Probabilistic Logics in Machine Learning

ML@Unife group members:
Elena Bellodi[2] Giuseppe Cota[1] Arnaud Nguembang Fadja[1]
Evelina Lamma[1] Fabrizio Riguzzi[2] Riccardo Zese[1]

[1] Dipartimento di Ingegneria – University of Ferrara

[2] Dipartimento di Matematica e Informatica – University of Ferrara
name.surname@unife.it

Outline

Logic

- Useful to model domains with complex relationships among entities
- Various forms:
  - First Order Logic
  - Logic Programming
  - Description Logics

First Order Logic

- Very expressive
- Open World Assumption
- Undecidable

$$\forall x \; Intelligent(x) \rightarrow GoodMarks(x)$$
$$\forall x, y \; Friends(x, y) \rightarrow (Intelligent(x) \leftrightarrow Intelligent(y))$$

Logic Programming

- A subset of First Order Logic
- Closed World Assumption
- Turing complete
- Prolog

$$
\begin{aligned}
&flu(bob).\\
&hay\_fever(bob).\\
&sneezing(X) \leftarrow flu(X).\\
&sneezing(X) \leftarrow hay\_fever(X).
\end{aligned}
$$

Description Logics

- Subsets of First Order Logic
- Open World Assumption
- Decidable, efficient inference
- Special syntax using concepts (unary predicates) and roles (binary predicates)[1]

$$fluffy : Cat$$
$$tom : Cat$$
$$Cat \sqsubseteq Pet$$
$$\exists hasAnimal.Pet \sqsubseteq NatureLover$$
$$(kevin, fluffy) : hasAnimal$$
$$(kevin, tom) : hasAnimal$$

[1] http://www.cs.man.ac.uk/~ezolin/dl/, a nice Web application that shows decidability and complexity issues depending on which operator you decide to include/exclude in your logic, with extensive references

Combining Logic and Probability

- Logic does not handle well uncertainty
- Graphical models do not handle well relationships among entities
- Solution: combine the two
- Many approaches proposed in the areas of Logic Programming, Uncertainty in AI, Machine Learning, Databases, Knowledge Representation
  - Distribution Semantics [Sato ICLP95]: given a query, find the worlds (normal logic programs) where the query is true and summing their probability

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

Example Program (LPAD) Worlds

$sneezing(bob) \leftarrow flu(bob).$
$sneezing(bob) \leftarrow hay\_fever(bob).$
$flu(bob).$
$hay\_fever(bob).$
$P(w_1) = 0.7 \times 0.8$

$null \leftarrow flu(bob).$
$sneezing(bob) \leftarrow hay\_fever(bob).$
$flu(bob).$
$hay\_fever(bob).$
$P(w_2) = 0.3 \times 0.8$

$sneezing(bob) \leftarrow flu(bob).$
$null \leftarrow hay\_fever(bob).$
$flu(bob).$
$hay\_fever(bob).$
$P(w_3) = 0.7 \times 0.2$

$null \leftarrow flu(bob).$
$null \leftarrow hay\_fever(bob).$
$flu(bob).$
$hay\_fever(bob).$
$P(w_4) = 0.3 \times 0.2$

$$P(Q) = \sum_{w \in W_{\mathcal{T}}} P(Q, w) = \sum_{w \in W_{\mathcal{T}}} P(Q|w)P(w) = \sum_{w \in W_{\mathcal{T}} : w \models Q} P(w)$$

- $sneezing(bob)$ is true in 3 worlds
- $P(sneezing(bob)) = 0.7 \times 0.8 + 0.3 \times 0.8 + 0.7 \times 0.2 = 0.94$

ProbLog

$$
\begin{aligned}
&sneezing(X) \leftarrow flu(X), flu\_sneezing(X).\\
&sneezing(X) \leftarrow hay\_fever(X), hay\_fever\_sneezing(X).\\
&flu(bob).\\
&hay\_fever(bob).\\
&0.7 :: flu\_sneezing(X).\\
&0.8 :: hay\_fever\_sneezing(X).
\end{aligned}
$$

- Distributions over facts
- Worlds obtained by selecting or not every grounding of each probabilistic fact

