Abstract. Grammatical inference is the problem of learning a language from examples and counter-examples of words in this language. The common formulations of this problem make the fundamental hypothesis of the existence of a formal language underlying the data, to be discovered. Here we follow another approach, based on the following remarks. First of all, the initial hypothesis of a formal language to be identified does not hold for real data. Secondly, the algorithmic complexity of language identification is huge, even in the case of the simplest class of languages. Our approach aims at removing these two limitations. It allows the grammars produced to discriminate the sample words imperfectly, while it introduces the use of classic optimization techniques. Here we apply Tabu Search to the inference of regular grammars.

1 Introduction

The problem of grammatical inference is commonly stated as follows (e.g. [Go67] or [Ang88]): given a language $L$ on $X^*$, and two sets of strings $I^+ \subseteq L$ and $I^- \subseteq X^* \sim L$, find a description for $L$ (i.e. a grammar, an automaton or an expression). For the most part, this formalization led to the development of methods that deal with the inference of regular grammars, i.e. the simplest in the Chomsky hierarchy. In conformity with the basic hypothesis of a language to be learnt, these methods aim at producing a grammar or an automaton that describes the strings in $I^+$ and $I^-$, that is a grammar or an automaton that accepts every string in $I^+$ and rejects every string in $I^-$. In practice, the grammatical inference algorithms have to be adapted in order to fit the nature of the data available in pattern recognition. A first way to palliate the inadequation between the descriptions produced and the data consists in extending the notion of recognized language, by allowing the insertion or deletion of characters in the strings of the language (error-correcting techniques). This technique makes the algorithms more efficient when applied to real data, for one thing because it allows to deal with noise. Another approach is to associate probabilities with the production rules. Several models are based on this idea: inference of stochastic grammars, fuzzy automata and to a certain extent of Markov models. The problem is better stated this way, but it is more complex too, as not only do the rules of the grammar have to be determined but also their associated probability.

However, the basic formulation corresponds to an ideal situation, for one knows that real patterns are not regular or context-free. This remark led us to consider that the point of grammatical inference is less to identify a supposedly existing language than to find a grammar that best discriminates the examples and counter-examples. Given two finite sets of strings $I^+$ and $I^-$, the problem comes to optimize the quality of the discrimination the grammar induces on $I^+$ and $I^-$. The descriptions produced are formal grammars as in the classic formulation, but this approach allows more flexibility in the description of the data (e.g. it takes account of atypic examples or
badly classified data), and can be combined with the existing mechanisms (particularly the error-correcting techniques).

The methods corresponding to this approach differ from the classic ones. It is a priori possible to apply the standard optimization techniques, as well as using the algebraic properties of the class under consideration to build specific methods. Here we study the inference of the simplest class of grammars (regular grammars) using Tabu Search. The tests are run on a typical set of difficult problems, the goal being the production of a grammar that exactly discriminates the examples and counter-examples.

2 Identifying vs. Discriminating

The presentation method for our approach corresponds to the standard situation in inductive learning when two finite sets (positive and negative instances) are given to the system, from which a concept is to be produced.

2.1 Discriminating Finite Samples

The identification of a language from examples and counter-examples can be assimilated to the production of a language that exactly discriminates these sets of strings. Here we relax the identification constraint, allowing the algorithm to produce a solution that imperfectly discriminates the samples:

Given two finite sets of strings $I^+$ and $I^-$, and a set $E$ of grammars, find the grammar of $E$ that induces the best discrimination on $I^+$ and $I^-$. 

The set $E$ of grammars has to be determined in accordance with the aims of the programmer. In the next section we suggest as a general setting that $E$ be restricted to the set of grammars with a bounded number of non terminals, which allows the grammars to realize an inductive leap, and restricts the search to a finite space.

The descriptions produced are formal grammars as in the classic formulations, but this approach allows more flexibility in the description of the data:

- The case of atypic examples is treated correctly. While such an example has an effect on the solution produced by classic methods (since it must be accepted by this solution), an imperfect discrimination treats them explicitly as particular cases, which are excluded from the description. The same remark holds for atypic counter-examples.
- This approach allows dealing with noisy or badly classified data. In practice this case is similar to the previous one.
- Real data often corresponds to complex languages. While identifying such a language is difficult, approximating the language underlying the data with a formal language is a less demanding task. It allows the production of grammars which may not be exact solutions but constitute a better or worse description of the target language, according to the quality of the method. Within this framework, the identification of a language determined in advance can be seen as a limit case, to be used as a validation procedure.