

Building partial domain theories from explanations

Eva Armengol

We propose the use of the explanations provided by a lazy learning method to build a domain theory. Explanations are understood here as generalizations and as such we interpret them as domain rules in the same sense that eager learning methods do. Differently than domain theories generated by eager learning methods, theories generated from explanations are partial. In this paper the utility of such partial theories according to the domain complexity is discussed. Moreover the equivalence of "global" in front of "partial" domain theories is compared from experimental results.

1 Introduction

From early expert systems, the idea of building computational systems capable of explain the results they produce has been an issue of great interest. In fact, this interest is currently increasing with the confirmation that most of automatic systems remain as academic prototypes due to the lack of confidence that the user has in the result.

The literature offers a wide population of works where authors propose several methods for constructing explanations. There are two main uses of explanations: those oriented to explain the result to the user and those oriented to be used by the system itself. In the present paper we propose to use the explanations for both. Because we focus on classification problems, the explanations we propose can be shown to the user to justify the classification. However, we point out that these explanations can form domain theory that can be used by the system when solving new problems.

This paper is organized as follows. First, we show several examples of both explanations for users and explanations for the system. In section 2 we justify why generalizations built by some systems can be interpreted as explanations. Section 3 proposes the use of explanations to form domain theories and section 4 shows how to build this theory. Finally, section 5 shows some experiments oriented to compare eager and lazy theories.

1.1 User-oriented Explanations

The first expert system concerned with an explanation module was MYCIN [6]. MYCIN used rules to represent the domain knowledge, therefore explanations consisted on showing the user the chain of rules supporting the proposed solution, i.e. the problem solving trace. Very early, authors detected that this kind of explanations was useful for knowledge engineers but that sometimes domain experts did not completely understand the procedure followed by the system. The main conclusion was that the problem solving trace represents the *line of reasoning* whereas most of times a domain expert is interested in the *line of explanation*, i.e. the user wants the justification of the result and not necessarily details of reasoning explaining that result (see [26] for an in-depth discussion about both lines).

Approaches such as XPLAIN [25] and NEOMYCIN [8] take the distinction between line of reasoning and line of expla-

nations to propose methods oriented to build appropriate explanations. The REX system [26] proposes the *reconstructive explanation* approach, consisting on a separate expert system that builds the explanation of the solution. A different approach was that taken by SWALE [24] that defines several explanation schemes according to the user to which the explanation is oriented. See recent approaches on explanations in [23, 12]

Different approaches are those taken by case-based reasoning systems (CBR) where no rules are used. The basis of CBR is to solve new problems based on their similarity with other known and solved problems. Commonly, explanation of CBR systems consists on showing the set of cases assessed as the most similar to the new problem in order to explore these cases and analyze their similarity with the new one. This approach seems not to be useful when cases are described using a lot of attributes and/or when these cases have not clear similarity with the new problem. A good summary of some of the approaches to explanation in CBR can be found in [21].

1.2 System-oriented Explanations

System-oriented explanations are those reusable by the system when solving new problems. This kind of explanations have several alternative purposes such as to improve the system efficiency, to perform system maintenance tasks or to reorganize the memory of cases in CBR systems.

The family of methods called *Explanation-based learning (EBL)* [19] solves a problem and then analyzes the problem solving trace in order to generalize it. The generalized trace is an *explanation* that is used as a new domain rule for solving new problems more efficiently. This explanation is represented using the same formalism as the problems, therefore it is perfectly understandable and usable by the system. Examples of systems using EBL are EBG [19], PRODIGY [7] and SOAR [15] among others. Several works on CBR generate some kind of structure that can be interpreted as an explanation. An example of such system is CHEF [13] whose goal is to produce a cooking recipe from a set of ingredients. The process followed by CHEF is to retrieve a similar recipe and to adapt it by replacing some ingredients. The process of discovering the causes of failures on recipes and how to solve them is an automatic process that can be seen as an explanation that the system builds for itself. Another approach is