning algorithm that assumes a noiseless training set can be now applied to the reduced training set E'.

EXPERIMENTAL EVALUATION ON UCI MEDICAL DATASETS

Experimental Setting

The goal of our experiments is to show the utility of noise elimination in learning from noisy datasets. The experiments were designed with the goal to verify whether the application of the multiclass saturation filter decreases the amount of random errors (misclassified training examples) in the dataset, and whether noise elimination improves the accuracy of induced hypotheses.

The experiments were performed using three machine-learning algorithms: inductive learning by logic minimization (ILLM) system (Gamberger, 1995), the C4.5 decision tree learning algorithm (Quinlan, 1993) and the Wettschereck's *k*-NN algorithm (Wettschereck, 1994). These algorithms can be used with and without their noise handling mechanisms. Inductive larning by logic minimization performs noise handling by using the saturation filter which is integrated into ILLM. In C4.5, noise handling is performed by pruning; no pruning means no noise handling. Noise handling in *k*-NN is achieved by the appropriate choice of the neighborhood of *k* nearest neighbors; 1-NN can be viewed as a variant of *k*-NN unsuited for dealing with noisy data.

Eight medical domains were chosen from the UCI repository (data repository at the University of Irwine) (Murphy & Aha, 1994). The reason for this choice is our interest in medicine and the fact that medical datasets represent real-world data usually containing substantial amounts of noise.

In each of the eight domains the evaluation was done using 10-fold stratified cross validation, as done by (Quinlan, 1996): the training instances were partitioned into 10 equal-sized blocks with similar class distributions. Each block in turn was then used as the test set for the classifier generated from the remaining nine blocks, and the average of the 10 results was reported.

When comparing the results of learning from nonfiltered and filtered data, the significance of accuracy difference was measured by requiring $p < 0.1$ using the two-tailed *t*-test for dependent samples.

Default values for the saturation filter's noise sensitivity parameter ε_h were: 1.5 for training sets with 2–50 examples, 1.0 for 51–100 examples, 0.75 for 101–200 examples, and 0.5 for more than 200 examples.

C4.5 was run with its default parameters when producing pruned trees (Quinlan, 1993) and with the parameter [− m 1] when producing unpruned trees. The *k*-NN algorithm was used in two ways: with and without feature-weights (Wettschereck, 1994). The *k* value used in the *k*-NN algorithm was automatically selected by the algorithm itself, using a leave-one-out strategy on the training set.

Results of the Experiments

ILLM Results

We first used the multiclass saturation filter to detect and eliminate noise from the training sets. The input to the algorithm were the original data sets (OrigEx) and the output were filtered data sets (FiltEx). We started with default values of the noise sensitivity parameter ε_h. The obtained reduced sets are denoted by FiltExR100. Next, the same procedure was repeated by setting ε_h to 75% of its default values. This change enabled the elimination of a larger amount of examples from the original training sets. The obtained reduced training sets are denoted by FiltExR75. In the same way, reduced training sets FiltExR50 and FiltExR25 for 50% and 25% of the default ε_h parameter values have been constructed.

The ILLM system without noise filtering was used to construct rules that are correct for all noncontradictory training examples in different training sets. Experiments were repeated for all OrigEx and different FiltEx training sets. The average prediction results of the obtained rules are summarized in Table 1.

The analysis of results in Table 1 shows that noise elimination, in general, improves the average prediction accuracy. This can be noticed especially from the last row with mean values. Mean prediction accuracy is higher for all training sets with different levels of noise elimination than for the original complete training sets. A rather great difference can be noticed by comparing mean results for R100 and R75 training sets on the one side, and R50 and R25 training sets on the other side. The possible conclusion is that, in general, medical domains include much noise and that the elimination of a larger number of potentially noisy examples is preferred.

