# Noise Elimination in Inductive Concept Learning: A case study in medical diagnosis

Dragan Gamberger[1], Nada Lavrač[2] and Sašo Džeroski[2]

[1] Rudjer Bošković Institute, Bijenička 54
10000 Zagreb, Croatia
tel. +385 1 4561142, fax +385 1 425497
[2] Jožef Stefan Institute, Jamova 39
1000 Ljubljana, Slovenia
tel. +386 61 1773272, fax +386 61 219385
gambi@lelhp1.irb.hr, nada.lavrac@ijs.si, saso.dzeroski@ijs.si

**Abstract.** Compression measures used in inductive learners, such as measures based on the MDL (Minimum Description Length) principle, provide a theoretically justified basis for grading candidate hypotheses. Compression-based induction is appropriate also for handling of noisy data. This paper shows that a simple compression measure can be used to detect noisy examples. 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. The separation of noise detection and hypothesis formation has the advantage that noisy examples do not influence hypothesis construction as opposed to most standard approaches to noise handling in which the learner typically tries to avoid overfitting the noisy example set. This noise elimination method is applied to a problem of early diagnosis of rheumatic diseases which is known to be a difficult problem, due both to its nature and to the imperfections in the dataset. The method is evaluated by applying the noise elimination algorithm in conjunction with the CN2 rule induction algorithm, and by comparing their performance to earlier results obtained by CN2 in this diagnostic domain.

## 1 Introduction

Machine learning systems [15, 17] have been applied to numerous practical problems. From given training examples, a machine learning system can construct explicit symbolic rules that generalize the training cases [15, 2]. The rules, induced from the set of training examples, can be used for classification (diagnosis/prediction) of new cases.

In an ideal inductive learning problem, the induced rules (hypothesis $H$) will 'agree' with the classifications of the descriptions of all the training examples $E$. 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 (imperfect) 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., [19]) and in stopping criteria (e.g., [2]) 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 post-processing, such as postpruning and simplifying of decision trees (e.g., [1, 18, 23]). Systems employing such techniques are called *noise-tolerant* systems since they try to avoid overfitting the possibly noisy training set.

The problem of noise handling has been addressed also in inductive logic programming (ILP) [20, 12] by adapting successful approaches to noise handling from attribute-value learning. Recently, compression measures [21, 24] that are theoretically based on the Minimum Description Length (MDL) principle [25] have gained much attention. These measures provide a theoretically justified basis for grading candidate hypotheses, integrating a measure of complexity (simplicity or understandability) and correctness (expected accuracy) into a single heuristic measure for hypothesis evaluation. Compression based induction is appropriate also for handling of noisy data [21].

This paper takes a different approach to dealing with noise. A technique is proposed that detects and eliminates noisy examples from the training set by using a simple compression measure. A consistent and complete hypothesis can be then built from the set of remaining examples by using a learning algorithm which is not noise-tolerant. The separation of noise detection and hypothesis formation has the advantage that noisy examples do not influence hypothesis construction. This is different from the approaches that handle noise in the hypothesis formation process by trying to avoid overfitting the noisy example set.

In our approach to noise detection and elimination, we assume an inductive learning setting consisting of three separate steps: preprocessing of the training set (described in Section 2), noise detection and elimination (Section 3), and hypothesis formation. Noise detection and elimination is the main part of this paper, whereas hypothesis formation is done by using an implementation of the well-known CN2 algorithm [3]. Section 4 presents the experimental domain (early diagnosis of rheumatic diseases), the experimental setting and the results achieved by applying the proposed noise-handling method in this diagnostic problem. The paper concludes with a discussion and gives directions for further research.

## 2 Preprocessing of training examples

The basic algorithm for detecting noisy examples, described in detail in Section 3, works for two-class learning problems where positive and negative examples of a single concept are described by literals (binary features). Therefore, for

each two-class problem, the training examples from $E$, described as tuples of attribute values, need to be mapped to tuples of truthvalues of literals.

