On State Merging in Grammatical Inference: A Statistical Approach for Dealing with Noisy Data

Marc Sebban MARC.SEBBAN@UNIV-ST-ETIENNE.FR
Jean-Christophe Janodet JANODET@UNIV-ST-ETIENNE.FR
EURISE, Faculty of Sciences, 23 rue Paul Michelon, University of Jean Monnet, 42023 Saint-Etienne, FRANCE

Abstract

In front of modern databases, *noise tolerance* has become today one of the most studied topics in machine learning. Many algorithms have been suggested for dealing with noisy data in the case of *numerical* instances, either by filtering them during a preprocess, or by treating them during the induction. However, this research subject remains widely open when one learns from unbounded *symbolic* sequences, which is the aim in grammatical inference. In this paper, we propose a statistical approach for dealing with noisy data during the inference of automata, by the state merging algorithm RPNI. Our approach is based on a proportion comparison test, which relaxes the merging rule of RPNI without endangering the generalization error. Beyond this relevant framework, we provide some useful theoretical properties about the behavior of our new version of RPNI, called RPNI*. Finally, we describe a large comparative study on several datasets.

1. Introduction

Thanks to recent advances in data acquisition and storage technologies, modern databases have the particularities of containing huge quantities of data, but also of presenting a high level of noise. In order to remain efficient, machine learning algorithms require more and more often specific treatments to address the problem of noisy data. Actually, the presence of strongly (and even weakly) irrelevant data can have a dramatic impact on classifier performances, not only in terms of generalization error, but also in terms of complexity. During the last decade, many data reduction algorithms have been proposed, aiming either at reducing the representation dimension by feature selection (Aha, 1992; John et al., 1994; Langley, 1994), or at removing irrelevant instances by prototype selection (Brodley & Friedl, 1996; Wilson & Martinez, 1997; Sebban & Nock, 2000; Sebban et al., 2003). One can surprisingly notice that the large majority of such data reduction techniques are only devoted to deal with numerical data, and not symbolic ones. Symbolic data reduction becomes even much more marginal when the learning data are unbounded symbolic sequences, which is notably the case in grammatical inference.

Grammatical inference is a subtopic of machine learning whose aim is to learn models of sequences, usually called *words* or *strings*, and made up of symbols, called *letters*. In this paper, we focus on particular sets of words, called *regular languages*. Roughly speaking, a language is regular if there exists some machine, called a *deterministic finite automaton* (DFA), which takes a word as input, and accepts or rejects it if it belongs or not to the language. A DFA is thus in a way a classifier which separates the set of all words in two classes, the positive one (for the accepted words) and a negative one (for the rejected words). During the last decades, many results in inductive learning theory have brought to the fore the conditions for learning DFA from a training set (de la Higuera, 1997). A practical reason which explains these efforts is that many applications, such as speech recognition, pattern matching, language processing, etc., can take advantage of the interesting structural and semantic properties of the DFA. In this paper, we only consider algorithms based on *state merging* mechanisms and in particular the one called RPNI (Oncina & García, 1992). Beyond this choice, note that all the results presented in this paper are easily extendable to other variants, such as EDSM (Lang et al., 1998).

Figure 1 describes a simple example of a DFA inferred with RPNI. Let E_+ (resp. E_-) be the set of positive (resp. negative) learning examples, such that $E_+ = \{b, ab, abb, bab\}$ and $E_- = \{aa, ba\}$. A DFA accepts a word if there exists a path, labeled with its letters, which starts from the initial state 0 (with the ingoing

Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC, 2003.

arrow), follows different transitions, and ends in the final state 0 (with a double circle). That is the case for all positive words of E_+. The negative example *ba* is rejected because it ends in a non-final state, whereas *aa* is rejected because there does not exist any path for it.

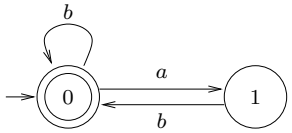

Figure 1. An example of a DFA containing two states and three state transitions.

RPNI is considered as an exact learning algorithm because it fits the data, *i.e.* a positive (resp. negative) example is always accepted (resp. rejected) by the DFA. Moreover, if the learning sample contains *characteristic* words, RPNI is theoretically able to infer, in polynomial time, the minimal target automaton (Oncina & García, 1992). However, despite these interesting theoretical properties, as soon as one treats real-world problems, one encounters two types of difficulties: firstly, it is impossible to know if such characteristic examples are present in the learning data, and if the problem is learnable with a DFA. Secondly, since RPNI achieves an exact learning, the presence of noisy data has dramatic effects. Actually, without precising now the state merging rule used in RPNI (that is the aim of the next section), we can easily think that a mislabeled example (*i.e.* a positive example which should belong to E_-, or *vice versa*) will penalize the DFA in terms of size and generalization accuracy. For instance, consider the new negative example *bbbb* in the learning sample. Figure 2 shows the new profoundly modified DFA inferred by RPNI with this noisy instance. This example expresses well the fact that RPNI is not immune to overfitting.

