Most Probable Explanation for MetaProbLog and its application in Heart Sound Segmentation

Theofrastos Mantadelis[1], Jorge Oliveira[2] and Miguel Coimbra[2]

{CRACS & INESC TEC[1]; Instituto de Telecomunicações[2]},
Faculdade de Ciências da Universidade do Porto
Rua do Campo Alegre 1021/1055,
4169-007 Porto, Portugal
{theo.mantadelis; oliveira_jorge; mcoimbra}@dcc.fc.up.pt

Abstract. This paper, presents ongoing work that extends MetaProbLog with Most Probable Explanation (MPE) inference method. The MPE inference method is widely used in Hidden Markov Models in order to derive the most likely states of a model. Recently, we started developing an application that uses MetaProbLog to models phonocardiograms. We target to use this application in order to diagnose heart diseases by using phonocardiogram classification. Motivated by the importance of phonocardiogram classification, we started the implementation of the MPE inference method and an improvement of representation for annotated disjunctions.

1 Introduction

MetaProbLog[1] is a framework of the ProbLog [4,6] 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 [12]. MetaProbLog first appeared in [10] where the focus was to solve meta calls for ProbLog.

The goal of this paper is to present ongoing work which extends MetaProbLog with MPE inference. In Hidden Markov Models (HMM) this inference task is known as the most likely sequence of hidden states and it is been solved with the Viterbi algorithm [15]. Similarly, in Bayesian Networks this inference task is called Maximum a posteriori (MAP) and is usually been estimated either by Monte Carlo approximation or by an expectation maximization (EM) algorithm.

ProbLog semantics allow the description of different type of models, including HMM, Bayesian Networks, and others. Solving this inference task for ProbLog programs, boils down to finding the most probable witness of satisfiability. The original implementation of ProbLog[2] used a greedy search with pruning to find the MPE [7], this algorithm was called MAX. While for the initial set of ProbLog features the MAX algorithm was sufficient, newly added features have increased

[1] https://www.dcc.fc.up.pt/metaproblog/tiki-index.php?page=HomePage
[2] https://dtai.cs.kuleuven.be/problog/problog1/problog1.html

the challenge for finding the MPE. ProbLog 2[3] [5] computes the MPE by first, collecting all explanations as an AND-OR tree then by handling the introduced cycles and finally traversing the tree to find the MPE. This approach currently is the most complete and sound approach but does not take advantage of any pruning strategy.

These new features are crucial for the implementation of an application that classifies phonocardiogram (PCG) signals. We intent to use MetaProbLog in order to implement an application that identifies the most likely characterizations of PCG signals. These characterized PCG signals can then be used in classification to diagnose heart diseases.

2 A Motivating Application

Lately, the classification of 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.

Several factors may complicate S1 (first heart sound) and S2 (second heart sound) detection, such as variability in heart rhythm, dynamic background noise, anatomical variations, artifacts, murmurs, respiratory sounds interference, clicks, and extra-sounds such as S3 and S4 sounds. These factors in combination, result in a low signal-to-noise ratio. A first fundamental step for the PCG analysis is to segment the signal into periods. Several algorithms were successfully implemented in heart sound segmentation problem, such as S-transform [11], recognition system based on Neural Networks [17] and Wavelet Decomposition [2].

Most of the heart sounds used in heart sound segmentation are primarily recorded with specialized equipment and in a controlled environment, ensuring a very high signal-to-noise ratio. However, routine sounds that are obtained with handheld stethoscopes in clinical environments (such as screening campaigns like Caravana do Coracão[4] from where we possess data) have a low signal-to-noise ratio. The correct identification of heart sounds is crucial for the analysis of these signals in more detail.

Recently, HMMs have being used for modeling and characterizing real-word signals such as heart sound signals [13]. 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. Driven by this real life problem we have set new feature requirements for MetaProbLog's implementation.

[3] `https://dtai.cs.kuleuven.be/problog/`

[4] `https://www.circulodocoracao.com.br/sites/caravanadocoracao/en`

3 ProbLog Semantics

A ProbLog program T [4,6] 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.

Definition 1. *ProbLog Program: 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.

The MPE of T for a query q, is the most probable set $L_{MPE} \subseteq L_T$ of probabilistic facts (pf) or their negation contained at a randomly sampled subprogram d of T that entail q. The probability of the MPE is the product of the probabilities ($P(pf)$) of each probabilistic fact contained in L_{MPE} or $1.0 - P(pf)$ for contained probabilistic facts that are negated.

$$P(L_{MPE}) = \prod_{pf_i \in L_{true}} P(pf_i) \cdot \prod_{pf_j \in L_{false}} (1.0 - P(pf_j)) \tag{1}$$

where $L_{true} \cap L_{false} = L_{MPE}$. Finally, L_{MPE} is the set where:

$$argmax_{(L_{MPE} \in H_{interpretations})} P(L_{MPE}). \tag{2}$$

Because of the one-to-one mapping of explanations with Herbrand interpretations, the MPE is also the most likely Herbrand interpretation of the program T for the query q.