The literals are constructed based on the analysis of values of positive and negative examples in the training set $E$. For each attribute $A_i$, let $v_{ij}$ ($j = 1..k_{ip}$) be $k_{ip}$ different values of the attribute that appear in positive examples in $E$ and let $w_{ij}$ ($j \in 1..k_{in}$) be the $k_{in}$ different values of the attribute that appear in negative examples in $E$.

- If $A_i$ is a discrete attribute, $k_{ip}$ different literals of the form $A_i = v_{ij}$ and $k_{in}$ different literals of the form $A_i \neq w_{ij}$ are created. In the case that a positive and a negative example have the same value $v_{ix} = w_{iy}$ of $A_i$ then two literals are created: $A_i = v_{ix}$ and $A_i \neq w_{iy}$.
  For illustration, consider a problem with two attributes, $A$ and $B$, and a training set of three examples, two positive examples $(a_2, b_1)$ and $(a_3, b_2)$ and a negative example $(a_1, b_2)$. Then the following literals are created: $A \neq a_1$, $A = a_2$, $A = a_3$, $B = b_1$, $B = b_2$, $B \neq b_2$, and the three examples correspond to the following truthvalue tuples of literals: 110101, 101010 and 000010, respectively (where 1 stands for $true$ and 0 stands for $false$). Literals $A = a_1$, $A \neq a_2$, ... are not even considered since they are either $false$ for all positive examples ($A = a_1$) or $true$ for all negative examples ($A \neq a_2$); as such they are useless for constructing a concept description.
- If $A_i$ is a continuous attribute, we first search for all value pairs $(v_{ix}, w_{iy})$, where $x = 1..k_{ip}$ and $y = 1..k_{in}$, that satisfy the property $v_{ix} < w_{iy}$ and for which the 'neighborhood' property holds (i.e., no value $v_{iz}$ or $w_{iz}$ of attribute $A_i$ exists such that $v_{ix} < v_{iz} < w_{iy}$ or $v_{ix} < w_{iz} < w_{iy}$). For such value pairs, literals of the form $A_i \leq (v_{ix} + w_{iy})/2$ are created. In a similar way, for all neighboring pairs $(w_{iy}, v_{ix})$, where $x = 1..k_{ip}$ and $y = 1..k_{in}$ (i.e., pairs such that $w_{iy} < v_{ix}$ and such that no other value $v_{iz}$ or $w_{iz}$ exists between values $v_{ix}$ and $w_{iy}$ such that $w_{iy} < v_{iz} < v_{ix}$ or $w_{iy} < w_{iz} < v_{ix}$ holds) literals of the form $A_i > (v_{ix} + w_{iy})/2$ are created. The motivation is similar to one suggested in [5].
- If $A_i$ is an attribute of type integer, then literals are generated by previously described procedures as if $A_i$ were both discrete and continuous. Depending on the training set, this may result in literals of four different forms: $A_i \leq (v_{ix} + w_{iy})/2$, $A_i > (v_{ix} + w_{iy})/2$, $A_i = v_{ij}$, and $A_i \neq w_{ij}$. In an extreme case, all four literals may be generated.

In our approach, we use an implicit definition of equality for continuous attributes which allows for all values (also non-integer) in some range. This has the following advantages. First, by allowing only literals of the form $A_i \leq (v_{ix} + w_{iy})/2$ and $A_i > (v_{ix} + w_{iy})/2$ we avoid the precision problem that would have occurred if literals of the form $A_i = v_{ij}$, say $A_i = 3.143$, were allowed for continuous attributes. Namely, if the system were to determine the truthvalue of the literal $A_i = 3.143$ for an example which has value 3.1432 of attribute $A_i$, should the literal be evaluated as $true$ or $false$? This decision depends on the precision applied in the comparison, but this precision is not defined. Moreover, the fact that continuous attributes do not have literals of the form $A_i = 3.143$

does not prevent testing whether $A_i$ has value 3.143 in the generated rule; a learner can namely build a condition as a conjunction of elementary literals, such as for instance $(A_i > 3.142) \wedge (A_i \leq 3.144)$. The advantage of constructing conditions in this form is that they have an explicitly defined precision.