The most important result is that in all eight domains noise elimination enabled prediction accuracy improvement, for seven of them the improvement is significant ($p < 0.1$ using two-tailed t-test for dependent samples),

TABLE 1 ILLM Results

	OrigEx	FiltEx R100	FiltEx R75	FiltEx R50	FiltEx R25
Breast	71.0	70.9	71.0	74.1	74.8
Cleveland	73.0	77.3	79.3	82.9	84.9
Echocardiogram	54.2	61.9	63.4	68.7	67.9
Hepatitis	71.5	71.6	71.6	70.5	71.1
Hungarian	75.6	75.6	74.9	76.3	79.4
Lymphography4	78.2	79.6	77.0	80.4	78.4
Thyroid	95.6	96.4	96.8	96.7	96.9
PrimaryTumor15	33.0	35.5	33.0	32.4	31.2
Mean	69.0	71.1	70.9	72.8	73.1

but for different domains the optimum is achieved by different ε_h parameter values. This means that the suggested approach to noise elimination is useful, but the results show that noise filtering should be used with caution since after the initial positive effects of noise elimination further elimination of training examples may result in decreased prediction accuracy. For example, in the *Cleveland* domain we have detected the increase in prediction accuracy by further elimination of training examples from the training set (the optimum is achieved by using the R25 sets or by using some even more filtered training set). In contrast, for the *PrimaryTumor15* domain we have a significant increase relative to the original data set when R100 training sets were tested, but for R75, R50, and R25 we have a constant decrease in prediction accuracy; the prediction accuracy for the R25 sets is significantly worse than the accuracy measured for the original data. The *Hungarian* and *Thyroid* domains are similar to the *Cleveland* domain with the optimum for R25 sets, while for the *Echocardiogram* and *Lymphography4* domains the optimum is achieved for R50 sets.

C4.5 results

In order to better verify the obtained gain in prediction accuracy due to noise elimination, a series of tests on both original and pruned training sets were performed. With the intention to simulate a real-life situation, we did not select the reduced sets so that different, optimal ε_h parameter values are selected for different domains. In the experiment with C4.5 algorithm (Quinlan, 1993), we used FiltExR25 reduced sets obtained by 25% of default ε_h parameter values for all eight domains. Table 2 presents average cross-validation prediction accuracies achieved. OrigEx denotes the original datasets while UnP denotes unpruned trees, and P denotes trees pruned by the C4.5 noise-handling mechanism.

The analysis of results in Table 2 shows that noise elimination improves the average accuracy (FiltEx UnP: 72.8%), when compared to using C4.5 on the original dataset (OrigEx UnP: 70.3%). The comparison of the per-

TABLE 2 C4.5 Results

	OrigEx UnP	FiltExR25 UnP	OrigEx P	FiltExR25 P
Breast	68.9	74.1	75.5	75.1
Cleveland	77.0	79.0	84.9	84.9
Echocardiogram	57.9	59.5	70.2	70.2
Hepatitis	74.2	78.8	73.0	73.6
Hungarian	76.6	78.0	80.0	79.7
Lymphography4	71.6	74.8	79.0	78.3
Thyroid	96.0	96.5	96.3	96.8
PrimaryTumor15	40.3	41.8	36.4	38.2
Mean	70.3	72.8	74.4	74.6

formance of noise elimination on individual domains reveals that in all eight domains noise elimination helps the classifier to achieve a better accuracy, and that the accuracy increase is significant ($p < 0.1$ using two-tailed t-test for dependent samples) in seven domains. This result supports our hypothesis that noise elimination by the multiclass saturation filter improves the accuracy of hypotheses.

When compared to the C4.5 pruning mechanism for noise-handling, average results achieved by pruning of C4.5 trees (OrigEx P: 74.4%) are significantly better than the results achieved by noise elimination in preprocessing for C4.5 used without pruning (FiltEx UnP: 72.8%). Using the multiclass saturation filter in conjunction with the C4.5 pruning mechanism results in a further slight accuracy improvement (OrigEx P: 74.4%, FiltEx P: 74.6%). It must be also noted that the C4.5 pruning mechanism resulted in significant prediction accuracy increase for six out of eight domains, but that the results for two domains are worse than the results obtained without pruning.

k-NN Results

The following tests were performed by the k-NN algorithm (Wettschereck, 1994), which selects the optimal k based on the analysis of the dataset. The intention was to show that noise elimination in preprocessing can be useful also for completely different machine-learning paradigms. Using k-NN versus 1-NN can be viewed as noise-handling, due to the influence on the bias/variance trade-off (Wettschereck, 1994) and personal communication). A larger k indicates a stronger bias; hence noise will have less effect. A decrease of the optimal value of k, selected by the k-NN algorithm, indicates a lower noise level in a dataset, which is supported by the theory about the bias/variance trade-off. Results of the experiment by the k-NN algorithm are presented in Table 3. The results were achieved with 1-NN and k-NN without feature weights. When applying feature weights, similar results are achieved. The difference to the previous experiment with

