PEPL: An implementation of FAM for SLPs

Jianzhong Chen[♭] Stephen Muggleton[♭]
James Cussens[†] Nicos Angelopoulos[‡]
[♭] Department of Computing, Imperial College, London
[†]University of York, UK [‡]Netherlands Cancer Institute

*P*epl, parameter estimation in Prolog, is an implementation of the failure adjusted maximisation algorithm (FAM) [4] for Stochastic Logic Programs (Slps) [6, 7]. *S*LPs extend logic programming by arithmetic labels on clausal definitions. They have well characterised log linear semantics and backtracking strategies [5, 3]. The FAM algorithm was introduced as an extension to the Expectation-Maximization (EM) algorithm and account for failed derivation paths in SLPs [4]. It provides a closed-form for computing the parameter weights within EM's iterative maximization approach. The algorithm has been shown to work for normalised *S*LPs, [4], and is in practice applicable to a wide class of programs. The failure adjusted aspect of the algorithm has also been incorporated in the PRISM system [8].

*P*epl is implemented in Prolog and is available as open source. Stochastic clauses are term expanded to standard Prolog ones. Unique identifiers and a path argument are added to the transformation of stochastic clauses. These are used to identify the path of each derivation. In addition failure paths are also recorded by term expansion techniques. The system provides three ways for computing the counts needed for the closed-form calculation: exact, sample and store. The first method is the straight forward approach where all solutions to the target goal are quarried at each iterative step. Sampling approximates the counts by only sampling from the target. The expressions associated with the exact computation can be stored as term structures of arithmetic expression that can be evaluated at each iteration with fresh instantiations of the labels. This trades space for speed, making the computation much faster by requiring larger amounts of memory. A number of examples are provided with the distribution. These include the blood type example from PRISM, a stochastic context free grammar and the worked example from [4]. *P*epl runs on the current Yap (6.2.0) and Swi (5.10.3/5.11.22) Prologs.

*P*epl have been well applied to a multi-class protein fold prediction problem [1], in which SLP structure has been learned by ILP system Progol and SLP parameters have been estimated using *P*epl . On the basis of several experiments, it was demonstrated that SLPs and *P*epl have advantages for solving multi-class prediction problems with the learned probabilities. The experiment results can be found at `http://www.doc.ic.ac.uk/~cjz/ProteinSLPs/`.

SLPs and *P*epl have also been applied in a framework of abductive SLPs [2],

which provides possible worlds semantics to SLPs through abduction. Examples with probability labels are introduced within a standard scientific experimental setting involving control and treated data. FAM and *P*epl are used to learn SLPs from the probabilistic examples. The results demonstrate that the probabilistic models learned from probabilistic examples lead to a significant decrease in error accompanied by improved insight from the learned results compared with the models learned from non-probabilistic examples. The experiment materials can be found at `http://www.doc.ic.ac.uk/~cjz/AbductiveSLPs/`. We also demonstrate that the parameter estimation results from PRISM also hold for *P*epl in the application.

Availability of *P*epl : `http://scibsfs.bch.ed.ac.uk/~nicos/sware/slps/pe`

References

[1] J. Chen, L. Kelley, S.H. Muggleton, and M. Sternberg. Protein fold discovery using Stochastic Logic Programs. In *Probabilistic ILP*, pages 244–262. 2007.

[2] J. Chen, S.H. Muggleton, and J. Santos. Learning probabilistic logic models from probabilistic examples. *Machine Learning*, 73(1):55–85, 2008.

[3] J. Cussens. Stochastic logic programs: Sampling, inference and applications. In *UAI'2000*, pages 115–122, 2000.

[4] J. Cussens. Parameter estimation in stochastic logic programs. *Machine Learning*, 44(3):245–271, 2001.

[5] James Cussens. Loglinear models for first-order probabilistic reasoning. In *UAI-99*, 1999.

[6] Stephen Muggleton. Stochastic logic programs. In L. de Raedt, editor, *Advances in Inductinve Logic Programming*, pages 254–264. IOS Press, 1996.

[7] Stephen Muggleton. Semantics and derivations for SLPs. In *Workshop on Fussion of Domain Knowledge with Data for Decision Support*, 2000.

[8] T Sato, Y. Kameya, and N.-F. Zhou. Generative modeling with failure in PRISM. In *IJCAI'2005*, page 847852, 2005.