Obviously, the implicit definition of equality introduced for continuous attributes does not make sense for integer valued attributes since in this case the precision problem does not exist. This is the reason for introducing a separate type for attributes with integer values.

## 3 Noise detection and elimination

### 3.1 Literals and $p/n$ pairs

In this section, we consider a two-class learning problem where the training set $E$ consists of tuples of truthvalues of literals. For noise elimination, we will investigate the properties of literals that hold on individual pairs of training examples, each pair consisting of a positive and a negative example.

**Definition 1.** Let $E = P \cup N$, where $P$ are positive and $N$ are negative examples. A $p/n$ pair is a pair of two examples $e_i$ and $e_j$ such that $e_i \in P$ and $e_j \in N$. When appropriate, we will use the notation $p_i/n_j$ for a $p/n$ pair consisting of $p_i \in P$ and $n_j \in N$.

**Definition 2.** Let $L$ denote a set of literals. A literal $l \in L$ covers a $p_i/n_j$ pair if the literal has value *true* for $p_i$ and value $false$ for $n_j$.[3]

The notion of $p/n$ pairs can be used to prove important properties of literals for building complete and consistent concept descriptions [6, 7].

**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 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 can not 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

[3] In the standard machine learning terminology we may reformulate the definition of coverage of $p/n$ pairs as follows: literal $l$ covers a $p_i/n_j$ pair if $l$ covers the positive example $p_i$ and does not cover the negative example $n_j$.

presented, based on $k$ runs of a covering algorithm, where $k$ is the cardinality of the set of positive examples ($k = |P|$). In the $i$-th 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: $H = h_1 \vee \ldots \vee h_k$. □

As will be shown in Section 3.3, this theorem plays a central role in noise elimination. 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. One possible measure of complexity of a hypothesis $H$ is the number of different literals that appear in it. 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. $E$ is denoted by $q(E, L)$. In fact, $q(E, L) = |L'|$, where $L'$ is the smallest subset of $L$ that allows the construction of a hypothesis $H$ consistent and complete w.r.t. $E$. As Theorem 1 suggests, one can compute $q(E, L)$ without actually constructing the hypothesis $H$. This idea plays a central role in noise elimination.

### 3.2 Background

Let us assume that for the given training set $E$ and the given set of literals $L$ a consistent and complete hypothesis $H$ (an approximation to an unknown target theory $T$) can be found.[4] Let $q(E, L)$ represent the complexity of the simplest hypothesis complete and consistent with training examples in $E$ and built of literals from $L$. Let us initially fix the set of literals $L$ and study $q$ only as function of $E$, i.e. $q(E, L) = q(E)$.

Given the sets $E$ and $L$, assume that the training set $E$ contains enough training examples to induce/identify a correct hypothesis $H$.[5] If the set of examples is large enough to identify the correct hypothesis $H$ then, by adding examples to $E$, the complexity of the simplest hypothesis that can be generated from the enlarged training set will not increase. Let $m = |E|$. This means that for training examples $e_{m+1}, e_{m+2}, \ldots$ that were not included in the initial training set $E$ it holds that:

$$q(E) = q(E \cup \{e_{m+1}\}) = q(E \cup \{e_{m+1}, e_{m+2}\}) = \ldots$$

But if we add to $E$ an example that is inconsistent with the target domain theory, i.e., an example $f$ that is noisy, then the complexity of the hypothesis will increase:

$$q(E \cup \{f\}) > q(E).$$

[4] This means that there are no contradictions in the training set, i.e., examples with same truth values of literals belonging to two different classes, since, as a consequence of Theorem 1, no complete and consistent hypothesis $H$ can be built from a training set that contains contradictions.

[5] Correctness of $H$ means that $H$ is complete and consistent for all, including unseen, examples of the unknown target theory $T$. In other words, $H$ denotes the same concept as $T$.

