

Affordable Discovery of Positive and Negative Rules in Knowledge-Bases

US

ABSTRACT

We present KRD, a system for the discovery of declarative rules over knowledge-bases (KBs). KRD does not limit its search space to rules that rely on “positive” relationships between entities, such as “if two persons have the same parent, they are siblings”, as in traditional mining of constraints for KBs. On the contrary, it extends the search space to discover also negative rules, i.e., patterns that lead to contradictions in the data, such as “if two persons are married, one cannot be the child of the other”. While the former class is fundamental to infer new relationships in the KB, the latter class is crucial for error detection in data cleaning, or for the creation of negative examples when bootstrapping learning algorithms.

The main technical challenges addressed in this paper consist in enlarging the expressive power of the considered rules to include comparison among constants, including disequalities, and in designing a disk-based discovery algorithm, effectively dropping the assumption that the KB has to fit in memory to have acceptable performance. To guarantee that the entire search space is explored, we formalize the mining problem as an incremental graph exploration. Our novel search strategy is coupled with a number of optimization techniques to further prune the search space and efficiently maintain the graph. Finally, in contrast with traditional ranking of rules based on a measure of support, we propose a new approach inspired by set cover to identify the subset of useful rules to be exposed to the user. We have conducted extensive experiments using both real-world and synthetic datasets to show that KRD outperform previous proposals in terms of efficiency and that it discovers more effective rules for the application at hand.

1. INTRODUCTION

2. PRELIMINARIES AND DEFINITIONS

Talk about KBs.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Copyright 20XX ACM X-XXXXX-XX-X/XX/XX ...\$15.00.

2.1 Language

Horn Rule, with the restriction of having each variables appearing twice. Extension of predicates with inequalities.

2.2 Coverage

Given a set of examples, talk about coverage and relative coverage of a rule.

2.3 Scoring Function

Given a generation and a validation set, define score for a rule and cumulative score for a set of rules.

2.4 Problem Definition

Introduce red-blue set cover as a possible solution, and say why it is not appropriate. Define the problem as computing the set that minimizes the cumulative scoring function.

3. RULES DISCOVERY

Talk about translation from Horn Rules to paths on the graph.

3.1 Literals and Constants

Generation of artificial edges to include inequalities. Substitutions of variables with constants if same value appears for each example.

3.2 Input Examples Generation

Define how we compute generation and validation set: how we generate positive and negative examples.

4. A* GREEDY ALGORITHM

Justify the use of a greedy algorithm: generating all the rules is very expensive. Better if we can prove that the minimization function algorithm is NP hard.

4.1 Optimality

Define property on why the A* algorithm produces the greedy solution. Maybe study when the greedy solution becomes optimal? (If all rules identify disjoint set of input example, then greedy solution is optimal)

5. EXPERIMENTS

5.1 Negative Rules Evaluation

Evaluation of negative rules.

5.2 Comparison Evaluation

Comparison against AMIE and evaluation of positive rules.

5.3 Machine Learning Application

DeepDive.

6. RELATED WORK

7. CONCLUSION