ProbKB: Managing Evolving Web Knowledge

Yang Chen
Department of CISE
University of Florida
Florida, USA
yang@cise.ufl.edu

Daisy Zhe Wang
Department of CISE
University of Florida
Florida, USA
daisyw@cise.ufl.edu

ABSTRACT

Abstract

1. INTRODUCTION

2. PRELIMINARIES

2.1 Markov Logic Networks

2.2 Factor Graphs

2.3 Metropolis-Hastings

3. THE PROBKB SYSTEM

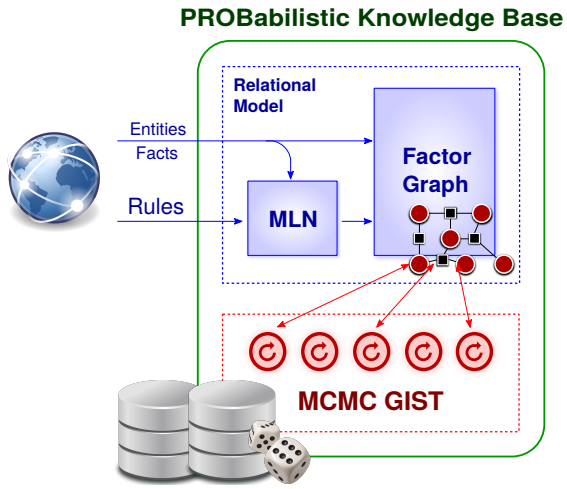

Figure 1: ProbKB system architecture.

3.1 First-Order Horn Clauses

In this paper, we focus on first-order Horn clauses only, though in general Markov logic supports arbitrary clauses. This decision is reasonable in our context since we use Markov logic to represent extracted general-purpose rules rather than hand-crafted ones encoding complex algorithms that work in specific domains. As an example of such generality, consider the NELL knowledge base [], where we have extracted entities and their relationships. Then our interest would be to know whether a certain fact `R(x,y)` is likely to hold given these extractions. This is best done by using rules such as $p \leftarrow q_1 \dots q_n$ to infer new facts using existing ones. We are less interested, however, in imposing constraints on the knowledge base using complex rules.

On the contrary, suppose we want to do joint segmentation using Markov logic, then we need more complex rules to specify assumptions and inputs to a specific algorithm. The Markov logic program provided on the Alchemy [] website shows how we can do it.

3.2 The Relational MLN Model

We restrict our attention to first-order Horn clauses. This greatly simplies the rules we need to consider.

For all rules of the form

$$p(x, y) \leftarrow q(x, z), r(z, y),$$

we have a row in a table called M_3.

3.3 Graph Aggregation

Shared memory?

3.4 Inference

3.5 Incremental Inference

4. EXPERIMENTS

4.1 Grounding

5. ACKNOWLEDGMENTS

6. REFERENCES

APPENDIX

You can use an appendix for optional proofs or details of your evaluation which are not absolutely necessary to the core understanding of your paper.