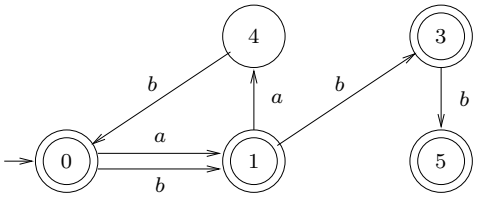

Figure 2. The effect of noisy data on the DFA.

Despite the fact that many kinds of noise can be observed in machine learning (noisy labels, noisy features, fuzzy data, incomplete data, etc.), we only consider in this paper the case where the labels can be erroneous. To deal with noisy data in grammatical inference, and extending the feature selection terminology (John et al., 1994), one could distinguish two approaches able to tackle this problem. The first one, called *filter*, would aim at removing irrelevant sequences before the induction process. In this context, we proposed a first solution in (Sebban et al., 2002) based on an entropy minimization in a neighborhood graph of sequences. We emphasized the difficulty of defining a judicious distance function on symbolic sequences, whose use has a direct effect on the selected prototypes. In order to avoid such bias, the second strategy, called *wrapper*, would aim at detecting and treating noisy data during the inference process. That is the goal of this paper. As far as we know, it is the first attempt to take into account noisy data directly in the merging rule of the inference algorithm.
The rest of this paper is organized as follows. Section 2 describes the algorithm RPNI and its state merging mechanism. In Section 3, we present our statistical framework for dealing with noisy data and the new state merging rule used in an algorithm called RPNI*. Then, Section 4 gives some theoretical properties about the behavior of DFA in the presence of noise, before a large experimental study, in Section 5, and a conclusion and some perspectives in Section 6.

2. The State Merging Algorithm RPNI

In this section we explain how RPNI works thanks to its pseudo-code presented in Algorithm 1. We also describe its state merging mechanism on the toy example already presented in the introduction (see (Oncina & García, 1992) for formal details). The first task RPNI achieves is the construction of the *prefix tree acceptor* (PTA) of the words of E_+, *i.e.* a tree-like DFA which accepts only the words of E_+ (see the upper DFA in Figure 3). States are numbered according to standard lexical ordering over prefixes (Oncina & García, 1992). RPNI then runs along these states following the ordering. When state i is considered, RPNI tries to merge it with states $0, \ldots, i-1$, in order. Merging two states means to collapse them to one new state, whose number is the smallest of the two merged ones. This state is considered as final if one of the merged states was final. As for the outgoing transitions, they are themselves merged together if they are labeled with the same letter, and in such a case, the two pointed states are recursively merged. In Figure 3, RPNI tries to merge states 1 and 0 of the PTA. This creates a loop labeled with a on state 0. Since states 1 and 0 have both an outgoing transition labeled with b, the pointed states, namely states 3 and 2, must be merged together. This leads to the middle DFA of Figure 3.

Algorithm 1 Pseudo-code of RPNI

Input: sets E_+, E_- of examples
Output: a DFA $\mathcal{A}$

```
A ⟵ PTA(E+)
for i = 1 to n do
  j ⟵ 0
  while (j < i) and not_mergeable(i, j) do
    j ⟵ j + 1
  end while
  if j < i then A ⟵ merge(i, j)
end for
return(A)
```

A merging *succeeds* if no example in E_- is accepted by the resulting DFA. It *fails* otherwise. Here, the merging of states 1 and 0 fails since $aa \in E_-$ is accepted by the resulting DFA. So RPNI abandons the merging of 1 and 0 and tries to merge 2 and 0. This leads to the lower DFA of Figure 3 that does not accept any example of E_-. So RPNI takes this new DFA and merges 3 and 0 with success, leading to the DFA of Figure 1, that is the global result of RPNI on the data.

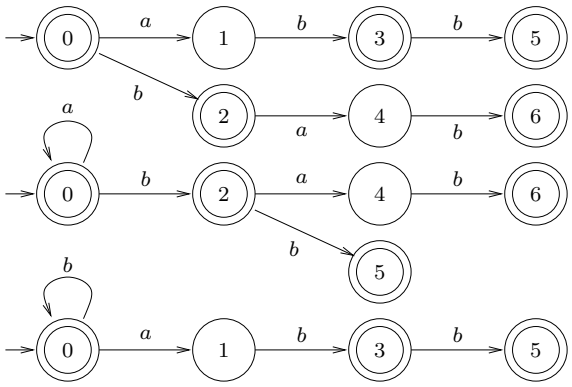

Figure 3. The upper DFA is the PTA of $\{b, ab, abb, bab\}$. The middle DFA results of the merging of 1 and 0. The lower results of the merging of 2 and 0.