Example Program (ProbLog) Worlds

- 4 worlds

$$
\begin{aligned}
&sneezing(X) \leftarrow flu(X), flu\_sneezing(X).\\
&sneezing(X) \leftarrow hay\_fever(X), hay\_fever\_sneezing(X).\\
&flu(bob).\\
&hay\_fever(bob).
\end{aligned}
$$

$$
\begin{array}{ll}
flu\_sneezing(bob). & \\
hay\_fever\_sneezing(bob). & hay\_fever\_sneezing(bob).\\
P(w_1) = 0.7 \times 0.8 & P(w_2) = 0.3 \times 0.8\\
flu\_sneezing(bob). & \\
P(w_3) = 0.7 \times 0.2 & P(w_4) = 0.3 \times 0.2
\end{array}
$$

- $sneezing(bob)$ is true in 3 worlds
- $P(sneezing(bob)) = 0.7 \times 0.8 + 0.3 \times 0.8 + 0.7 \times 0.2 = 0.94$

Probabilistic Logic Programming Online

- `http://cplint.eu/`
  - Inference (knwoledge compilation, Monte Carlo)
  - Parameter learning (EMBLEM)
  - Structure learning (SLIPCOVER)
- `https://dtai.cs.kuleuven.be/problog/`
  - Inference (knwoledge compilation, Monte Carlo)
  - Parameter learning (LFI-ProbLog)

Expressive Power

- All languages under the distribution semantics have the same expressive power
- LPADs have the most general syntax
- There are transformations that can convert each one into the others
- ProbLog to LPAD: direct mapping

Description Logics

- DISPONTE: "DIstribution Semantics for Probabilistic ONTologiEs" [Riguzzi et al. SWJ15]
- Probabilistic axioms:
  - $p :: E$
    e.g., $p :: C \sqsubseteq D$ represents the fact that we believe in the truth of $C \sqsubseteq D$ with probability p.
- DISPONTE applies the distribution semantics of probabilistic logic programming to description logics

DISPONTE

- World w: regular DL KB obtained by selecting or not the probabilistic axioms
- Probability of a query Q given a world w: $P(Q|w) = 1$ if $w \models Q$, 0 otherwise
- Probability of Q
  $P(Q) = \sum_w P(Q, w) = \sum_w P(Q|w)P(w) = \sum_{w:w \models Q} P(w)$

Example

$0.4 :: fluffy : Cat$

$0.3 :: tom : Cat$

$0.6 :: Cat \sqsubseteq Pet$

$\exists hasAnimal.Pet \sqsubseteq NatureLover$

$(kevin, fluffy) : hasAnimal$

$(kevin, tom) : hasAnimal$

- $P(kevin : NatureLover) =$
  $0.4 \times 0.3 \times 0.6 + 0.4 \times 0.7 \times 0.6 + 0.6 \times 0.3 \times 0.6 = 0.348$

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

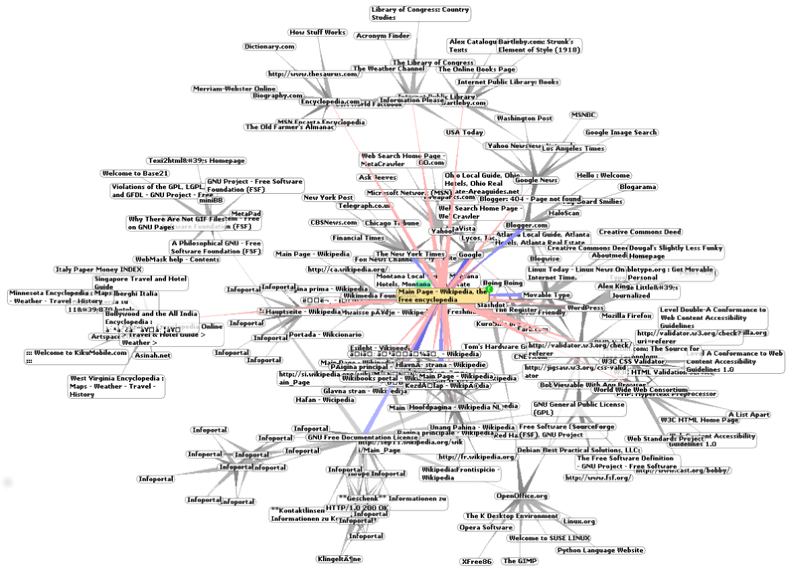
Applications

- Link prediction: given a (social) network, compute the probability of the existence of a link between two entities (UWCSE)

```
advisedby(X, Y) :0.7 :-
  publication(P, X),
  publication(P, Y),
  student(X).
```