**Theorem 2** *Suppose that the set of examples $E$ is non-noisy and is large enough to enable a unique simplest correct hypothesis $H$ to be generated. By adding an example to the training set, a new training set is generated $E' = E \cup \{e\}$. If $e$ is consistent with the unknown target theory $T$ ($e$ is non-noisy), then $q(E') = q(E)$. If $e$ is inconsistent with $T$ ($e$ is noisy), then $q(E') > q(E)$.*

**Proof:**
For a 'large enough' training set $E$ consisting of all correct training examples it must be true that $H$ is the simplest hypothesis that is complete and consistent with $E$. If this is not true then either set $E$ is not large enough or $H$ is not the simplest hypothesis, both being in contradiction with the assumptions of the theorem. Let $q_H$ denote the complexity of $H$. Then for every $E$ consisting of non-noisy examples: $q(E) \leq q_H$. For the large enough non-noisy $E$: $q(E) = q_H$. If $e$ is non-noisy, then $E'$ is a non-noisy set as well and $q(E') = q_H = q(E)$. This proves the first part of the theorem.

The condition of unique induction of the correct hypothesis $H$ from large enough $E$ assumed by the theorem means that for $E$ no other complete and consistent hypothesis of complexity $q_H$ exists. If $e$ is a noisy example then $H$ is not a complete and consistent hypothesis for $E'$. A different hypothesis must be constructed and for this, according to the condition of uniqueness of the hypothesis $H$, $q(E') > q_H = q(E)$ must hold. Note that it is supposed that for $E'$ a complete and consistent hypothesis can be constructed. If this is not true it means that $e$ is in contradiction with some training example from $E$. In this case the presence of noise can be detected by the existence of contradictions in the training set. □

Theorem 2 presents the basis for the suggested noise elimination algorithm. But it must be noted that the theorem makes use of the definition of a large enough training set $E$. In practice it is very hard to verify whether a training set $E$ is large enough. In practice, there need not exist a unique simplest correct hypothesis $H$. The consequence is that the claims of the theorem need not necessarily hold, which results in imperfect noise detection. The assumption is that with the increase of the number of non-noisy examples in $E$, the conditions of the theorem will be better satisfied thus increasing the probability of successful noise detection. The theorem correctness does not depend on the definition of the hypothesis complexity function $q(E)$, but the sensitivity of this function can influence the value $m$, the number of training examples necessary for the applicability of the theorem.

### 3.3 The noise elimination algorithm for two-class learning

Now we can try to answer the following question: Does the training set $E$ contain noise? Based on Theorem 2 we can answer the question in the following way. If it holds that

$$q(E) = q(E \setminus \{e\})$$

for all possible examples $e \in E$ then there is no noise in the domain and the domain is large enough for a correct hypothesis $H$ to be induced. If this is not the

case and an example $e$ enables complexity reduction, i.e., $q(E) > q(E \setminus \{e\})$, then the example $e$ is potentially incorrect. If there is more than one example whose elimination can lead to complexity reduction, it is advisable to eliminate only the one that reduces the complexity the most. The elimination of 'suspicious' examples is an iterative process where one example is eliminated in each iteration until no elimination can result in further complexity reduction.

Theorem 1 can be used as a basis for the implementation of an efficient algorithm for noise elimination. Let the heuristic estimate of the complexity of the hypothesis $q(E, L)$ be defined as the minimal number of literals $|L'|$ needed to build a complete and consistent hypothesis $H$. The minimal set $L'$ can be selected from $L$ by either the heuristic or the exhaustive literal minimization algorithm of [6, 13]. The basic idea of the heuristic noise elimination algorithm is to use the minimal subset $L'$, $L' \subseteq L$, and to answer the following question: 'Can any of the literals in $L'$ be made unnecessary for building a complete and consistent hypothesis if one (or a few) training example(s) are eliminated?' The practical applicability of the proposed approach follows from the observation that, in most cases, for a noisy example $e_i$ a set $L'_i$ can be found such that $L'_i \subset L'$, where $L'_i$ covers all $p/n$ pairs of the set $E \setminus \{e_i\}$. The noisy examples that do not have this property will not be detected by the noise elimination algorithm. The heuristic noise elimination algorithm (Algorithm 1), which employs the heuristic literal minimization algorithm (Algorithm 2) is presented below.