3. A Statistical Test to Deal with Noise

We describe here our statistical approach for treating noisy data. We slightly modify the merging rule of RPNI in order to tolerate the presence of noise. Our approach is based on a proportion comparison test, which verifies that the proportion of *misclassified* examples does not significantly increase after a state merging.

3.1. Definitions and Notations

So far, two states were mergeable in RPNI if no negative example was accepted by the current DFA. Stated differently, it means that all the states of the DFA are used either to accept only positive examples, or to reject only negative examples. However, in the presence of noise, the parsing of some positive and negative examples may lead to the same state. In this case, RPNI would systematically reject such merging whereas it should have been accepted in the absence of noise. So, we aim here at relaxing the merging constraint by authorizing the presence of some *misclassified* examples in the final DFA.

We say that *an example w is within a state s* if its parsing by the DFA terminates to s. We say that a state is *positive* if it contains more positive words (of E_+) than negative ones (of E_-). We say that it is *negative* otherwise. A negative (resp. positive) example within a state s is *misclassified* if s is positive (resp. negative). We now modify the merging rule as follows: a merging is statistically *acceptable* if and only if the proportion p_2 of misclassified examples in the whole DFA after the merging is not significantly higher than the proportion p_1 computed before the merging.

This way to proceed allows us to avoid overfitting phenomena, due to the presence of noise, and resulting in the construction of large DFA. We accept then to reduce, by state merging, the DFA size if it does not result in a significant increase of the error. Since the sampling fluctuations on the learning set can have an influence on a given merging decision, a simple comparison between p_1 and p_2 would not be relevant. A judicious statistical rule would require in fact the use of a comparison test with confidence intervals.

3.2. Test of Proportion Comparison

Let H_0 be the null hypothesis of this test, which expresses the idea that a given merging does not statistically increase the proportion of misclassified examples in the learning set, *i.e.* without reasonably endangering further classification tasks. In such a context, we test the following null hypothesis $H_0 : p_1 = p_2$, versus the alternative one $H_a : p_2 > p_1$. Note that it is a one-tailed test, where only a sufficiently large value of our statistic (here $p_2 - p_1$) will lead to rejection of the hypothesis tested. Actually, a small value of the statistic (and of course a negative one) does not challenge the quality of the merging. The quantities p_1 and p_2 are unknown, because they correspond to the theoretical errors of the current DFA respectively before and after the merging. They can only be assessed by the empirical errors $\hat{p}_1 = \frac{N_1}{N}$ and $\hat{p}_2 = \frac{N_2}{N}$ computed from the learning set, where N_1 (resp. N_2) is the number of misclassified learning examples before (resp. after) the merging, and N is the learning set size. $\hat{p}_1$ and $\hat{p}_2$ are independent random variables and are unbiased

estimators of p_1 and p_2. In our test, we are interested in the difference $\hat{p_2} - \hat{p_1}$ which has the following mean and variance under the null hypothesis H_0:

$$E(\hat{p_2} - \hat{p_1}) = p_2 - p_1 = 0$$
$$Var(\hat{p_2} - \hat{p_1}) = \frac{p_2(1-p_2)}{N} + \frac{p_1(1-p_1)}{N} = \frac{2pq}{N}$$

where $p = p_1 = p_2$ under H_0 and $q = 1 - p$. p is usually estimated by the mean of the two proportions of misclassified examples before and after the merging:

$$\hat{p} = \frac{1}{2}(\frac{N_1 + N_2}{N})$$

If the constraints $Np > 5$ and $Nq > 5$ are verified[1], the approximation conditions to the normal distribution are satisfied, and then the variable $T = \hat{p_2} - \hat{p_1}$ follows the normal law $N(p_2 - p_1, \sqrt{\frac{2\hat{p}\hat{q}}{N}})$. We have to determine the threshold Z_α, called the critical value at the risk α, which defines the bound of the rejection of H_0, and which corresponds to the $(1-\alpha)$-percentile of the distribution of T under H_0. It means that:

$$P(T > Z_\alpha) = P(\frac{T - (p_2 - p_1)}{\sqrt{\frac{2\hat{p}\hat{q}}{N}}} > \frac{Z_\alpha - (p_2 - p_1)}{\sqrt{\frac{2\hat{p}\hat{q}}{N}}})$$

$$= P(T^{cr} > \frac{Z_\alpha}{\sqrt{\frac{2\hat{p}\hat{q}}{N}}})$$

$$P(T > Z_\alpha) = \alpha \text{ } \textit{iff} \text{ } Z_\alpha = U_\alpha . \sqrt{\frac{2\hat{p}\hat{q}}{N}}$$

where T^{cr} is the centered and reduced variable and U_α is the $(1-\alpha)$-percentile of the normal law N(0,1). If $T > Z_\alpha$ we reject H_0 with a risk of $\alpha\%$. On the contrary, if $T < Z_\alpha$, the merging is statistically validated, and then accepted. We use this new statistical merging rule in a slightly modified algorithm, called RPNI*. We do not present here the pseudo-code of RPNI* which is the same as RPNI except for the merging rule.