TABLE 3 k-NN Results

	OrigEx 1-NN	FiltExR100 1-NN	OrigEx k-NN	FiltExR100 k-NN
Breast	64.4	65.0	72.7 (25.0)	71.0 (12.6)
Cleveland	75.9	76.6	76.6 (37.4)	77.9 (13.0)
Echocardiogram	58.1	62.7	63.5 (20.8)	68.8 (11.0)
Hepatitis	78.7	79.3	79.4 (13.4)	80.7 (12.6)
Hungarian	74.0	74.0	78.7 (21.0)	80.1 (19.4)
Lymphography4	81.0	82.3	82.3 (4.2)	81.0 (4.6)
Thyroid	95.8	96.1	96.4 (8.6)	96.1 (5.4)
PrimaryTumor 15	40.6	40.9	43.6 (31.0)	42.4 (20.6)
Mean	71.1	72.1	74.2 (20.2)	74.8 (12.4)

C4.5 is that for all domains FiltExR100 reduced sets with default ε_h parameter values are used and that besides the prediction accuracy, the mean values of the used k values are reported in *k*-NN columns.

The analysis of results shows that noise elimination slightly improves the average accuracy (FiltEx 1-NN: 72.1%), when compared to using 1-NN on the original dataset (OrigEx 1-NN; 71.1%). The improvement occurs in seven out of eight datasets. The results are moderately in favor of our hypothesis that noise elimination by the multiclass saturation filter improves the accuracy of induced hypotheses.

k-NN with the automatic detection of the optimal value of *k* (which performs noise handling by extending the neighborhood to *k* nearest neighbors) performs significantly better (OrigEx *k*-NN: 74.2%) than 1-NN on the pruned datasets (FiltEx 1-NN: 72.1%). A further slight improvement of *k*-NN may be achieved by preprocessing the dataset using the multiclass saturation filter (FiltEx *k*-NN: 74.8%).

More importantly, the average optimal value of *k*, when *k*-NN is applied to the datasets after noise reduction, significantly decrease when compared to the situation when *k*-NN is applied to the original datasets. The decrease is present in seven domains, in five of them it is significant. It must be noted that the effect is detected on the least reduced training sets (FiltExR100) and it can be expected to be even more favorable on other, more reduced training sets.

EXPERIMENTAL EVALUATION ON THE PROBLEM OF EARLY DIAGNOSIS OF RHEUMATIC DISEASES

Problem Description

Experiments in this section are aimed at evaluating the performance of the multiclass saturation filter on the problem of early diagnosis of rheumatic diseases. This problem was addressed in our earlier experiments with rule-induction algorithms (Džeroski & Lavrač, 1996; Lavrač et al., 1993; Lavrač & Džeroski, 1994, Lavrač et al., 1997). This is an eight-class diagnostic problem. Table 4 gives the names of the diagnostic classes and the numbers of patients belonging to each class.

To facilitate the comparison with earlier experiments in this domain, the experiments were performed on anamnestic data, without taking into account data about patients' clinical manifestations, laboratory and radiological findings. There are 16 anamnestic attributes: sex, age, family anamnesis, duration of present symptoms (in weeks), duration of rheumatic diseases (in weeks), joint pain (arthrotic, arthritic), number of painful joints, number of swollen joints, spinal pain (spondylotic spondylitic), other pain (headache, pain in muscles, thorax, abdomen, heels), duration of morning

TABLE 4 Class

	Name	# of patients
A1	degenerative spine diseases	158
A2	degenerative joint diseases	128
B1	inflammatory spine diseases	16
B234	other inflammatory diseases	29
C	extraarticular rheumatism	21
D	crystal-induced synovitis	24
E	nonspecific rheumatic manifestations	32
F	nonrheumatic diseases	54

stiffness (in hours), skin manifestations, mucosal manifestations, eye manifestations, other manifestations, and therapy. Some of these attributes are continuous (e.g., age, duration of morning, stiffness) and some are discrete (e.g., joint pain, which can be arthrotic, arthritic, or not present at all).