3.1 New Challenges

Originally, ProbLog did not supported general negation but only negated probabilistic facts. General negation was introduced by [8]. General negation introduces a new challenge for calculating the most probable witness of satisfiability.

The inference task for a negated subgoal is converted into finding the most probable witness of unsatisfiability.

A second challenge, originates in the introduction of Annotated Disjunctions (ADs) for ProbLog. ADs in ProbLog are compiled, by using a program transformation technique, to a set of probabilistic facts that through negation form a mutually exclusive structure. While this modification, does not affect most inference tasks, it makes the MPE humanly unreadable by returning the compiled probabilistic facts and not the ADs.

Finally, the addition of evidence in the new implementations of ProbLog impose a new challenge for computing the MPE. Evidence can be of an important use for our application as in some cases we might have a priori knowledge of some PCG signal characterizations. As shown and tackled in [14] ADs introduce a further complication for computing the MPE together with evidence.

3.2 MetaProbLog

MetaProbLog is an implementation of the ProbLog semantics within Yap Prolog [3]. In addition, MetaProbLog extends the semantics of ProbLog by defining a "ProbLog engine" which permits the definitions of probabilistic meta calls as presented in [10]. The "ProbLog engine" approach permits a more elegant handling of general negation for ProbLog, the use of any inference approaches as subgoals and the use of probabilistic meta calls. MetaProbLog inference, currently allows the computation of marginal probabilities with or without evidence and implements two inference methods **exact** and **program sampling**. Furthermore, it allows the computation of marginal probabilities for the answers of non-ground queries through the use of a special meta inference task called **ProbLog answers**. MetaProbLog has two unique features as a ProbLog implementation: probabilistic meta calls and datasets [9].

4 MetaProbLog's MPE for General ProbLog Programs and Future Support

While this work is still ongoing, we do have a preliminary implementation of ProbLog MPE inference that supports general negation. The original ProbLog implementation based its MPE inference on the fact that derivations are monotonic and that every added probabilistic fact will decrease the probability of the explanation. Because of that the original ProbLog MPE implementation was able to prune a big part of the search space.

Adding negated probabilistic facts (which the original ProbLog implementation supports) does not alter the MPE algorithm as the derivations remain monotonous decreasing. On the other hand general negation creates a significant complication. While the probability calculation remains monotonously decreasing and from the logic programming point of view the task remains the

same (find the most probable Herbrand interpretation), the search tree[5] traversed alters significantly. From a SAT point of view, general negation, converts the task in finding the most probable witness of unsastisfiability which is a hard problem [16].

For ProbLog programs without negation the search tree is composed by disjoint branches which are composed by conjoint literals (probabilistic facts). When general negation is applied on a subtree then by De Morgan's law the disjunctions are converted to conjunctions, the conjunctions to disjunctions and the literals become negated. In order to address this change on the search tree we are forced to collect all the negated goal's subtree and convert it by De Morgan's law. This however, delays the usage of the pruning mechanism which in some cases results to a computational overhead. Fortunately, this approach though does not increase the overall complexity of the inference algorithm. Furthermore, this mechanism is only activated for negated subgoals and not for the rest parts of the search tree.

While, currently we have already a first implementation of the described approach the next features are part of our work in progress.

4.1 Annotated Disjunction Representation

The existing representation for ADs creates a linear expansion of probabilistic facts for each AD value. This representation imposes two problems: first, the MPE returned to the user does not use the representation of the AD values instead it uses the linear expansion of probabilistic facts; second as shown in [14] the current representation computes wrongly the MPE in presence of evidence. We are working on a novel approach which uses the AD values directly for MPE and the linear representation only for other inference tasks solving both problems in a different way than the one presented in [14].

4.2 ProbLog Queries with Evidence

Furthermore, we need to address the conditional MPE task. In order to address this task we require a second search tree that provides the "evidence explanations (EEs)". Then we need to combine the candidate MPEs with the EEs in order to find the combination that maximizes the conditioned probability (for every probabilistic fact that is contained in EE, the probability of that fact does not contribute for the probability of the candidate explanation). Fortunately, the combinations can be pruned significantly.

4.3 Cycles and Tabling

Finally, in order for the algorithm and the implementation to fully support general ProbLog programs we need to be able to compute the MPE task in the presence of cycles. While this task is not needed for computing the MPE in

[5] For our task the search tree is the SLD tree used to prove the goal.

HMMs neither needed for our motivating application, it is necessary for the completeness of the resulting algorithm and implementation.

Cyclic programs in MetaProbLog are only allowed/handled with the conjuction of tabling (similarly as in Prolog). Handling this type of cycles has been already solved for the exact inference method in [8]. However, the combination of tabling and pruning is tricky imposing a different challenge. Possibly, MPE for cyclic programs will not be able to use pruning in the collection of explanations and only at a second stage, similarly to when handling general negation. ProbLog 2 MPE algorithm uses a similar strategy.