3.3. RPNI* is a generalization of RPNI

Our approach is able not only to deal with noisy data but also to remain relevant in noise-free situations. Actually, RPNI* is tested with different values of the parameter α. We keep the optimal value for which the best DFA, in terms of error and number of states, is inferred. Nevertheless, RPNI* is strictly equivalent to RPNI when data is noise-free. Indeed, RPNI must not accept misclassified examples in order to infer the target DFA. This situation is possible with RPNI* if $U_\alpha = 0$, *i.e.* when $\alpha = 0.5$. In this case, the merging is refused if $T > 0$, *i.e* if $p_2 > p_1$. When data is noise-free, $p_1 = 0$ at the first step of the algorithm. This case represents a merging for which $p_2 > 0$ would always be refused, that is strictly the merging rule of RPNI. In conclusion, RPNI* does generalize RPNI to noisy data.

[1]Otherwise, a Fisher exact test could be used.

4. Theoretical Results with Noisy Data

In this section, we provide some theoretical results about the behavior of procedures which would treat noise in grammatical inference such as RPNI*. Before presenting specific experimental results with RPNI*, we aim at studying until which level of noise such procedures are able to work. In our context, RPNI* will achieve perfectly its task if it finds the target DFA, despite the presence of noisy data. Since RPNI* reduces the merging constraints of RPNI, a given state can now contain not only negative but also positive examples. The label of this state is then given by a majority rule. We can intuitively think that the probability to have a mislabeled state (*i.e.* a positive (resp. negative) state which should be negative (resp. positive) in the absence of noise increases with the level of noise. Such increase should have a direct consequence on the DFA performances. We can also think that even an optimal procedure will slightly diverge in terms of error with the increase of noise. Proving these phenomena would allow us not only to justify the small divergence of RPNI* (that we will note in the experimental section), but also to provide an estimation of this deviation. That is the goal of the following sections.

4.1. Probability of a State Mislabeling

We only study here the situation of a negative state which should be positive. We aim at computing the probability that this state is in fact mislabeled because of noise. The reasoning remains the same for the opposite situation.

Let s be a state with n_1 positive and n_2 negative examples. We assume without loss of generalization that $n_2 > n_1$, so that s is labeled negatively. Among the n_1 positive examples, let n_+ be the number of instances mislabeled because of noise (they should be negative). Among the n_2 negative examples, let n_- be the number of mislabeled instances. Assuming that the noise is uniformly distributed on the learning set, the state s will be mislabeled if and only if: (i) n_- is higher than n_+. If $n_- < n_+$, we could not actually have a different label for s because we assume $n_2 > n_1$. Then, s could not be mislabeled; (ii) n_- is high enough not only to

compensate for the n_+ positive mislabeled examples, but also to change the label of s from '+' to '-'.

Theorem 4.1 *The probability $P_s(\gamma, n_1, n_2)$ of having a mislabeled state s because of the presence of a noise rate γ is equal to:*

$$\sum_{n_+=0}^{n_1} \binom{n_1}{n_+} \gamma^{n_+} \bar{\gamma}^{n_1-n_+} \sum_{n_- > n_+ + \frac{n_2-n_1}{2}}^{n_2} \binom{n_2}{n_-} \gamma^{n_-} \bar{\gamma}^{n_2-n_-}$$

where $\bar{\gamma} = 1 - \gamma$.

Proof

Let X_1 be the random variable of the number of mislabeled examples among the n_1 positive ones. X_1 corresponds to the number of successes in n_1 independent trials with a probability γ. Then X_1 follows the Binomial law $B(n_1, \gamma)$ whose probability function is:

$$P(X_1 = n_+) = \binom{n_1}{n_+} \gamma^{n_+} (1-\gamma)^{n_1-n_+}.$$

With the same reasoning, let X_2 be the number of mislabeled examples among the n_2 negative ones. Then,

$$P(X_2 = n_-) = \binom{n_2}{n_-} \gamma^{n_-} (1-\gamma)^{n_2-n_-}.$$

We can now finish the construction of $P_s(\gamma, n_1, n_2)$. While there is no constraint on n_+ whose values can vary from 0 to n_1, the quantity n_- is conditional to n_+ (we saw that $n_- > n_+$). So, given a value for n_+, the only condition for having a mislabeled state s is:

$$n_1 + n_- > n_2 - n_- \text{ } \textit{iff} \text{ } n_- > \frac{n_2 - n_1}{2}.$$

From this fact, we deduce that:

$$P_s(\gamma, n_1, n_2) = \sum_{n_+=0}^{n_1} P(X_1 = n_+) \sum_{n_- > n_+ + \frac{n_2-n_1}{2}}^{n_2} P(X_2 = n_-).$$

