Probabilistic Logic Learning

Fabrizio Riguzzi

Dipartimento di Matematica e Informatica
Università di Ferrara

joint work with
Elena Bellodi, Giuseppe Cota, Nicola Di Mauro, Evelina Lamma, Riccardo Zese

UNIVERSITÀ
DEGLI STUDI
DI FERRARA
- EX LABORE FRUCTUS -

Outline

Combining Logic and Probability

- Useful to model domains with complex and uncertain relationships among entities
- Many approaches proposed in the areas of Logic Programming, Uncertainty in AI, Machine Learning, Databases
- Distribution Semantics [Sato ICLP95]
- A probabilistic logic program defines a probability distribution over normal logic programs (called instances or possible worlds or simply worlds)
- The distribution is extended to a joint distribution over worlds and interpretations (or queries)
- The probability of a query is obtained from this distribution

Probabilistic Logic Programming (PLP) Languages under the Distribution Semantics

- Probabilistic Logic Programs [Dantsin RCLP91]
- Probabilistic Horn Abduction [Poole NGC93], Independent Choice Logic (ICL) [Poole AI97]
- PRISM [Sato ICLP95]
- Logic Programs with Annotated Disjunctions (LPADs) [Vennekens et al. ICLP04]
- ProbLog [De Raedt et al. IJCAI07]
- They differ in the way they define the distribution over logic programs

Logic Programs with Annotated Disjunctions

$$sneezing(X) : 0.7 \vee null : 0.3 \leftarrow flu(X).$$
$$sneezing(X) : 0.8 \vee null : 0.2 \leftarrow hay\_fever(X).$$
$$flu(bob).$$
$$hay\_fever(bob).$$

- Distributions over the head of rules
- *null* does not appear in the body of any rule
- Worlds obtained by selecting one atom from the head of every grounding of each clause

Reasoning Tasks

- Inference: we want to compute the probability of a query given the model and, possibly, some evidence
- Weight learning: we know the structural part of the model (the logic formulas) but not the numeric part (the weights) and we want to infer the weights from data
- Structure learning we want to infer both the structure and the weights of the model from data

Inference for PLP under DS

- Computing the probability of a query (no evidence)
- Knowledge compilation:
  - compile the program to an intermediate representation
    - Binary Decision Diagrams (ProbLog [De Raedt et al. IJCAI07], `cplint` [Riguzzi AIIA07,Riguzzi LJIGPL09], PITA [Riguzzi & Swift ICLP10])
    - deterministic, Decomposable Negation Normal Form circuit (d-DNNF) (ProbLog2 [Fierens et al. TPLP13])
    - Sentential Decision Diagrams
  - compute the probability by weighted model counting
- Bayesian Network based:
  - Convert to BN
  - Use BN inference algorithms (CVE [Meert et al. ILP09])
- Lifted inference

Parameter Learning

- Problem: given a set of interpretations, a program, find the parameters maximizing the likelihood of the interpretations (or of instances of a target predicate)
- Exploit the equivalence with BN to use BN learning algorithms
- The interpretations record the truth value of ground atoms, not of the choice variables
- Unseen data: relative frequency can't be used
- An Expectation-Maximization algorithm must be used:
  - Expectation step: the distribution of the unseen variables in each instance is computed given the observed data
  - Maximization step: new parameters are computed from the distributions using relative frequency
  - End when likelihood does not improve anymore

EMBLEM

- EM over Bdds for probabilistic Logic programs Efficient Mining [Bellodi & Riguzzi IDA13]
- Input: an LPAD; logical interpretations (data); *target* predicate(s)
- all ground atoms in the interpretations for the target predicate(s) correspond to as many queries
- BDDs encode the disjunction of explanations for each query Q