**Algorithm 1: Heuristic noise elimination**

**Given:** set of literals $L$, quality function $q(E, L) = |L'|$ (i.e. the number of literals in the minimal set $L'$), noise sensitivity parameter $\varepsilon_h$
**Input:** training set $E$ with $m$ examples
**Output:** pruned example set $E$

**repeat**
  find by heuristic search a minimal subset $L'$ of $L$ such that a complete and consistent hypothesis $H(E, L')$ can be built (Algorithm 2)
  **for** all examples $e_i \in E$ $(i = 1 \ldots m)$ set weight $w(e_i) = 0$
  **repeat** for all literals $l \in L'$
    find the minimal subset $E'$ of examples by whose elimination the literal $l$ can be made unnecessary (Procedure 1)
    for all examples $e_i \in E'$ compute $w(e_i) = w(e_i) + 1/|E'|$
  **endrepeat**
  select the example $e_i$ with greatest weight $w(e_i)$
  **if** $w(e_i) > \varepsilon_h$ **then** eliminate $e_i$ from $E$ and begin a new iteration with $E = E \setminus \{e_i\}$ and $m = m - 1$
  **else** stop example elimination
**endrepeat**

The literal minimization algorithm (Algorithm 2) employing heuristic search is outlined below.

**Algorithm 2: Heuristic minimization of literals**

**Input:** set of literals $L$, set of all $p/n$ pairs $PN$
**Output:** minimal set of literals $L'$
initialize the set of $p/n$ pairs not covered by any literal $PN' := PN$
initialize the minimal set of literals $L' := \emptyset$
for each $p/n \in PN'$ compute the weight $v(p/n) = 1/z$ where $z$ is the number of literals that satisfy the pair
**while** not covered $p/n$ pairs exist **do**
select an uncovered $p/n$ pair from $PN'$ (called $p_s/n_s$) that can be covered by the least number of literals (i.e., $p/n$ with the maximal value of $v(p/n)$)
for each $l \in L$ covering $p_s/n_s$ compute $w(l) = \sum v(p/n)$ where summation is over all uncovered $p/n$ pairs from $PN'$ covered by $l$
select the literal $l_s$ with the maximal $w(l)$ value and include it in the set of selected literals $L' := L' \cup \{l_s\}$
remove from the set of uncovered $p/n$ pairs $PN'$ all $p/n$ pairs covered by the selected literal $l_s$
**endwhile**

The algorithm results in a list of selected literals $L'$ which is the heuristically selected minimal literal set.

Algorithm 1 calls another procedure which, for the given subset $L'$ and some literal $l \in L'$, determines the minimal subset of examples that must be eliminated in order to make $l$ unnecessary when building a complete and consistent hypothesis.

**Procedure 1: Finding the mininal example set for elimination**

**Input:** a set of literals $L'$ such that a complete and consistent hypothesis $H(E, L')$ can be generated, a literal $l \in L'$
**Output:** the minimal set of examples $E'$ that should be eliminated in order to make $l$ unnecessary
find all $p_i/n_j$ pairs that are covered by $l$ and not covered by any other literal from $L'$
form set $P'$ consisting of all $p_i$ examples
form set $N'$ consisting of all $n_j$ examples
**if** $|P'| < |N'|$ **then** $E' = P'$ is the minimal example set
**else** $E' = N'$ is the minimal example set

### 3.4 The noise elimination algorithm for multi-class learning

The basic algorithm for detecting noisy examples described in Section 3.3, works for two-class learning problems where positive and negative examples of a single concept are described by literals (binary features). This section describes how to perform noise elimination for disjoint multi-class problems.