CN2 and the Relative Information Score Measure

In these experiments the CN2 rule-learning algorithm (Clark & Niblett, 1989; Clark & Boswell, 1991) was used. The most recent version of CN2 (Džeroski et al., 1993) can measure its classification performance in terms of the classification accuracy as well as in terms of the *relative information score* (Kononenko Bratko, 1991). The relative information score is a performance measure which is not biased by the prior class distribution. It accounts for the possibility to achieve high accuracy easily in domains with a very likely majority class by taking into account the prior probability distribution of the training examples.

Let the correct class of example e_k be c_k, its prior probability $p_k = p(c_k)$, and the probability returned by the classifier $p'_k = p'(c_k)$. The *information score* of this answer is

$$I(e_k) = \begin{cases} -\log p_k + \log p'_k & p'_k \geq p_k \\ \log(1 - p_k) - \log(1 - p'_k) & p'_k < p_k \end{cases}$$

As $I(e_k)$ indicates the amount of information about the correct classification of e_k gained by the classifier's answer, it is positive if $p'_k > p_k$, negative if the answer is misleading ($p'_k < p_k$) and zero if $p'_k = p_k$.

The *relative information score* I_r of the answers of a classifier on a testing set consisting of examples $e_1, e_2, \ldots, e_t$ each belonging to one of the classes $c_1, c_2, \ldots, c_N$ can be calculated as the ratio of the average information score of the answers and the entropy of the prior distribution of classes:

$$I_r = \frac{\frac{1}{t} \times \sum_{k=1}^{t} I(e_k)}{-\sum_{i=1}^{N} p_i \log p_i}$$

Experimental Setting

Experiments were performed on the same training and testing sets as used in our previous experiments (Lavrač et al., 1993; Lavrač & Džeroski, 1994) using 10 different random partitions of the data into 70% training and 30% testing examples. In this way, 10 training sets E_i and 10 testing sets T_i, $i = 1 \ldots 10$ were generated. In these experiments, CN2 (Džeroski et al., 1993) was applied with and without its significance test noise-handling mechanism. When using the significance test in CN2, a significance level of 99% was applied. For an eight-class problem, this corresponds to a threshold 18.5 for the value of the likelihood ratio statistic. The other CN2 parameters were set to their default values (Clark & Boswell, 1991). When using the multiclass saturation filter for noise elimination, the noise sensitivity parameter ε_h had its default values 1.5 for training sets with 2–50 examples (training sets for diagnosis *B*1, *B*234, *C*, *D*, *E*), 1.0 for 51–100 examples (*F*), 0.75 for 101–200 examples (*A*1, *A*2), and 0.5 for more than 200 examples (there was no such training set).

CN2 Results

Previous experiments (Lavrač et al., 1993; Lavrač & Džeroski, 1994) show that the CN2 noise-handling mechanism improves the classification accuracy, but decreases the relative information score. These results are reproduced in Table 5, columns OrigEx CN2-ST and OrigEx CN2-NoST (ST denotes the use of significance test in CN2, and NoST means that no significance test was used).

In our experiments, we tested the performance of the noise elimination

TABLE 5 CN2 Results

	Accuracy			Relative information score		
Partition	OrigEx CN2-ST	OrigEx CN2-NoST	FiltEx CN2-NoST	OrigEx CN2-ST	OrigEx CN2-NoST	FiltEx CN2-NoST
1	47.5	38.1	45.3	17.0	21.0	26.0
2	45.3	44.6	44.6	20.0	23.0	28.0
3	51.1	45.3	47.5	17.0	19.0	24.0
4	44.6	43.9	38.8	17.0	24.0	20.0
5	46.0	40.3	41.7	21.0	22.0	25.0
6	49.6	48.2	50.4	15.0	26.0	24.0
7	44.6	42.4	46.8	21.0	27.0	31.0
8	41.0	38.8	43.2	21.0	19.0	25.0
9	43.9	45.3	48.2	16.0	23.0	29.0
10	39.6	41.7	43.2	23.0	23.0	25.0
Mean	45.3	42.9	45.0	18.8	22.7	25.7

algorithm by comparing the results achieved by CN2 before and after noise elimination. These results are given in columns CN2–NoST of Table 5.

