International Journal of Approximate Reasoning 78 (2016) 223–240

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

www.elsevier.com/locate/ijar

Probabilistic abductive logic programming using Dirichlet priors ☆

Calin Rares Turliuc [a,*], Luke Dickens [b], Alessandra Russo [a], Krysia Broda [a]

[a] Department of Computing, Imperial College London, United Kingdom
[b] Department of Information Studies, University College London, United Kingdom

ARTICLE INFO

Article history:
Received 22 January 2016
Received in revised form 4 June 2016
Accepted 1 July 2016
Available online 15 July 2016

Keywords:
Probabilistic programming
Abductive logic programming
Markov Chain Monte Carlo
Latent Dirichlet allocation
Repeated insertion model

ABSTRACT

Probabilistic programming is an area of research that aims to develop general inference algorithms for probabilistic models expressed as probabilistic programs whose execution corresponds to inferring the parameters of those models. In this paper, we introduce a probabilistic programming language (PPL) based on abductive logic programming for performing inference in probabilistic models involving categorical distributions with Dirichlet priors. We encode these models as abductive logic programs enriched with probabilistic definitions and queries, and show how to execute and compile them to boolean formulas. Using the latter, we perform generalized inference using one of two proposed Markov Chain Monte Carlo (MCMC) sampling algorithms: an adaptation of uncollapsed Gibbs sampling from related work and a novel collapsed Gibbs sampling (CGS). We show that CGS converges faster than the uncollapsed version on a latent Dirichlet allocation (LDA) task using synthetic data. On similar data, we compare our PPL with LDA-specific algorithms and other PPLs. We find that all methods, except one, perform similarly and that the more expressive the PPL, the slower it is. We illustrate applications of our PPL on real data in two variants of LDA models (Seed and Cluster LDA), and in the repeated insertion model (RIM). In the latter, our PPL yields similar conclusions to inference with EM for Mallows models.

1. Introduction

Probabilistic programming is an area of research that aims to develop general inference algorithms for probabilistic models expressed as probabilistic programs whose execution corresponds to inferring the parameters of the probabilistic model. A wide range of probabilistic programming languages (PPLs) have been developed to express a variety of classes of probabilistic models. Examples of PPLs include Church [1], Anglican [2], BUGS [3], Stan [4] and Figaro [5].[1] Some PPLs, such as Church, enrich a functional programming language with exchangeable random primitives, and can typically express a wide range of probabilistic models. However, inference is not always tractable in these expressive languages. Other PPLs are logic-based. They typically add probabilistic annotations or primitives to a logical encoding of the model. This encoding usually relates either to first-order logic, e.g. Alchemy [6], BLOG [7] or to logic programming, e.g. PRiSM [8], ProbLog [9]. Most

☆ This paper is part of the virtual special issue on Probabilistic logic programming 2015, edited by J. Vennekens and F. Riguzzi.

* Corresponding author.
E-mail address: ct1810@imperial.ac.uk (C.R. Turliuc).

[1] For a more comprehensive list cf. http://probabilistic-programming.org/.

http://dx.doi.org/10.1016/j.ijar.2016.07.001

logic-based PPLs focus only on discrete models, and consequently are equipped with more specialized inference algorithms, with the advantage of making the inference more tractable.

However, logic-based PPLs generally do not consider Bayesian inference with prior distributions. For instance, Alchemy implements Markov logic, encoding a first-order knowledge base into Markov random fields. Uncertainty is expressed by weights on the logical formulas, but it is not possible to specify prior distributions on these weights. ProbLog is a PPL that primarily targets the inference of conditional probabilities and the most probable explanation (maximum likelihood solution); it does not feature the specification of prior distributions on categorical distributions. PRiSM is a PPL which introduces Dirichlet priors over categorical distributions and is deigned for efficient inference in models with non-overlapping explanations.