□

4.2. Theoretical Divergence of $P_s(\gamma, n_1, n_2)$

Theorem 4.2 *The probability $P_s(\gamma, n_1, n_2)$ is an increasing function of the noise γ.*

Proof

We have already shown that

$$P_s(\gamma, n_1, n_2) = \sum_{n_+=0}^{n_1} P(X_1 = n_+) \sum_{n_- > n_+ + \frac{n_2-n_1}{2}}^{n_2} P(X_2 = n_-)$$

$$= \sum_{n_+=0}^{n_1} P(X_1 = n_+) P(X_2 > n_+ + \frac{n_2 - n_1}{2}) = P(X_2 - X_1 > \frac{n_2 - n_1}{2})$$

Since the probability density function of a Binomial law is difficult to manipulate, we use here its convergence properties to the Normal law. In such a context, X_1 and X_2 follow asymptotically the standard normal distribution $X_1 \approx N(n_1\gamma, \sqrt{n_1\gamma\bar{\gamma}})$ and $X_2 \approx N(n_2\gamma, \sqrt{n_2\gamma\bar{\gamma}})$.

Since X_1 and X_2 are independent variables following the normal law, the difference $X_2 - X_1$ also follows a normal distribution with parameters:

$$E(X_2 - X_1) = E(X_2) - E(X_1) = (n_2 - n_1)\gamma$$
$$V(X_2 - X_1) = V(X_2) + V(X_1) = (n_1 + n_2)(\gamma\bar{\gamma})$$

We deduce that:

$$P_s(\gamma, n_1, n_2) = P(X_2 - X_1 > \frac{n_2 - n_1}{2}) =$$

$$P(N(0,1) > \frac{\frac{n_2-n_1}{2} - E(X_2 - X_1)}{\sqrt{V(X_2 - X_1)}}) = P(N(0,1) > u(\gamma))$$

where $u(\gamma) = \frac{n_2-n_1}{2\sqrt{n_1+n_2}} \frac{1-2\gamma}{\sqrt{\gamma\bar{\gamma}}}$. It is easy to show that:

$$\frac{\partial u}{\partial \gamma} = \frac{n_2 - n_1}{4\sqrt{n_1 + n_2}} \frac{-1}{(\gamma\bar{\gamma})^{\frac{3}{2}}}.$$

Since $n_2 > n_1$, we deduce that $\frac{\partial u}{\partial \gamma} < 0$, thus that $u(\gamma)$ decreases on $[0, 1]$. Therefore $P_s(\gamma, n_1, n_2)$ is an increasing function of γ. □

We have seen that $P_s(\gamma, n_1, n_2)$ corresponds to the probability of having a mislabeled state s. What is the consequence of its increase on the state performances? Since each state is, in a way, a sub-classifier of the final DFA, answering this question would allow to assess the effect of noise on the behavior of the DFA.

4.3. Margin Expression in Terms of $P_s(\gamma, n_1, n_2)$

When an example w stops on a state s, it inherits the label of s (*positive* or *negative*). Since $P_s(\gamma, n_1, n_2)$ measures the risk to have a mislabeling on s, the quantity $[1 - P_s(\gamma, n_1, n_2)]$ expresses the confidence in this classification. Actually, w is correctly classified with a probability $[1 - P_s(\gamma, n_1, n_2)]$ and misclassified with a probability $P_s(\gamma, n_1, n_2)$. According to the margin theory used in support vector machines or in boosting (Schapire et al., 1998), the classification margin for an example is defined as the difference between the weight assigned to the correct label and the maximal weight assigned to any single incorrect label. Schapire et al.

(1998) proved that an improvement in this measure of confidence on the learning set guarantees an improvement in the upper bound on the generalization error. In our case, we can take into account $P_s(\gamma, n_1, n_2)$ in the definition of the margin of each learning example w. Consider that there are n_{max} (resp. n_{min}) examples of the majority (resp. minority) class in s. For the n_{max} examples the weight assigned to the correct (resp. incorrect) label is equal to $1 - P_s(\gamma, n_1, n_2)$ (resp. $P_s(\gamma, n_1, n_2)$), and then the margin is:

$$m(w) = 1 - 2P_s(\gamma, n_1, n_2).$$

Note that the margin is a number in the range $[-1, +1]$ and that an example is classified correctly iff its margin is positive. Moreover, a large positive margin can be interpreted as a "confident" correct classification. The margin is equal to 1 when γ, and then $P_s(\gamma, n_1, n_2)$, is null. For the n_{min} examples, the margin is:

$$m(w) = 2P_s(\gamma, n_1, n_2) - 1.$$

This margin is negative for a value of $P_s(\gamma, n_1, n_2)$ smaller than 0.5. Beyond 0.5, the margin becomes positive, that means that, despite its belonging to the minority class, the label of w is in fact the true one. More generally, when the noise increases, we proved in the previous section that $P_s(\gamma, n_1, n_2)$ increases too, resulting in a drop of the margin for the n_{max} examples and an increase for the n_{min} ones. Since $n_{max} > n_{min}$, the mean of the margins decreases on each state, resulting in an increase of the generalization error according to Schapire's margin theory.