Given an example set $E$ of a multi-class learning problem, the elimination of noisy examples is performed as follows.

1. For each of the $N$ 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 $(e_i, c_{ij})$, 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 truthvalues of literals (see Section 2 describing this transformation). This results in new example sets $E_j$, $j = 1..N$.
3. For each of the $N$ two-class learning problems, a set of noisy examples $O'_j$ is detected by applying Algorithm 1 (see Section 3.3). Let $O_j = \{e_i \mid f(e_i) \in O'_j\}$.
4. Finally, the noisy examples $O_j$ of each $E_j$ are eliminated from the original multi-class training set $E$. This results in a reduced training set

$$NE = E \setminus \bigcup_{j=1..N} O_j.$$

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

## 4 Experimental evaluation

### 4.1 Early diagnosis of rheumatic diseases

Correct diagnosis in the early stage of rheumatic disease is a difficult problem. Having passed all the investigations, many patients can not be reliably diagnosed after their first visit to the specialist. The reason is that anamnestic, clinical, laboratory and radiological data of patients with different rheumatic diseases are frequently similar. In addition, diagnosis can also be incorrect due to the subjective interpretation of data [22].

Data about 462 patients were collected at the University Medical Center in Ljubljana [22]. There are over 200 different rheumatic diseases which can be grouped into three, six, eight or twelve diagnostic classes. As suggested by a specialist, eight diagnostic classes were considered [8] in this experiment. The names of diagnostic classes and the number of patients per class (in parentheses) is as follows: $A1$ - 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), and $F$ - nonrheumatic diseases (54).

To facilitate the comparison with earlier experiments in rule induction in this domain [11], the experiments were performed on anamnestic data, without taking into account data about patients' clinical manifestations, laboratory

and radiological findings. The sixteen anamnestic attributes are as follows: 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 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).

### 4.2 Experimental setting and previous results

The goal of our experiments is to show the utility of noise elimination in learning from noisy data. Since the ultimate test of the quality of induced rules is their performance on unseen examples, experiments were performed on ten different random partitions of the data into 70% training and 30% testing examples. In this way, ten training sets $E_i$ and ten testing sets $T_i$, $i = 1..10$ were generated. These are the same training and testing sets as used by Lavrač et al. [11, 12], from where the results of CN2 on the original datasets are taken. In these experiments, CN2 [4] 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 (see the Appendix). The other CN2 parameters had default values [3]. Algorithm 1 was used for noise elimination. The noise sensitivity parameter $\varepsilon_h$ had its default values: 1.5 for training sets with 2–50 examples (training sets for diagnoses $B1$, $B234$, $C$, $D$, $E$), 1.0 for 51–100 examples ($F$), 0.75 for 101–200 examples ($A1$, $A2$) and 0.5 for more than 200 examples (no such training set).

Previous experiments [11, 12] show that the CN2 noise-handling mechanism improves the classification accuracy, but decreases the relative information score (see the Appendix for the definition of the relative information score). These results are reproduced in Table 1, columns Original CN2-ST and Original CN2-NoST.

### 4.3 Results of noise elimination

In our experiments, we tested the performance of the noise elimination algorithm (outlined in Sections 3.3 and 3.4) by comparing the results achieved by CN2 before and after noise elimination. Noise elimination resulted in reduced datasets $NE_1$, ..., $NE_{10}$. Rules were induced by CN2 from $NE_i$, while the accuracy and relative information score were measured on $T_i$, $i = 1..10$. These results are given in columns Reduced CN2-NoST of Table 1.

In order to observe the effect of noise elimination, the reader should compare the results in columns Original CN2-NoST and Reduced CN2-NoST of Table 1. On the other hand, in order to compare the noise elimination algorithm to the CN2 noise-handling mechanism using the significance test, the reader should

| | accuracy | | | relative information score | | |
|---|---|---|---|---|---|---|
| Partition | Original CN2-ST | Original CN2-NoST | Reduced CN2-NoST | Original CN2-ST | Original CN2-NoST | Reduced 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 |
| Average | 45.3 | 42.9 | 45.0 | 18.8 | 22.7 | 25.7 |

**Table 1.** Accuracy and relative information score

compare the columns Original CN2-ST and Reduced CN2-NoST in Table 1. 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 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%.

