International Journal of Approximate Reasoning 67 (2015) 73–110

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

www.elsevier.com/locate/ijar

Learning failure-free PRISM programs

Waleed Alsanie[a,*], James Cussens[b]

[a] *The National Center for Computation Technology and Applied Mathematics, Communication and Information Technology Research Institute, King Abdulaziz City for Science and Technology, Riyadh, Saudi Arabia*
[b] *Department of Computer Science, University of York, York, UK*

ARTICLE INFO

Article history:
Received 15 December 2014
Received in revised form 14 May 2015
Accepted 8 June 2015
Available online 11 June 2015

Keywords:
Probabilistic inductive logic programming
Probabilistic logic
PRISM
Structure learning

ABSTRACT

PRISM is a probabilistic logic programming formalism which allows defining a probability distribution over possible worlds. This paper investigates learning a class of generative PRISM programs known as failure-free. The aim is to learn recursive PRISM programs which can be used to model stochastic processes. These programs generalise dynamic Bayesian networks by defining a halting distribution over the generative process. Dynamic Bayesian networks model infinite stochastic processes. Sampling from infinite process can only be done by specifying the length of sequences that the process generates. In this case, only observations of a fixed length of sequences can be obtained. On the other hand, the recursive PRISM programs considered in this paper are self-terminating upon some halting conditions. Thus, they generate observations of different lengths of sequences. The direction taken by this paper is to combine ideas from inductive logic programming and learning Bayesian networks to learn PRISM programs. It builds upon the inductive logic programming approach of learning from entailment.

1. Introduction

Dynamic Bayesian networks are models which represent infinite stochastic processes [30]. In order to generate observations of an infinite dynamic process, a fixed length of sequences needs to be specified, and observations of this length are generated. However, observations of different lengths of sequences, possibly random, cannot be obtained.

PRISM is a probabilistic logic programming formalism built upon the concept of *distribution semantics* [37]. A PRISM program is a probabilistic Turing machine [37,38], and thus accepts a *recursively enumerable language* [21].

This paper concerns learning a class of PRISM programs known as *failure-free* which generalise *dynamic Bayesian networks* by defining a halting distribution over the generative process. By defining a halting distribution, the model allows observations consisting of sequences of different lengths to be sampled. The sampling process is self-terminating based upon some probabilistic condition. This is an additional modelling property over those in *dynamic Bayesian networks*. The importance of modelling a self-terminating process is that it represents a *stochastic grammar* which models a *recursive language* [21]; therefore, these processes correspond to stochastic automata which accept these languages. In this paper, we focus on tail recursive PRISM programs which encode dependencies amongst words in the strings of a language.

Different formalisms have been proposed to represent dependencies amongst words in the strings generated by grammars, some of which are *probabilistic feature grammar* [18], *probabilistic context-free grammar with latent annotation* [25] and

[*] Corresponding author.
E-mail addresses: walsanie@kacst.edu.sa (W. Alsanie), james.cussens@york.ac.uk (J. Cussens).

http://dx.doi.org/10.1016/j.ijar.2015.06.003
0888-613X/© 2015 Elsevier Inc. All rights reserved.

Bayesian network automata [20]. These formalisms simplify modelling stochastic grammars, *regular* or *context-free*, where the probability distribution of the derivation rules of some non-terminals depends on the derivation rules that have taken place for other non-terminals. For example, in natural language processing, a probability distribution can be defined over the derivation of non-terminals according to some semantic roles. Consider the following grammar modelling a simple natural language clause:

$$\begin{array}{lcl} S & \longrightarrow & A \text{ “is expected to” } V \text{ “again”} \\ A & \longrightarrow & \text{“someone”} \\ A & \longrightarrow & \text{“the sun”} \\ V & \longrightarrow & \text{“shine”} \\ V & \longrightarrow & \text{‘come”} \end{array} \tag{1}$$

where S is a starting non-terminal symbol and A and V are non-terminals. One might want to model the clause such that the distribution of V yielding ”shine“ or ”come“ depends on whether A has yielded ”someone“ or ”the sun“. The distribution then can be defined as follows:

$$\begin{array}{rcl} S & \xrightarrow{\qquad\qquad} & A \text{ “is expected to” } V \text{ “again”} \\ A & \xrightarrow{P(A=\text{“someone”})} & \text{“someone”} \\ A & \xrightarrow{P(A=\text{“the sun”})} & \text{“the sun”} \\ \text{“someone is expected to” } V & \xrightarrow{P(V=\text{“shine”}|A=\text{“someone”})} & \text{“someone is expected to shine”} \\ \text{“the sun is expected to” } V & \xrightarrow{P(V=\text{“shine”}|A=\text{“the sun”})} & \text{“the sun is expected to shine”} \\ \text{“someone is expected to” } V & \xrightarrow{P(V=\text{“come”}|A=\text{“someone”})} & \text{“someone is expected to come”} \\ \text{“the sun is expected to” } V & \xrightarrow{P(V=\text{“come”}|A=\text{“the sun”})} & \text{“the sun is expected to come”} \end{array} \tag{2}$$

The grammar above represents a static process. By modifying it to have the following additional 3 rules:

$$\begin{array}{lcl} S_0 & \longrightarrow & S\ H. \\ . & & \\ . & & \\ H & \xrightarrow{P(H=\text{“,”})} & \text{“,” } S \\ H & \xrightarrow{P(H=\text{“.”})} & \text{“.”} \end{array} \tag{3}$$

such that S_0 is the starting non-terminal symbol instead of S, and H is non-terminal, we obtain a grammar representing a compound sentence whose main clauses are of the kind modelled by the grammar in (2), and whose termination is marked by a full stop. The length of the compound sentence is undetermined and it is defined by the distribution $P(H)$. The distribution of the derivation of H can also be defined to be dependent on other derivations.

The importance of temporal models with a halting distribution rises in modelling and analysing sequences up to the occurrence of some event. Besides modelling sentences in natural languages, in monitoring and diagnosis, one might be interested in a model which represents the states of a system until some system condition is met, e.g. a fault or an abnormal event [1,27].

Let **N** be a set of non-terminals and Σ be a set of terminals. Let S, B and $\forall i : 1 \le i \le n$ and $\forall v : 1 \le v \le n'$, A_i and C_v be non-terminals in **N** such that S is a starting non-terminal symbol. Let a_i, c_v and b be terminals in Σ. Then, the PRISM programs considered in this paper represent either one of the two following stochastic grammars:

$$\begin{array}{rcl} S & \xrightarrow{\qquad\qquad} & A_1 \ldots A_n B \\ B & \xrightarrow{\qquad\qquad} & C_1 \ldots C_{n'} B \\ A_1 & \xrightarrow{P(A_1)} & a_1^1 | \ldots | a_1^j \\ a_1 \ldots a_{i-1} A_i & \xrightarrow{P(A_i|a_1 \ldots a_{i-1})} & a_1 \ldots a_{i-1} a_i^1 | \ldots | a_1 \ldots a_{i-1} a_i^m \\ & . & \\ & . & \\ & . & \\ a_1 \ldots a_n C_1 & \xrightarrow{P(C_1|a_1 \ldots a_n)} & a_1 \ldots a_n c_1^1 | \ldots | a_1 \ldots a_n c_1^{j'} \\ a_1 \ldots a_n c_1 \ldots c_{v-1} C_v & \xrightarrow{P(C_v|a_1 \ldots a_n c_1 \ldots c_{v-1})} & a_1 \ldots a_n c_1 \ldots c_{v-1} c_v^1 | \ldots | a_1 \ldots a_n c_1 \ldots c_{v-1} c_v^{m'} \\ a_1 \ldots a_n c_1 \ldots c_{n'} B & \xrightarrow{P(B|a_1 \ldots a_n c_1 \ldots c_{n'})} & a_1 \ldots a_n c_1 \ldots c_{n'} b_0 | a_1 \ldots a_n c_1 \ldots c_{n'} b_1 B | \ldots | \\ & & a_1 \ldots a_n c_1 \ldots c_{n'} b_k B \end{array} \tag{4}$$