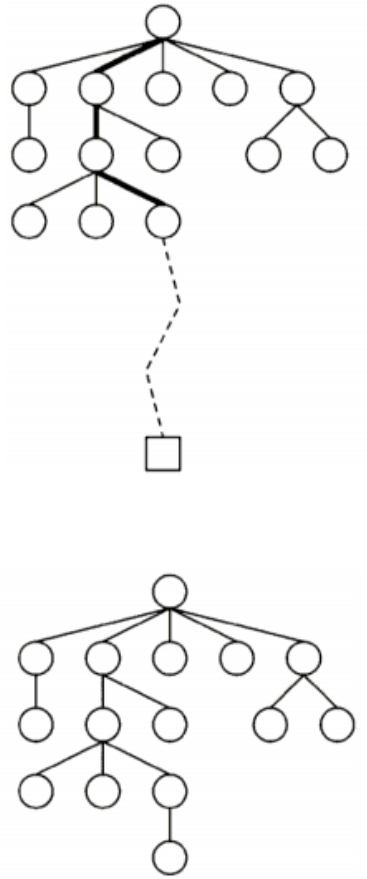
Structure Learning for LPADs

1. *Find the model and the parameters* that maximize the probability of the data (log-likelihood)
   1. SLIPCASE: Structure LearnIng of ProbabilistiC logic progrAmS with Em over bdds [Bellodi & Riguzzi ILP11]
      Beam search in the space of probabilistic programs
   2. SLIPCOVER: Structure LearnIng of Probabilistic logic program by searching OVER the clause space [Bellodi & Riguzzi TPLP14]
      1. Beam search in the space of clauses to find the promising ones
      2. Greedy search in the space of probabilistic programs guided by the LL of the data.

   - Both perform *parameter learning* by means of EMBLEM

Monte Carlo Tree Search

- MCTS: take random samples in the decision space and build a search tree in an incremental and asymmetric manner
- First a *tree policy* is used in order to find the most urgent node of the tree to expand
- Then a *simulation* phase is conducted from the selected node, by adding a new child node and using a *default policy* that suggests the sequence of actions ("simulation") to be chosen from this new node.
- Finally, the simulation result is *backpropagated* upwards to update the statistics of the nodes.



LEMUR: *LEarning with a Monte carlo Upgrade of tRee search*

- We consider each logic theory as a bandit problem, where each legal theory revision is an arm with unknown reward
- Tree policy: LEMUR selects one move, corresponding to a possible theory revision, according to a formula
- LEMUR descends to the selected child node and selects a new move until it reaches a leaf
- Then LEMUR starts the Monte Carlo simulation phase to score the theory at this leaf
- One random sequence of revisions is applied starting from the leaf theory until a *finite unknown horizon* is reached
- LEMUR stops the simulation after k steps, where k is a uniformly sampled random integer smaller than d, an input parameter.
- Once the horizon is reached, LEMUR produces a reward value Δ

Experiments - Area Under the PR Curve

System	HIV	UW-CSE	Mondial
LEMUR	0.83 ± 0.05	0.22 ± 0.08	0.95 ± 0.05
SLIPCOVER	0.82 ± 0.05	0.11 ± 0.08	0.86 ± 0.07
SLIPCASE	0.78 ± 0.05	0.03 ± 0.01	0.65 ± 0.06
LSM	0.37 ± 0.03	0.07 ± 0.02	-
ALEPH++	-	0.05 ± 0.01	0.87 ± 0.07
RDN-B	0.28 ± 0.06	0.28 ± 0.06	0.77 ± 0.07
MLN-BT	0.29 ± 0.04	0.18 ± 0.07	0.74 ± 0.10
MLN-BC	0.51 ± 0.04	0.06 ± 0.01	0.59 ± 0.09
BUSL	0.38 ± 0.03	0.01 ± 0.01	-

System	Carcinogenesis	Mutagenesis	Hepatitis
LEMUR	0.74	0.98 ± 0.02	0.85 ± 0.01
SLIPCOVER	0.60	0.95 ± 0.01	0.80 ± 0.01
SLIPCASE	0.63	0.92 ± 0.08	0.71 ± 0.05
LSM	-	-	0.53 ± 0.04
ALEPH++	0.74	0.95 ± 0.01	-
RDN-B	0.55	0.97 ± 0.03	0.88 ± 0.01
MLN-BT	0.50	0.92 ± 0.09	0.78 ± 0.02
MLN-BC	0.62	0.69 ± 0.20	0.79 ± 0.02
BUSL	-	-	0.51 ± 0.03