## 5 Discussion and further work

This paper presents a method for noise detection and elimination for inductive learning and its application in the domain of early diagnosis of rheumatic diseases. The latter is indeed a difficult problem, as reflected by the relatively low classification accuracy results achieved by various learning algorithms [1, 22, 8, 11, 12].

The results achieved in this study show the adequacy of the elimination of noisy examples as a noise-handling mechanism. On the reduced datasets, obtained by applying our noise elimination procedure, the CN2 learning algorithm without its noise-handling mechanism (significance test) yielded accuracies comparable to those of CN2 with its noise-handling mechanism (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 [11, 12].

The results presented here show the potential of the noise elimination approach. While elimination of noisy examples is well suited for domains with few errors in the data (results of experiments with controlled amounts of noise added to a noiseless dataset [7]), our experiments indicate that it can also yield good results in a very noisy domain. Further experimental evaluation of the noise elimination approach is planned, in particular on other medical datasets that are usually used to compare the performance of different machine learning algorithms [4].

## Acknowledgements

This research was financially supported by the Slovenian Ministry of Science and Technology, the Croatian Ministry of Science and the ESPRIT Project 20237 Inductive Logic Programming II. The authors are grateful to the specialists of the University Medical Center in Ljubljana who helped collecting the data, especially to Vladimir Pirnat. Aram Karalič and Igor Kononenko prepared the data in a form appropriate for the experiments. Sašo Džeroski was an ERCIM Fellow at the German National Research Center for Information Technology (GMD), Sankt Augustin, Germany at the time of writing this paper.

## References

1. Cestnik, B., and Bratko, I. (1991). On estimating probabilities in tree pruning. In *Proc. 5th European Working Session on Learning*, pages 138–150. Springer, Berlin.
2. Clark, P. and Niblett, T. (1989). The CN2 induction algorithm. *Machine Learning*, 3(4): 261–283.
3. Clark, P. and Boswell, R. (1991). Rule induction with CN2: Some recent improvements. In *Proc. 5th European Working Session on Learning*, pages 151–163. Springer, Berlin.
4. Džeroski, S., Cestnik, B., and Petrovski, I. (1993) Using the $m$-estimate in rule induction. *Journal of Computing and Information Technology*, 1: 37–46.
5. Fayyad, U.M. and Irani, K.B. (1992). On the handling of continuous-valued attributes in decision tree generation. *Machine Learning*, 8: 87–102.
6. Gamberger, D. (1995). A minimization approach to propositional inductive learning. In *Proc. 8th European Conference on Machine Learning*, pages 151–160. Springer, Berlin.

