Using Probabilistic Logic Programming to Find Patterns

Theofrastos Mantadelis[1]
theo.mantadelis@dcc.fc.up.pt
Ricardo Rocha[1]
ricroc@dcc.fc.up.pt
Jorge Oliveira[2]
oliveira_jorge@dcc.fc.up.pt
Miguel Tavares Coimbra[2]
mcoimbra@dcc.fc.up.pt

[1] CRACS & INESC TEC
Faculdade de Ciências da Universidade do Porto
Rua do Campo Alegre 1021/1055,
4169-007 Porto, Portugal

[2] Instituto de Telecomunicações
Faculdade de Ciências da Universidade do Porto
Rua do Campo Alegre 1021/1055,
4169-007 Porto, Portugal

Abstract

This short paper, briefly presents the probabilistic logic programming language ProbLog and the system MetaProbLog. We present an example Hidden Markov Model to illustrate the three main tasks of the system. Furthermore, we mention some of the existing ProbLog applications which are used to find connections/patterns in relational databases. Finally, we present an application that uses MetaProbLog for phonocardiogram classification which is used in order to diagnose heart diseases.

1 Introduction

Probabilistic Logic Programming (PLP) combines technologies from logic programming, knowledge representation and reasoning and machine learning. Probabilistic models in our days are principled and a widely used approach to deal with uncertainty. First order logic can elegantly represent complex situations involving a variety of objects as well as relations among the objects.

MetaProbLog[1] [6] is a framework of the ProbLog [3, 5] probabilistic logic programming language. ProbLog extent Prolog programs by annotating facts with probabilities. In that way it defines a probability distribution over all Prolog programs. ProbLog follows the distribution semantics presented by Sato [9]. MetaProbLog extends the semantics of ProbLog by defining a "ProbLog engine" which permits the definitions of probabilistic meta calls [6]. MetaProbLog inference, currently allows the computation of marginal probabilities with or without evidence. Furthermore, it allows the computation of marginal probabilities for the answers of non-ground queries.

MetaProbLog has three primary inference methods: exact inference, program sampling and most probable explanation. The exact inference method, uses state of the art knowledge compilation methods [2]; program sampling, is a rejection sampling approach; and finally, the most probable explanation inference uses a dynamic algorithm to find the most probable explanation of a query.

2 Semantics

A ProbLog program T consists of a set of facts annotated with probabilities $p_i :: pf_i$ – called *probabilistic facts* – together with a set of standard definite clauses $h : -b_1,\ldots,b_n$. that can have positive and negative probabilistic literals in their body. A probabilistic fact pf_i is true with probability p_i. These facts correspond to random variables, which are assumed to be mutually independent. Together, they thus define a distribution over subsets of $L_T = \{pf_1,\ldots,pf_n\}$. The definite clauses add arbitrary *background knowledge* (BK) to those sets of *logical* facts. To keep a natural interpretation of a ProbLog program we assume that probabilistic facts cannot unify with other probabilistic facts or with the background knowledge rule heads. Formally, a ProbLog program is of the form $T = \{pf_1,\ldots,pf_n\} \cup BK$.

Given the one-to-one mapping between ground definite clause programs and Herbrand interpretations, a ProbLog program defines a distribution over its Herbrand interpretations.

The distribution semantics are defined by generalising the least Herbrand models of the clauses by including subsets of the probabilistic facts. If fact pf_i is annotated with p_i, pf_i is included in a generalised least Herband model with probability p_i and left out with probability $1 - p_i$. The different facts are assumed to be probabilistically independent, however, negative probabilistic facts in clause bodies allow the user to enforce a choice between two clauses.

As such, a ProbLog program specifies a probability distribution over all its possible non-probabilistic subprograms. The success probability of a query is defined as the probability that the query succeeds in such a random subprogram. ProbLog follows the distribution semantics [9] proposed by Sato.

3 Example Program & Queries

The syntax of MetaProbLog uses logic programming, specifically Prolog, in order to be very expressive as a language and be able to describe complex models. Next we present a small MetaProbLog program that defines the Hidden Markov Model illustrated at Figure 1, the different colors indicate the possible transitions from each state which are modeled by **annotated disjunctions**[2] in the program.