In order to observe the effect of noise elimination, the results in columns OrigEx CN2–NoST and FiltEx CN2–NoST of Table 5 need to be compared. On the other hand, in order to compare the noise elimination algorithm with the CN2 noise–handling mechanism using the significance test, compare the columns OrigEx CN2–ST, and FiltEx CN2–NoST in Table 5. This is actually the most interesting comparison since, in terms of classification accuracy, CN2 with significance test CN2–ST) is known to perform well on noisy data.

The elimination of noisy examples increases the classification accuracy from 42.9% to 45.0%. This increase is statistically significant at the 96% level according to the one–tailed paired t–test. This result is in favor of our expectation that the elimination of noisy examples is useful for concept learning. The effect of noise–handling by the noise elimination algorithm (accuracy 45.0%) is comparable to the effect of the significance test (accuracy 45.3% achieved by CN2–ST); the difference in performance is not significant.

In terms of the relative information score, substantial improvements are achieved by applying the noise elimination algorithm. The relative information score significantly increases (from 22.7% to 25.7%) after the elimination of noisy examples. Particularly favorable is the comparison between the noise elimination and the significance test used as the CN2 noise–handling mechanism: there is an almost 7% difference in favor of noise elimination, i.e., an increase from 18.8% to 25.7%.

SUMMARY

The reported results indicate the utility of the multiclass saturation filtering for finding noisy examples in the dataset. Similar as in Brodley & Friedl (1996), the approach may introduce an important new paradigm for learning: noise elimination in preprocessing. The set of examples, detected as potentially noisy by the saturation filter, may also include outliers. The proposed paradigm suggests that the potentially noisy examples should be shown and analyzed by an expert; examples representing real noise should be eliminated from the training set, whereas outliers should be, after hypothesis construction, added as exceptions to the generated rule.

The results of experiment show that the prediction accuracy of machine–learning algorithms increases when combined with our noise detection and elimination approach. Despite the fact that the noise–handling mechanisms of C4.5, *k*–NN, and CN2 outperform the use of the saturation filter in preprocessing, the comparative advantage of our approach is the explicit detection of errors and outliers in the training set. The most interesting results are

those achieved by CN2 on the problem of early diagnosis of rheumatic diseases, which show the adequacy of the elimination of noisy examples as a noise-handling mechanism. On the reduced datasets, obtained by applying the multiclass saturation filter, the CN2 learning algorithm without its noise-handling mechanism (no significance test) yielded accuracies comparable to those of CN2 with its noise-handling mechanism (with significance test) on the original datasets. More importantly, noise elimination resulted in significantly better relative information scores, thus improving the overall performance. These relative information scores are the best scores achieved with CN2 in this domain (Lavrač et al., 1993; Lavrač & Džeroski, 1994).

NOTES

1. In the standard machine-learning terminology we may reformulate the definition of coverage of p/n pairs as follows: literal l covers a (e_i, e_j) pair if l covers the positive example e_i and does not cover the negative example e_j.
2. Alternatively, an exhaustive minimal-covering algorithm could be used (Gamberger, 1995; Gamberger & Lovrač, 1996) but its time complexity prevents its use in real-life applications.

REFERENCES

Brodley, C. E., and M. A. Friedl. 1996. Identifying and eliminating mislabeled training instances. In *Proceedings of the 13th National Conference on Artificial Intelligence*, 799–805. Menlo Park: AAAI Press.

Cestnik, B., and I. Bratko. 1991. On estimating probabilities in tree pruning. In *Proceedings of the Fifth European Working Session on Learning*, 138–150. New York: Springer-Verlag.

Cestnik, B., I. Kononenko, and I. Bratko. 1987. ASSISTANT 86: A knowledge elicitation tool for sophisticated users. In I. Bratko and N. Lavrač, eds., *Progress in Machine Learning*. 31–45. Wilmslow, UK: Sigma Press.

Clark, P., and T. Niblett. 1989. The CN2 induction algorithm. *Machine Learning* 3(4):261–283.