Applications

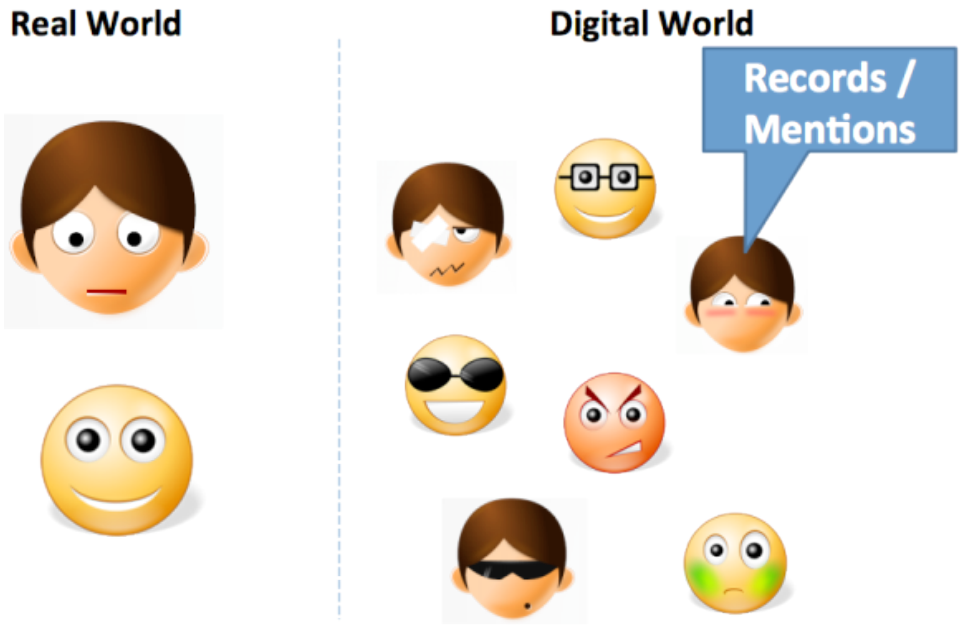
- Classify web pages on the basis of the link structure (WebKB)



```
coursePage(Page1): 0.3 :- linkTo(Page2,Page1),coursePage(Page2).
coursePage(Page1): 0.6 :- linkTo(Page2,Page1),facultyPage(Page2).
...
coursePage(Page): 0.9 :- has('syllabus',Page).
...
```

Applications

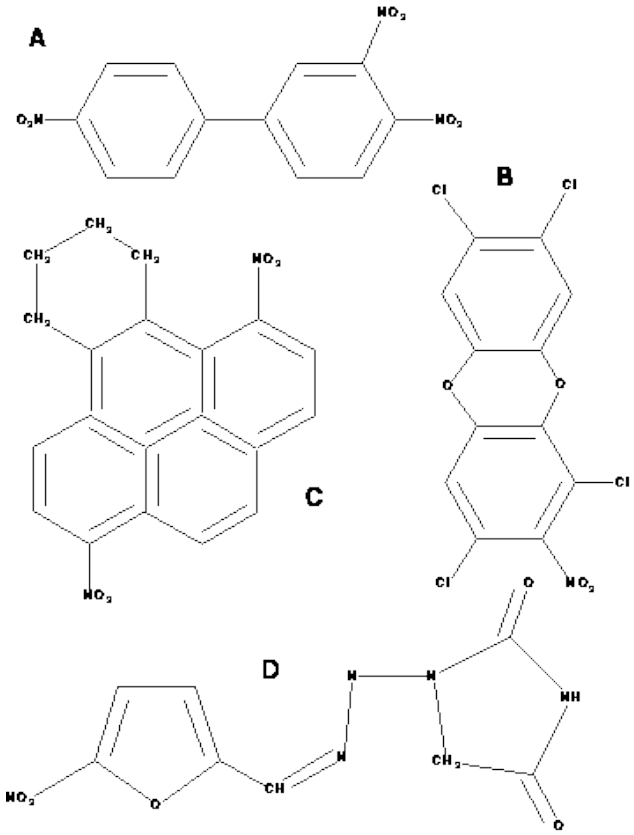
- Entity resolution: identify identical entities in text or databases




```
samebib(A,B):0.9 :-
samebib(A,C), samebib(C,B).
sameauthor(A,B):0.6 :-
  sameauthor(A,C), sameauthor(C,B).
sametitle(A,B):0.7 :-
  sametitle(A,C), sametitle(C,B).
samevenue(A,B):0.65 :-
  samevenue(A,C), samevenue(C,B).
samebib(B,C):0.5 :-
  author(B,D),author(C,E),sameauthor(D,E).
samebib(B,C):0.7 :-
  title(B,D),title(C,E),sametitle(D,E).
samebib(B,C):0.6 :-
  venue(B,D),venue(C,E),samevenue(D,E).
samevenue(B,C):0.3 :-
  haswordvenue(B,logic),
  haswordvenue(C,logic).
...
```