Semantic Web

- Aims at making information available in a form that is understandable by machines.
- Development of the Web Ontology Language (OWL)
  - Family of knowledge representation formalisms for defining knowledge bases.
  - Based on Description Logics
  - The OWL DL sublanguage is based on $\mathcal{SHOIN}(\mathbf{D})$.
- Incompleteness or uncertainty are intrinsic of much information on the World Wide Web

DISPONTE: DIstribution Semantics for Probabilistic ONTologiEs

- Idea: **annotate axioms of an ontology with a probability** and assume that the axioms are independent of the others.

$$0.6 :: Cat \sqsubseteq Pet$$

- A probabilistic ontology defines thus a distribution over normal theories (worlds) obtained by including an axiom in a world with a probability given by the annotation.

Example

$$0.4 :: \mathit{fluffy} : \mathit{Cat} \quad (1)$$
$$0.3 :: \mathit{tom} : \mathit{Cat} \quad (2)$$
$$0.6 :: \mathit{Cat} \sqsubseteq \mathit{Pet} \quad (3)$$

$$\exists \mathit{hasAnimal}.\mathit{Pet} \sqsubseteq \mathit{NatureLover} \quad (4)$$
$$(\mathit{kevin}, \mathit{fluffy}) : \mathit{hasAnimal} \quad (5)$$
$$(\mathit{kevin}, \mathit{tom}) : \mathit{hasAnimal} \quad (6)$$

- $Q = \mathit{kevin} : \mathit{NatureLover}$ is true in 3 worlds:
- $P(Q) = 0.348$
- Inference: BUNDLE by means of knowledge compilation

Learning in DISPONTE

- EDGE
  1. learns the parameters of a probabilistic KB by taking as input a DL KB and a number of positive and negative examples
- LEAP
  1. learns the parameters and the structure of a probabilistic KB by taking as input a DL KB and a number of positive and negative examples
  2. exploits EDGE for learning the parameters

Experiments

	gov.uk		DBPedia	
	EDGE	ARs	EDGE	ARs
AUCPR	0.9702 ± 0.029	0.8804 ± 0.016	0.9784 ± 0.048	0.5916 ± 0.099
AUCROC	0.9796 ± 0.017	0.9158 ± 0.017	0.9902 ± 0.022	0.4346 ± 0.132

LEAP: "LEArning Probabilistic description logics"

- LEAP learn the structure and the parameters of DISPONTE KB and combines the learning system CELOE with EDGE.
- CELOE [Lehmann et. al. JWS11] solves the learning problem
- Given:
  - a concept name `Target`;
  - a knowledge base $\mathcal{K}$ not containing `Target`;
  - a space of possible concepts $\mathcal{C}$;
  - a set of positive examples E^+ with elements of the form a : `Target` ($a \in \mathbf{I}$);
  - a set of negative examples E^- with elements of the form a : `Target` ($a \in \mathbf{I}$);
- Find a concept expression $C \in \mathcal{C}$ such that:
  - `Target` does not occur in C (acyclic definition);
  - $\forall e^+ \in E^+, \mathcal{K} \cup \{\texttt{Target} \equiv C\} \models e^+$;
  - $\forall e^- \in E^-, \mathcal{K} \cup \{\texttt{Target} \equiv C\} \not\models e^-$.

Experiments

Carcinogenesis

Original KB		LEAP	
AUCPR	AUCROC	AUCPR	AUCROC
0.534 ± 0.1082	0.4452 ± 0.0510	0.8006 ± 0.2399	0.798 ± 0.2463

Conclusions

- Probabilistic Logic Programming
- Probabilistic Description Logics
- Inference Algorithms
- Learning Algorithms
- Future work
  - Implement new search types for structure learning
  - Improve the speed of learning by map reduce
  - exploit lifted inference in learning