5 Conclusion

We presented the added challenges for calculating the MPE for general ProbLog programs in MetaProbLog. The added challenges compared with the original ProbLog are imposed by the new features of the new ProbLog implementations. Furthermore, we outlined how we intent to tackle these challenges. We require these features in order to apply them in our recent motivating application of computing the most probable characterization of PCG signals.

As we presented PCG signal characterization is an important motivating task that can aid in the diagnosis of heart diseases. We have already modeled the general behaviour of PCG signals in ProbLog as a HMM. We intent to grow our model to input real PCG signals and by using specific features of the signals to extract possible characterizations that later are been used as evidence in MPE inference in order to characterize the remaining signal features.

Acknowledgments: We want to thank the anonymous reviewers for their comments and help to improve our paper. Theofrastos Mantadelis is funded by the Portuguese Foundation for Science and Technology (FCT) within the project UID/EEA/50014/2013. Jorge Oliveira is funded by the Portuguese Foundation for Science and Technology (FCT) within the project HeartSafe, PTDC/EEI-PRO/2857/2012.

References

1. Bentley, P., Nordehn, G., Coimbra, M., Mannor, S.: The PASCAL classifying heart sounds challenge 2011 (CHSC2011) results., `http://www.peterjbentley.com/heartchallenge/`
2. Castro, A., Vinhoza, T., Mattos, S., Coimbra, M.: Heart sound segmentation of pediatric auscultations using wavelet analysis. In: Engineering in Medicine and Biology Society (EMBC). pp. 3909–3912 (July 2013)
3. Costa, V.S., Rocha, R., Damas, L.: The YAP prolog system. TPLP 12(1-2), 5–34 (2012)
4. De Raedt, L., Kimmig, A., Toivonen, H.: ProbLog: A probabilistic Prolog and its application in link discovery. In: IJCAI. pp. 2468–2473 (2007)

5. Fierens, D., Van den Broeck, G., Renkens, J., Shterionov, D., Gutmann, B., Thon, I., Janssens, G., De Raedt, L.: Inference and learning in probabilistic logic programs using weighted Boolean formulas. Theory and Practice of Logic Programming 15(03), 358–401 (May 2015), https://lirias.kuleuven.be/handle/123456789/392821
6. Kimmig, A., Demoen, B., Raedt, L.D., Costa, V.S., Rocha, R.: On the implementation of the probabilistic logic programming language ProbLog. TPLP 11, 235–262 (2011)
7. Kimmig, A., De Raedt, L., Toivonen, H.: Probabilistic explanation based learning. In: European Conference on Machine Learning (ECML), vol. 4701, pp. 176–187 (2007)
8. Mantadelis, T., Janssens, G.: Dedicated tabling for a probabilistic setting. In: Hermenegildo, M.V., Schaub, T. (eds.) International Conference on Logic Programming (ICLP Technical Communications). LIPIcs, vol. 7, pp. 124–133. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2010)
9. Mantadelis, T., Janssens, G.: MetaProbLog. ALP Newsletter (April 2015), http://www.cs.nmsu.edu/ALP/2015/04/metaproblog/
10. Mantadelis, T., Janssens, G.: Nesting probabilistic inference. CoRR abs/1112.3785 (2011), http://arxiv.org/abs/1112.3785
11. Moukadem, A., Dieterlen, A., Hueber, N., Brandt, C.: A robust heart sounds segmentation module based on s-transform. Biomedical Signal Processing and Control 8(3), 273 – 281 (2013)
12. Sato, T.: A statistical learning method for logic programs with distribution semantics. In: International Conference on Logic Programming (ICLP). pp. 715–729. MIT Press (1995)
13. Sedighian, P., Subudhi, A., Scalzo, F., Asgari, S.: Pediatric heart sound segmentation using hidden markov model. In: Engineering in Medicine and Biology Society (EMBC). pp. 5490–5493 (Aug 2014)
14. Shterionov, D., Renkens, J., Vlasselaer, J., Kimmig, A., Meert, W., Janssens, G.: The most probable explanation for probabilistic logic programs with annotated disjunctions. In: International Conference on Inductive Logic Programming, Nancy, France, 14-16 September 2014 (2014), https://lirias.kuleuven.be/handle/123456789/474713
15. Viterbi, A.: Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. Information Theory 13(2), 260–269 (April 1967)
16. Wu, L., Zhou, H., Alava, M., Aurell, E., Orponen, P.: Witness of unsatisfiability for a random 3-satisfiability formula. CoRR abs/1303.2413 (2013), http://arxiv.org/abs/1303.2413
17. Ölmez, T., Dokur, Z.: Classification of heart sounds using an artificial neural network. Pattern Recognition Letters 24(1–3), 617 – 629 (2003)