This paper contributes to the field of logic-based PPL by proposing an alternative approach to probabilistic programming. Specifically, we introduce a PPL based on logic programming for performing inference in probabilistic models involving categorical distributions with Dirichlet priors. We encode these models as abductive logic programs [10] enriched with probabilistic definitions and inference queries, such that the result of abduction allows overlapping explanations. We propose two Markov Chain Monte Carlo (MCMC) sampling algorithms for the PPL: an adaptation of the uncollapsed Gibbs sampling algorithm, described in [11], and a newly developed collapsed Gibbs sampler. Our PPL is similar to PRiSM and different from ProbLog in that it can specify Dirichlet priors. Unlike PRiSM, but similarly to ProbLog, we allow overlapping explanations. However, in this paper, all the models we study have non-overlapping explanations.

We show how our PPL can be used to perform inference in two classes of probabilistic models: Latent Dirichlet Allocation (LDA, [12]), a well studied approach for topic modeling, including two variations thereof (Seed LDA and Cluster LDA); and the repeated insertion model (RIM, [13]), a model used for preference modeling and whose generative story can be expressed using recursion. Our experiments demonstrate that our PPL can express a broad class of models in a compact way and scale up to medium size real-data sets, such as LDA with approximately 5000 documents and 100 words per document. On synthetic LDA data, we compare our PPL with two LDA-specific algorithms: collapsed Gibbs sampling (CGS) and variational expectation maximization (VEM), and two state-of-the-art PPLs: Stan and PRiSM. We find that all methods, with the exception of the chosen VEM implementation, perform similarly, and that the more expressive the method, the slower it is, in the following order, with the exception of VEM: CGS (fastest), PRiSM, VEM, our PPL, Stan (slowest).

The paper is organized as follows. Section 2 presents the class of probabilistic models supported by our PPL. In Section 3 we outline the syntax and the semantics of the PPL, whereas our two Gibbs sampling algorithms are discussed in Section 4. Section 5 shows our experimental results and Section 6 relates our PPL to other PPLs and methods. Finally, Section 7 concludes the paper.

2. The probabilistic model

This section begins with the description of a particular approach to probabilistic programming. Then we introduce `peircebayes`[2] (PB), our probabilistic abductive logic programming language designed to perform inference in discrete models with Dirichlet priors. Throughout the paper we will use normal font for scalars (α), arrow notation for vectors ($\vec{\alpha}$), and bold font for collections with multiple indexes ($\boldsymbol{\alpha}$), e.g. sets of vectors.

Probabilistic programs, as defined in [14], are "'usual' programs with two added constructs: (1) the ability to draw values at random from distributions, and (2) the ability to condition values of variables in a program via observe statements". We can instantiate this general definition by considering the notions of hyperparameters $\boldsymbol{\alpha}$, parameters $\boldsymbol{\theta}$, and observed variables $\vec{f}$ and assuming the goal to be the characterization of the conditional distribution $P(\boldsymbol{\theta}|\vec{f};\boldsymbol{\alpha})$.

Most PPLs assume such a conditional with a continuous sample space, i.e. they allow, in principle, probabilistic programs with an uncountably infinite number of unique outputs, should one not take into account issues of real number representations. In our approach, the conditional sample space is assumed to be finite, i.e. one can enumerate all possible executions. Specifically, in our PPL we restrict the class of distributions from which we can draw to categorical distributions with Dirichlet priors. The Dirichlet priors are chosen for their conjugacy, which supports efficient marginalization.

The generality of our PPL is not given by the range of probability distributions that we can draw from, but rather by the way the draws from categorical distributions interact in the "generative story" of the model. We choose our "usual programs" to be abductive logic programs enriched with probabilistic primitives. Similarly to Church, Anglican and other PPLs this is declarative programming language, but one in which the generative story is expressed as an abductive reasoning task responsible for identifying relevant draws from the categorical distributions given the observations. Our choice is motivated by the significant amount of related work in probabilistic logic programming, although both functional and logic programming are Turing complete, so they are equally general. In what follows we present the class of probabilistic models that are supported by our PPL. We define them first as uncollapsed models, then show how they can be collapsed. As demonstrated in Section 4, this dual formalization leads naturally to the possibility of using in our PPL uncollapsed as well as collapsed MCMC sampling methods.

[2] Named, in Church style, after Charles Sanders Peirce, the father of logical abduction, and Thomas Bayes, the father of Bayesian reasoning. Pronounced ['pɜrs'beɪz].