Expressing the margin in terms of $P_s(\gamma, n_1, n_2)$ allows us to make the following remarks. Firstly, as we will see in the experimental section, small values of γ often lead to a null value of $P_s(\gamma, n_1, n_2)$, resulting in a margin equal to 1. It means that despite the presence of misclassified examples during the learning step, we can theoretically learn the target function, and then infer the DFA guaranteeing a null generalization error. Secondly, the more γ increases, the more $P_s(\gamma, n_1, n_2)$ differs from 0, justifying, even for an optimal algorithm, a slight divergence of the generalization error using the DFA. Despite this, we will see in the next section that RPNI* is much better than RPNI, and is thus a relevant way to deal with noisy data.

5. Experimental Results

In this section, we assess the efficiency of RPNI* according to the two following performance measures: *generalization accuracy* and *number of states*. To achieve this task, we carried out three types of test. The first one aims at studying RPNI*'s behavior on a specific database according to different levels of noise. This way to proceed will be useful to experimentally confirm the remarks we expressed in the previous section. With the second series of experiments, we will compare, for a given level of noise, RPNI* and RPNI on two types of datasets: (i) 8 synthetic databases artificially generated using the simulator Gowachin[2], and (ii) 3 databases of the UCI database repository [3]. We also test the algorithms on a French first name database. Finally, we will study RPNI*'s tolerance to different levels of noise and on different databases. In these last experiments, γ will evolve from 0 to 20%.

During all the experiments, we used a cross-validation procedure (with 10 folds) to estimate the generalization error. To verify if the target DFA has been correctly inferred, only the learning set (*i.e.* a set of 9 folds at each step) was built from noisy data, the validation set remaining noise-free. Then, for the 8 synthetic databases, we aimed at obtaining a null error.

5.1. Validation of the Theoretical Properties

We aim here at experimentally verifying the remarks we expressed in the previous section. To achieve this task, we used a given dataset, called BASE1, that we simulated via Gowachin. The 2000 instances are originally noise free, and using RPNI on BASE1 we are then able to infer the target DFA, with 9 states, and guaranteeing a 100% generalization accuracy. We improved the noise rate γ from 0 to 20% and compared RPNI and RPNI* in terms of accuracy and number of states.

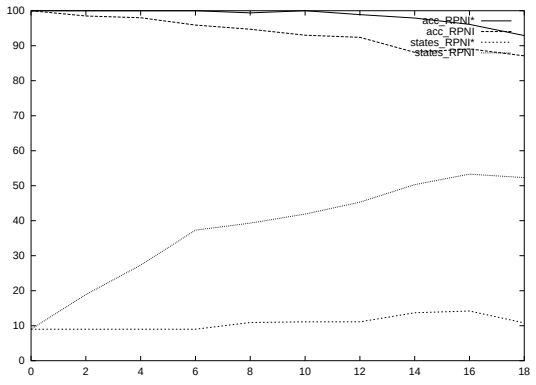

Figure 4. Results on the dataset BASE1: acc_RPNI* (resp. acc_RPNI) represents RPNI*'s accuracy (resp. RPNI's accuracy) estimated by cross-validation and states_RPNI* (resp. states_RPNI) represents the number of states of the DFA inferred using RPNI* (resp. RPNI).

From the results described in Figure 4, we can note that until 6%, RPNI* is able to totally tolerate noisy

[2] http://www.irisa.fr/Gowachin

[3] http://www.ics.uci.edu/~mlearn/MLRepository.html

data. It means that RPNI* infers, for the 10 folds of the cross-validation procedure, the target DFA with 9 states and guaranteeing a 100% generalization accuracy. It confirms that for small noise values, margins are equal to 1 despite the presence of misclassified examples on some states. From 6% to 10%, RPNI* still allows us to have a null generalization error, despite slightly larger DFA (10 states on average). Beyond 10%, the level of noise is too important to be totally tolerated, resulting in a little divergence of the performances in terms of accuracy and number of states. That confirms the theoretical results presented in the previous section, even if this degradation is relative, authorizing an efficient behavior (an accuracy about 93% and 11 states on average for 20% of noise). Concerning RPNI, the acknowledgment of failure is glaring. As soon as the noise appears, the degradation of the performances is perceptible. Not only the generalization accuracy regularly drops, but also the DFA size increases conveying an overfitting phenomenon.

5.2. RPNI*'s Behavior on 12 Datasets

We carried out a large comparative study on 12 datasets between RPNI* and RPNI for a level of noise $\gamma = 4\%$. The goal is to provide significant results about RPNI*'s efficiency on a large panel of datasets, described in Table 1. Note that for the eight simulated databases BASE1, ..., BASE8, we *a priori* know the target DFA, and RPNI is able to infer it in the absence of noise ($\gamma = 0$). That is not the case for the 4 other datasets, called AGARICUS, BADGES, PROMOTERS and FIRSTNAME, for which we do not know the target DFA.