Clark, P., and R. Boswell. 1991. Rule induction with CN2: Some recent improvements. In *Proceedings of the Fifth European Working Session on Learning*, 151–163. New York: Springer-Verlag.

Džeroski, S., B. Cestnik, and I. Petrovski. 1993. Using the m-estimate in rule induction. *Journal of Computing and Information Technology* 1:37–46.

Džeroski, S., and N. Lavrač. 1996. Rule induction and instance-based learning applied in medical diagnosis. *Technology and Health Care* 4(2):203–221.

Fayyad, U. M., and K. B. Irani. 1992. On the handling of continuous-valued attributes in decision tree generation. *Machine Learning* 8:87–102.

Gamberger, D. 1995. A minimization approach to propositional inductive learning. In *Proceedings of the 8th European Conference on Machine Learning*, 151–160. Berlin, Springer.

Gamberger, D., and N, Lavrač. 1996. Noise detection and elimination applied to noise handling in a KRK chess endgame. In *Proceedings of the 5th International Workshop on Inductive Logic Programming*, 59–75. Berlin, Springer

Gamberger, D., and N. Lavrač. 1997. Conditions for Occam's razor applicability and noise elimination. In *Proceedings of the 9th European Conference on Machine Learning*, 108–123. Berlin, Springer.

Gamberger, D., N. Lavrač, and S. Džeroski. 1996. Noise elimination in inductive concept learning: A case study in medical diagnosis. In *Proceedings of the 7th International Workshop on Algorithmic Learning Theory*, 199–212. Berlin, Springer.

John, G. H. 1995. Robust decision trees: Removing outliers from data. In *Proceedings First Int. Conference on Knowledge Discovery and Data Mining*, 174–179. Menlo Park: AAAI Press.

Kononenko, I., and I. Bratko. 1991. Information based evaluation criterion for classifier's performance. *Machine Learning* 6(1):67–80.

Lavrač, N., and S. Džeroski. 1994. *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood.

Lavrač, N., S. Džeroski, V. Pirnat, and V. Križman. 1993. The utility of background knowledge in learning medical diagnostic rules. *Applied Artificial Intelligence* 7:273–293.

Lavrač, N., D. Gamberger, and S. Džeroski. 1995. An approach to dimensionality reduction in learning from deductive databases. In *Proceedings of the 5th International Workshop on Inductive Logic Programming*, 337–354.

Lavrač, N., D. Gamberger, and S. Džeroski. 1997. Noise elimination applied to early diagnosis of rheumatic diseases. In N. Lavrač, E. Keravnou, and B. Zupan, eds. *Intelligent Data Analysis in Medicine and Pharmacology*, 187–205. Boston: Kluwer.

Mingers, J. 1989. An empirical comparison of selection measures for decision-tree induction. *Machine Learning* 3(4):319–342.

Mingers, J. 1989. An empirical comparison of pruning methods for decision tree induction. *Machine Learning* 4(2):227–243.

Muggleton, S. H., A. Srinivasan, and M. Bain. 1992. Compression, significance and accuracy. In *Proceedings of the 9th International Conference on Machine Learning*, 338–347. San Mateo, CA: Morgan Kaufmann.

Murphy, P. M., and D. W. Aha. 1994. *UCI Repository of machine learning databases* [http://www.ics.uci.edu/mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

Quinlan, J. R. 1987. Simplifying decision trees. *International Journal of Man-Machine Studies* 27(3):221–234.

Quinlan, J. R. 1993. *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Quinlan, J. R. 1996. Boosting first-order learning. In *Proceedings of the 7th International Workshop on Algorithmic Learning Theory*, 143–155. New York: Springer-Verlag.

Rissanen, J. 1978. Modeling by shortest data description. *Automatica* 14:465–471.

Srinivasan, A., S. Muggleton, and M. Bain. 1992. Distinguishing exceptions from noise in non-monotonic learning. In *Proceeding 2nd International Workshop on Inductive Logic Programming*. Tokyo, Japan.

Wettschereck, D. 1994. *A study of distance-based machine learning algorithms*. Ph.D. Thesis, Department of Computer Science, Oregon State University, Corvallis, OR.