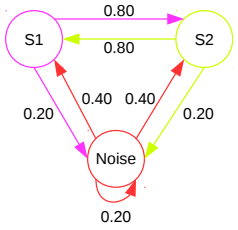

Figure 1: The graphical representation of a 3 state Hidden Markov Model

Below follows the MetaProbLog program that models the Hidden Markov Model of Figure 1.

```
0.80::trans(s1,s2,T1,T2);
0.20::trans(s1,noise,T1,T2) <- next(T1, T2).

0.80::trans(s2,s1,T1,T2));
0.20::trans(s2,noise,T1,T2) <- next(T1, T2).

0.40::trans(noise,s1,T1,T2);
0.40::trans(noise,s2,T1,T2);
0.20::trans(noise,noise,T1,T2) <- next(T1, T2).

0.20::start(noise,0);
0.40::start(s1,0);
0.40::start(s2,0) <- true.

signal(State, 0) :- start(State, 0).
signal(State2, T2) :-
  trans(State1,State2,T1,T2),
  signal(State1, T1).

next(T1, T2) :- integer(T1), !, T2 is T1 + 1.
next(T1, T2) :- integer(T2), T1 is T2 - 1.
```

For our example we use the term `trans/4` to describe a transition of the model from one state to another (first and second argument of the

[1] MetaProbLog's website: `www.dcc.fc.up.pt/metaproblog`

[2] ProbLog's syntax: `P1::Choice1 ; ⋯ ; PN::ChoiceN <- Body`, is used to model exclusive choices with $\sum P1 \cdots PN = 1.0$. This construct is called annotated disjunction and two choices of an annotated disjunction can never be true at the same time.

term), at a time to the next time (third and fourth argument of the term). By composing an annotated disjunction with the appropriate `trans/4` terms we compactly describe all the transitions our model can take. Furthermore, the term `start/2` is used to define the possible starting states. Finally, the predicate `signal/2` defines the infinite chain and dependencies of the model.

MetaProbLog can ask queries about the probability distribution of such a model. For example: we can ask what is the probability that at a specific time we have a specific state (Query 1); we can ask what is the conditional probability of a specific state using some prior knowledge like the starting state of the model (Query 2); and finally, we can query what is the most likely set of states that reach to a specific result (Query 3).

Below we present the three example queries and their results.

```
% Query 1: Probability of a state
   ?- problog_exact(signal(s1,5),P).
P = [0.4]

% Query 2: Conditional probability of a state
%          with prior knowledge
   ?- problog_exact(signal(s1,5)/start(s1,0),P).
P = [0.23616]

% Query 3: Most probable explanation of a query
   ?- problog_mpe(signal(s1,5),Res).
Res = [0.131072/[start(s2,0)->true,
                 trans(s2,s1,0,1)->true,
                 trans(s1,s2,1,2)->true,
                 trans(s2,s1,2,3)->true,
                 trans(s2,s1,4,5)->true,
                 trans(s1,s2,3,4)->true]]
```

While the ProbLog language was introduced to answer statistical questions for relational models one can easy observe that the same question could be addressed on patterns. For example: Query 3 could be asking what is the most probable pattern of the model.

MetaProbLog is able to model all statistical graphical models such as Hidden Markov Models, Bayesian Networks, Probabilistic Graphs. Furthermore, any Prolog program could be extended with MetaProbLog to use probabilities and take decisions with them.

4 Applications

ProbLog systems have found applications in many fields with most common examples to include:

- Link discovery in Biomine Alzheimer database [11]. Biomine Alzheimer database is a real-world biological dataset of Alzheimer genes which corresponds to a directed probabilistic graph of 11530 edges and 5220 nodes. ProbLog was used to discover relations among genes and other biological properties [3].
- WebKB (`http://www.cs.cmu.edu/~webkb`) is a dataset from a collective classification domain in which university webpages are classified according to their textual content. ProbLog has been used to learn the probabilities that two webpages are related and to query the WebKB [4].
- The probabilistic Dictionary [12] is used to discover the probability that two words have the same meaning. It includes around 250 different words from the English language and meanings for about 30 of them. Some words are related together according to their semantic relatedness. This relation is marked with the probability that the two words have the same meaning.
- ProbLog has also been used for robotic affordance model learning by [8]. In this application ProbLog was successfully used to learn a robotic task with multiple objects and complex spacial relations.
- Finally, ProbLog has been used to model Mobile Ad hoc Networks and analyse Fadip [7], a Publish/Subscribe protocol for Mobile Ad hoc Networks. ProbLog efficiently calculates the probability that a message would be transmitted from one device to another in the network, analysing statistics for the traffic and reachability of the protocol.