Each value in Table 1 represents the mean computed from the 10 cross-validation results. Many interesting remarks can be made. Firstly, the difference between RPNI* and RPNI is significant for 10 datasets using a Student paired t-test over accuracies. Only AGARICUS and BASE6 do not satisfy the statistical constraint. However, we can note that RPNI is never better than RPNI*, not only in terms of accuracy but also concerning the DFA size. Moreover, the difference is highly significant over the global means of accuracies (88.1 vs 82.9). The remark is similar concerning the DFA size. Actually, over the 12 datasets, while the number of states does not dramatically increase with RPNI* (25.6 states on average), the mean with RPNI (95.6) conveys the difficulties to infer the target DFA resulting to an overfitting phenomenon. Finally, note that despite 4% of noise, RPNI* is able to infer the target DFA for 3 datasets (BASE1, BASE6 and BASE8), and allows a 100% generalization accuracy for a fourth dataset (BASE7).

Table 1. Results of RPNI* and RPNI on 12 datasets. Means and standard deviations are computed from the cross-validation results. The number of states of the target DFA is indicated between brackets for the simulated datasets.

	RPNI*		RPNI	
DATASET	ACCURACY	NBSTATES	ACCURACY	NBSTATES
BASE1 (9)	100 ± 0	9 ± 0	98.0 ± 1.15	27.3 ± 5.3
BASE2 (9)	94.8 ± 8.0	14.1 ± 2.2	86.3 ± 4.5	39.8 ± 2.6
BASE3 (21)	94.1 ± 4.0	29.2 ± 2.8	86.5 ± 3.2	192.7 ± 5.5
BASE4 (38)	99.7 ± 0.4	59.5 ± 4.5	92.2 ± 1.1	252.5 ± 10.7
BASE5 (38)	89.5 ± 5.8	52.2 ± 4.9	85.2 ± 2.5	284.6 ± 9.5
BASE6 (10)	100 ± 0	10 ± 0	99.8 ± 0.2	17.4 ± 5.5
BASE7 (12)	100 ± 0	16.5 ± 0.5	95.8 ± 1.1	131.4 ± 10.7
BASE8 (15)	100 ± 0	15 ± 0	98.9 ± 2.5	77 ± 9.5
AGARICUS	88.9 ± 1.7	82.7 ± 3.3	88.9 ± 1.7	82.7 ± 3.3
BADGES	72.1 ± 7.6	2.3 ± 0.4	55.4 ± 10.5	8.7 ± 0.9
PROMOTERS	51.9 ± 15.7	5.3 ± 0.5	48.8 ± 21.7	23.1 ± 0.9
FIRSTNAME	66.5 ± 8.8	3.5 ± 0.5	59.5 ± 9.1	10.0 ± 0.6
AVERAGE	88.1	25.6	82.9	95.6

Another concise way to display the results is proposed in Figure 5. Each dataset is plotted according to 2 coordinates: x which corresponds to RPNI's accuracy and y which represents RPNI*'s accuracy. We can note that all the dots are over the bisecting line $y = x$ expressing that RPNI* is always better than RPNI.

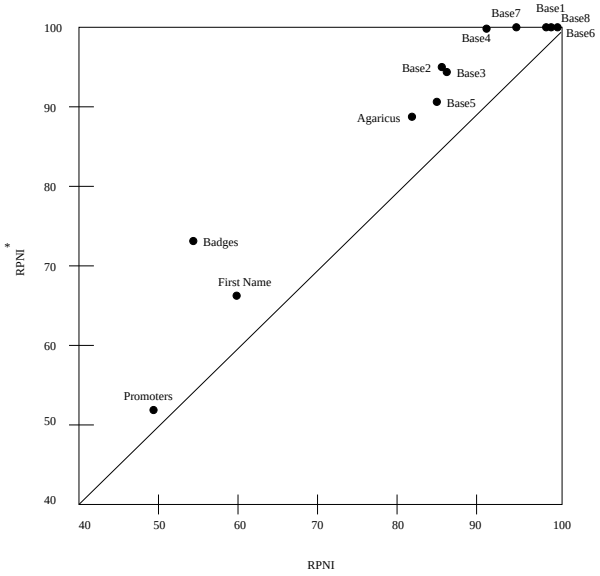

Figure 5. Scatter plot on 12 datasets.

