Probabilistic Inductive Constraint Logic

Fabrizio Riguzzi[1], Elena Bellodi[2], Riccardo Zese[2], Marco Alberti[1], Evelina Lamma[2]

Machine Learning 110(4): 723-754 (2021)

[1]Department of Mathematics and Computer Science and [2]Department of Engineering, University of Ferrara, Italy

Probabilistic Constraint Logic Theories

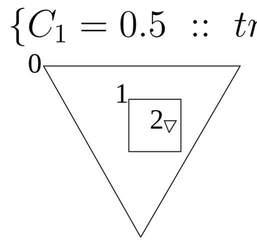
Inductive Constraint Logic (De Raedt and Van Laer): learning Constraint Logic Theories (sets of integrity constraints) from interpretations rather than from entailment.
In this work: *Probabilistic* Constraint Logic Theories (PCLTs) and a system (PASCAL) to perform discriminative learning of their structure and parameters from interpretations.

Language A PCLT is a set of probabilistic integrity constraints (ICs) of the form $p_i \ :: \ L_1, \ldots, L_b \rightarrow \exists(P_1); \ldots; \exists(P_n); \forall\neg(N_1); \ldots; \forall\neg(N_m)$

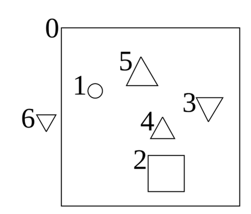
Semantics A PCLT T defines a probability distribution on the set W of ground constraint logic theories called *possible theories*; for each grounding of the body of each IC, we include the IC in a possible theory with probability p_i and we assume all groundings to be independent.
An IC's probability is the sum of the probabilities of the possible theories where a grounding of the constraint is present.
In each possible theory, An IC C is *true in an interpretation* I ($I \models C$) if and only if, for each substitution θ such that each literal in $Body(C)\theta$ is ground and true in I, at least one disjunct in $Head(C)\theta$ is true in I.
The probability $P(\oplus|I)$ of the positive class given an interpretation I, the probability of a PCLT T satisfying I, is

$$P(\oplus|I) = \sum_{w\in W} P(\oplus, w|I) = \sum_{w \in W} P(\oplus|w, I)P(w|I) = \sum_{w\in W, M(\mathbf{BG}\cup I)\models w} P(w)$$

Example (from Bongard)
Background knowledge:
$in(A, B) \leftarrow inside(A, B)$
$in(A, D) \leftarrow inside(A, C), in(C, D)$
PCLT:
$\{C_1 = 0.5 \ :: \ triangle(T), square(S), in(T, S) \rightarrow false\}$



The body of C_1 is true for the single substitution $T/2$ and $S/1$ thus $m_1 = 1$ and $P(\oplus|I) = 0.5$.



The body of C_1 is true for three pairs (triangle, square) thus $m_1 = 3$ and $P(\oplus|I) = 0.125$.

PCLT parameter learning

Problem Given
- a PCLT theory T
- a set $\mathcal{I}^+ = \{I_1, \ldots, I_Q\}$ of positive interpretations
- a set $\mathcal{I}^- = \{I_{Q+1}, \ldots, I_R\}$ of negative interpretations
- a normal logic program **BG** (background knowledge)

find the parameters of T such that the likelihood

$$L = \prod_{q=1}^{Q} P(\oplus|I_q) \prod_{r=Q+1}^{R} P(\ominus|I_r)$$

is maximized. The likelihood is given by the probability that the example labels are observed for each example.
The equation $\frac{\partial L}{\partial p_i} = 0$ does not admit a closed form solution, so we must use optimization to find the maximum of L.
We can optimize the likelihood with gradient descent, where weights are updated using the formula

$$\mathbf{p}_{n+1} = \mathbf{p}_n - \epsilon\nabla_{\mathbf{p}} L(\mathbf{p}) = \mathbf{p}_n - \epsilon\nabla_{\mathbf{p}} \frac{\partial L}{\partial \mathbf{p}}$$

where ϵ is the learning rate defining the size of the step done by gradient descent along the gradient and $\mathbf{p}$ is the vector containing the parameters p_i, or with a second order method such as Limited-memory BFGS (L-BFGS).
Experiments show that gradient descent outperforms L-BFGS in most cases in terms of area under the PR and ROC curves, and execution time.