Lately, the classification of phonocardiogram (PCG) signals has got significant attention in the academic community [1]. Classifying PCGs is both a challenging and an important task. Heart sounds are non-trivial signals, since they might contain non-stationary noise, have artifacts and murmur sounds. Heart sound auscultation techniques is one of the most reliable and successful tools in early diagnosis used for potentially deadly heart diseases, such as natural and prosthetic heart valve dysfunction or even in heart failure. Therefore a computer-aided auscultation may allow detection of diseases that are hardly recognized through the traditional methods, for instance ischemic heart disease.

Recently, HMMs have being used for modeling and characterizing real-world signals such as heart sound signals [10]. For future work, we aim to model PCG signals as a HMM and use MetaProbLog to find the most likely sequence of events (S1, S2, S3, S4, noise, murmur, etc.) and finally, use our model in order to characterize real life segmented signals.

Acknowledgements

This work is funded by the Instituto de Telecomunicações in the scope of Project Rheumus (Projeto QREN no: 38505) and the Fundação para a Ciência e a Tecnologia (FCT) (Portuguese Foundation for Science and Technology) within the project UID/EEA/50014/2013.

References

[1] P. Bentley, G. Nordehn, M. Coimbra, and S. Mannor. The PASCAL classifying heart sounds challenge 2011 results. `http://www.peterjbentley.com/heartchallenge`.

[2] A. Darwiche and P. Marquis. A knowledge compilation map. *Journal of Artificial Intelligence Research*, 17:229–264, 2002.

[3] L. De Raedt, A. Kimmig, and H. Toivonen. ProbLog: a probabilistic Prolog and its application in link discovery. In *International Joint Conferences on Artificial Intelligence*, pages 2468–2473, 2007.

[4] D. Fierens, G. Van Den Broek, J. Renkens, D. Shterionov, B. Gutmann, I. Thon, G. Janssens, and L. de Raedt. Inference and learning in probabilistic logic programs using weighted boolean formulas. *Theory and Practice of Logic Programming*, 15(03):358–401, 2015.

[5] A. Kimmig, B. Demoen, L. De Raedt, V. Santos Costa, and R. Rocha. On the implementation of the probabilistic logic programming language ProbLog. *Theory and Practice of Logic Programming*, 11:235–262, 2011.

[6] T. Mantadelis and G. Janssens. Nesting probabilistic inference. *Computing Research Repository*, abs/1112.3785, 2011.

[7] T. Mantadelis, K. Paridel, G. Janssens, Y. Vanrompay, and Y. Berbers. Analysing a Publish/Subscribe System for Mobile Ad Hoc Networks with ProbLog. In *Practical Aspects of Declarative Languages*, pages 34–37, 2011.

[8] B. Moldovan, P. Moreno, M. van Otterlo, J. Santos-Victor, and L. De Raedt. Learning relational affordance models for robots in multi-object manipulation tasks. In *International Conference on Robotics and Automation*, pages 4373–4378, May 2012.

[9] T. Sato. A statistical learning method for logic programs with distribution semantics. In *International Conference on Logic Programming*, pages 715–729, 1995.

[10] P. Sedighian, A.W. Subudhi, F. Scalzo, and S. Asgari. Pediatric heart sound segmentation using Hidden Markov Model. In *Engineering in Medicine and Biology Society*, pages 5490–5493, Aug 2014.

[11] P. Sevon, L. Eronen, P. Hintsanen, K. Kulovesi, and H. Toivonen. Link discovery in graphs derived from biological databases. In *International Conference on Data Integration in the Life Sciences*, pages 35–49, 2006.

[12] D. Shterionov and G. Janssens. Data acquisition and modeling for learning and reasoning in probabilistic logic environment. In *Portuguese Conference on Artificial Intelligence*, pages 298–312, 2011.