5.3. RPNI*'s Tolerance to Noise

We aim here at studying RPNI*'s tolerance to different levels of noise. In these last experiments, γ will evolve from 0 to 20%. Results are displayed in Figure 6. Note that each point of the curves represents the mean over the 12 datasets. Globally, the difference between RPNI and RPNI*, of course in favor of the latter, increases with the level of noise. Not only it means that RPNI is always less efficient than RPNI*, but also it expresses that RPNI's degradation is larger. Concerning the generalization accuracy, we logically note that

the performances of both algorithms decrease with the level of noise, that confirms once again the theoretical divergence properties. While RPNI* seems to be able to effectively control the size of the DFA (from 18 to 30 states on average), RPNI seems to suffer from the increase of noise. Actually, it infers on average a DFA with 160 states when the level of noise is of 20%. That strengthens the idea that RPNI is not immune to overfitting.

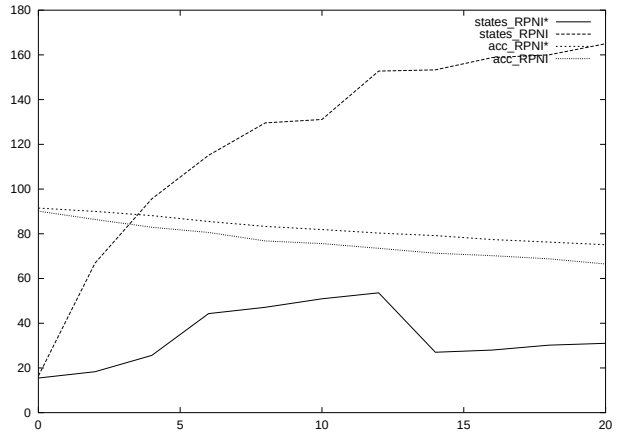

Figure 6. Accuracy and number of states *w.r.t.* different levels of noise.

6. Conclusion and Future Research

In this paper, we proposed a statistical approach for dealing with noisy data in grammatical inference. So far, our strategy has just been applied on RPNI, whose merging rule is relatively simple. However, other algorithms propose more sophisticated rules, such as EDSM (Lang et al., 1998). An extended adaptation of our approach deserves then further investigations. Moreover, we saw that RPNI* presents a good noise tolerance until a certain quantity of noise (about 10% for the artificial datasets). We think that we can improve this threshold by studying the nature of the misclassified examples before achieving any merging. In fact, the classification errors on the learning set can be the result of two phenomena. The first one is directly due to the presence of noise. The second one is a bad effect of the less restricting merging rule of RPNI*, which can (wrongly) accept some true counter-examples. So far, we did not differentiate these two types of misclassified examples. However, while the first category (the noisy data) should not be in theory opposed to the merging, the second one (the counter-examples) must not be merged with examples of the concept for fear of problems to learn the target DFA. Finally, we assumed in this paper that the noise only affects the label of a sequence. While this assumption is not too strong for treating a large panel of real world problems, we are thinking of adapting our method to cases where some letters of the sequences are also noisy.

References

Aha, D. (1992). Tolerating noisy, irrelevant and novel attributes in instance-based learning algorithms. *Int. Journal Man-Machine Studies*, 267–287.

Brodley, C., & Friedl, M. (1996). Identifying and eliminating mislabeled training instances. *Thirteenth National Conference on Artificial Intelligence* (pp. 799–805).

de la Higuera, C. (1997). Characteristic sets for polynomial grammatical inference. *Journal of Machine Learning*, *27*, 125–138.

John, G., Kohavi, R., & Pfleger, K. (1994). Irrelevant features and the subset selection problem. *Eleventh Int. Conference on Machine Learning* (pp. 121–129).

Lang, K., Pearlmutter, B., & Price, R. (1998). Results of the abbadingo one DFA learning competition and a new evidence-driven state merging algorithm. *Fourth Int. Colloquium on Grammatical Inference* (pp. 1–12).

Langley, P. (1994). Selection of relevant features in machine learning. *AAAI Fall Symp. on Relevance.*

Oncina, J., & García, P. (1992). *Inferring regular languages in polynomial update time*, vol. 1 of *Machine Perception and Artificial Intelligence*, 49–61. World Scientific.

Schapire, R., Freund, Y., Bartlett, P., & Lee, W. (1998). Boosting the margin: a new explanation for the effectiveness of voting methods. *The Annals of Statistics*, *26*, 1651–1686.

Sebban, M., Janodet, J.-C., & Yahiaoui, A. (2002). Removing noisy data in grammatical inference (in french). *Seventh National Conference on Clustering* (pp. 311–314).

Sebban, M., & Nock, R. (2000). Instance pruning as an information preserving problem. *Seventeenth Int. Conference on Machine Learning* (pp. 855–862).

Sebban, M., Nock, R., & Lallich, S. (2003). Stopping criterion for boosting-based data reduction techniques: from binary to multiclass problems. *Int. Journal of Machine Learning Research*, *3*, 863–885.

Wilson, D., & Martinez, T. (1997). Instance pruning techniques. *Fourteenth Int. Conference on Machine Learning* (pp. 404–411).