PASCAL: PCLT structure learning

Problem Given
- a set $\mathcal{I}^+ = \{I_1, \ldots, I_Q\}$ of positive interpretations
- a set $\mathcal{I}^- = \{I_{Q+1}, \ldots, I_R\}$ of negative interpretations
- a normal logic program **BG** (background knowledge)
- a language bias

find a PCLT T that maximizes the likelihood

$$L = \prod_{q=1}^{Q} P(\oplus|I_q) \prod_{r=Q+1}^{R} P(\ominus|I_r)$$

PASCAL solves this problem by first identifying good candidate ICs and then searching for a theory guided by the log likelihood (LL) of the data.
Parameters:
- MLB, the maximum number of literals in the body of ICs;
- MD, the maximum number of disjuncts in the head of ICs;
- MLP and MLN, the maximum number of literals allowed in a P disjunct and a N disjunct respectively.

```
function PASCAL(I+, I−, BG, NC, MLB, MD, MLP, MLN, BeamSize, MaxSteps)
    Steps = 1
    Beam ← (false ← true, −∞)                                   ▷ Empty IC
    repeat
        NewBeam = []
        while Beam is not empty do                               ▷ ICs search
            Remove the first IC (C, LL) from Beam
            Ref ← all refinements of C respecting MLB,MD,MLP, and MLN
            for all C′ ∈ Ref do
                ({C″}, LL″) ←LearnParams({C′}, I+, I−, BG)       ▷ gradient descent
                NewBeam ←Insert((C″, LL″), NewBeam)
                if size(NewBeam) > BeamSize then
                    Remove the last element of NewBeam
                end if
            end for
        end while
        Beam ← NewBeam
        Steps = Steps + 1
    until Steps > MaxSteps
    T ← ∅, LL ← −∞                                               ▷ Theory search
    repeat
        Remove the first couple (C, LL) from Beam
        (T′, LL′) ←LearnParams(T ∪ {C}, I+, I−, BG)
        if LL′ > LL then
            T ← T′, LL ← LL′
        end if
    until Beam is empty or T contains NC ICs
    return T
end function
```

Experimental results

We compared PASCAL with the PLP algorithms LIFTCOVER, SLIPCOVER and LEMUR, the MLNs algorithms MLN-BC/MLN-BT, the (probabilistic) relational classifier TILDE.

Average AUC-PR

Dataset	SLIPCOVER	LIFT-EM	LIFT-LBFGS	PASCAL	TILDE
Bupa	**1**	**1**	**1**	**1**	0.420
Carcinogenesis	0.745	0.672	0.561	**0.770**	0.707
Financial	0.173	0.126	0.187	**0.317**	0.123
Mondial	**0.776**	0.763	0.723	0.652	0.650
Mutagen.	0.920	**0.971**	0.725	0.902	0.851
Pyrimidine	0.956	**1**	0.819	0.990	0.769
Sisya	**0.708**	0.706	0.706	0.622	0.621
Sisyb	**0.287**	0.286	0.286	0.286	0.286
Triazine	0.560	0.734	0.760	**0.855**	0.685
Yeast	0.428	0.502	0.448	0.469	**0.588**
Bongard	0.899	0.966	**0.970**	0.635	0.300

Dataset	LEMUR	MLN-BC	MLN-BC samp.	MLN-BT	MLN-BT samp.	PASCAL
Carcinogenesis	0.691	0.619	0.633	0.503	0.494	**0.770**
Mondial	**0.864**	0.585	0.742	0.735	0.781	0.652
Mutagenesis	**0.952**	0.690	0.831	0.872	–	0.902

Average AUC-ROC

Dataset	SLIPCOVER	LIFT-EM	LIFT-LBFGS	PASCAL	TILDE
Bupa	**1**	**1**	**1**	**1**	0.500
Carcinogenesis	0.695	**0.766**	0.472	0.763	0.667
Financial	0.568	0.432	0.535	**0.745**	0.478
Mondial	0.630	**0.663**	0.643	0.495	0.500
Mutagen.	0.826	**0.931**	0.649	0.806	0.778
Pyrimidine	0.925	**1**	0.850	0.993	0.815
Sisya	0.719	0.372	**0.721**	0.502	0.499
Sisyb	**0.500**	**0.500**	**0.500**	**0.500**	**0.500**
Triazine	0.544	0.713	0.760	**0.803**	0.600
Yeast	0.733	0.786	0.721	**0.794**	0.718
Bongard	0.944	0.975	**0.987**	0.749	0.500

Dataset	LEMUR	MLN-BC	MLN-BC samp.	MLN-BT	MLN-BT samp.	PASCAL
Carcinogenesis	0.691	0.619	0.633	0.503	0.494	**0.770**
Mondial	**0.864**	0.585	0.742	0.735	0.781	0.652
Mutagenesis	**0.952**	0.690	0.831	0.872	–	0.902

The three datasets where PASCAL performs well - Triazine, Financial and Carcinogenesis - have a small number of examples but a large number of different predicates, possibly indicating that the expressive language bias is beneficial when the dataset is not very big but has a rich structure.