7. Gamberger, D., and Lavrač, N. (1996). Noise elimination in inductive learning. Technical report IJS-DP-7400, J. Stefan Institute, Ljubljana, 1996.
8. Karalič, A., and Pirnat, V. (1990). Machine learning in rheumatology. *Sistemica* 1(2): 113–123.
9. Kononenko, I. and Bratko, I. (1991). Information-based evaluation criterion for classifier's performance. *Machine Learning*, 6(1): 67–80.
10. Kononenko, I., and Kukar, M. (1995). Machine learning for medical diagnosis. In *Proc. Workshop on Computer-Aided Data Analysis in Medicine*, IJS Scientific Publishing, IJS-SP-95-1, Ljubljana.
11. Lavrač, N., Džeroski, S., Pirnat, V., and Križman, V. (1993). The utility of background knowledge in learning medical diagnostic rules. *Applied Artificial Intelligence*, 7: 273–293.
12. Lavrač, N. and Džeroski, S. (1994). *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood, Chichester.
13. Lavrač, N., Gamberger, D., and Džeroski, S. (1995). An Approach to Dimensionality Reduction in Learning from Deductive Databases. In *Proc. 5th International Workshop on Inductive Logic Programming*. Katholieke Universiteit Leuven, 1995.
14. Lavrač, N., Džeroski, S., and Bratko, I. (1996). Handling imperfect data in inductive logic programming. In L. De Raedt (ed.) *Advances in Inductive Logic Programming*. pages 48–64. IOS Press, Amsterdam.
15. Michalski, R., Carbonell, J., and Mitchell, T., editors (1983). *Machine Learning: An Artificial Intelligence Approach*, volume I. Tioga, Palo Alto, CA.
16. Michalski, R., Mozetič, I., Hong, J., and Lavrač, N. (1986). The multi-purpose incremental learning system AQ15 and its testing application on three medical domains. In *Proc. Fifth National Conference on Artificial Intelligence*, 1041–1045. Morgan Kaufmann, San Mateo, CA.
17. Michie, D., Spiegelhalter, D.J., and Taylor, C.C., editors (1994). *Machine Learning, Neural and Statistical Classification*. Ellis Horwood, Chichester.
18. Mingers, J. (1989). An empirical comparison of pruning methods for decision tree induction. Machine Learning, 4(2):227–243.
19. Mingers, J. (1989). An empirical comparison of selection measures for decision-tree induction. *Machine Learning*, 3(4): 319–342.
20. Muggleton, S., editor (1992). *Inductive Logic Programming*. Academic Press, London.
21. Muggleton, S., Srinivasan, A., and Bain, M. (1992). Compression, significance and accuracy. In *Proc. 9th International Conference on Machine Learning*, 338–347. Morgan Kaufmann, San Mateo, CA.
22. Pirnat, V., Kononenko, I., Janc, T., and Bratko, I. (1989). Medical analysis of automatically induced rules. In *Proc. 2nd European Conference on Artificial Intelligence in Medicine*, pages 24–36. Springer, Berlin.
23. Quinlan, J.R. (1987) Simplifying decision trees. *International Journal of Man-Machine Studies*, 27(3): 221–234.
24. J.R. Quinlan.(1990) Learning logical definitions from relations. *Machine Learning*, 5(3): 239–266.
25. J. Rissanen. (1978) Modeling by shortest data description. *Automatica*, 14: 465–471.

## Appendix: Significance and information score

### Significance

CN2 can use a significance measure to enforce the induction of reliable rules. A rule is deemed reliable (significant) if the class distribution of the examples it covers is significantly different from the prior class distribution as given by the entire training set. This is measured by the likelihood ratio statistic [3].

Suppose a rule covers $r_i$ examples of class $c_i, i \in \{1, \ldots, N\}$. Let $q_i = r_i/(r_1 + \ldots + r_N)$ and let $p_i$ be the prior probability of class $c_i$. The value of the likelihood ratio statistic is then

$$2(r_1 + \ldots + r_N) \sum_{i=1}^{N} q_i log_2(q_i/p_i)$$

This statistic is distributed as $\chi^2$ with $N - 1$ degrees of freedom. If its value is above a specified significance threshold, the rule is deemed significant.

### Information score

A recent implementation of CN2 [4] can measure the classification performance of induced rules in terms of the *relative information score* [9], as well as in terms of the classification accuracy.

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:

$$I(e_k) = \begin{cases} -log_2(p_k) + log_2(p'_k) & p'_k \geq p_k \\ log_2(1 - p_k) - log_2(1 - p'_k) & p'_k < p_k \end{cases}$$

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 $e_1, e_2, ..., e_t$ belonging to one of classes $c_1, c_2, ..., c_N$ can be calculated as the ratio of the average information score of the answers and the entropy of the prior distribution of classes:

$$I_r = \frac{\frac{1}{t} \times \sum_{k=1}^{t} I(e_k)}{-\sum_{i=1}^{N} p_i \ log_2(p_i)}$$

This article was processed using the LaTeX macro package with LLNCS style