Applications

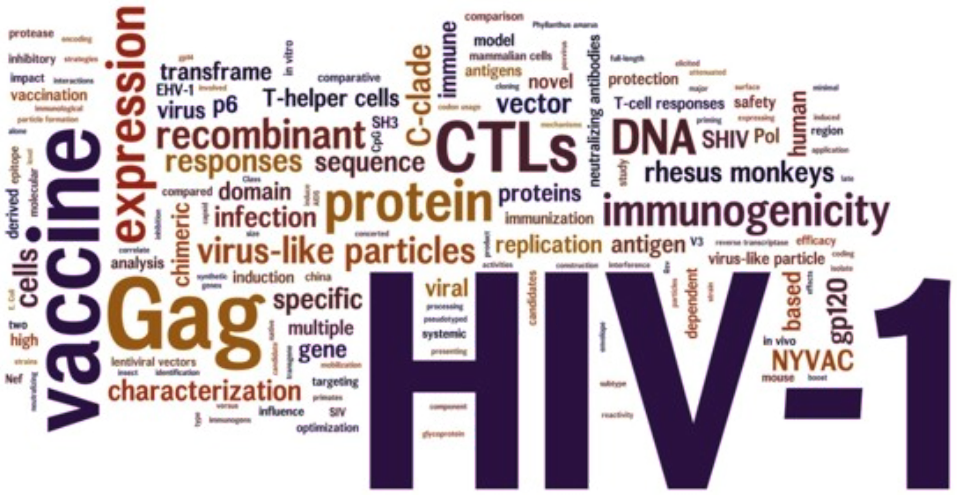
- Chemistry: given the chemical composition of a substance, predict its mutagenicity or its carcenogenicity




```
active(A):0.4 :-
   atm(A,B,c,29,C),
   gteq(C,-0.003),
   ring_size_5(A,D).
active(A):0.6:-
   lumo(A,B), lteq(B,-2.072).
active(A):0.3 :-
   bond(A,B,C,2),
   bond(A,C,D,1),
   ring_size_5(A,E).
active(A):0.7 :-
   carbon_6_ring(A,B).
active(A):0.8 :-
   anthracene(A,B).
...
```

Applications

- Medicine: diagnose diseases on the basis of patient information (Hepatitis), influence of genes on HIV, risk of falling of elderly people



Inference for probabilistic logic programming under distribution semantics

- Computing the probability of a query
- Knowledge compilation:
  - compile the program to an intermediate representation
  - compute the probability by weighted model counting
- Bayesian Network based:
  - Convert to Bayesian Network
  - Use Bayesian Network inference algorithms (CVE [Meert et al. ILP09])
- Lifted inference

Approximate Inference

- Inference problem is #P hard
- For large models inference is intractable
- Approximate inference
  - Monte Carlo: draw samples of the truth value of the query
  - Iterative deepening: gives a lower and an upper bound
  - Compute only the best k explanations: branch and bound, gives a lower bound

PITA and MCINTYRE

- PITA performs a program transformation technique and uses techniques alternative to tabling and answer subsumption on knowledge compilation to compute the probability of queries.
- MCINTYRE performs approximate inference by sampling. It uses a different program transformation technique than that of PITA. It is able to handle continuous random variables.

Inference in DISPONTE

- The probability of a query Q can be computed according to the distribution semantics by first finding the explanations for Q in the knowledge base
- Explanation: subset of axioms of the KB that is sufficient for entailing Q
- All the explanations for Q must be found, corresponding to all ways of proving Q
- Application of knowledge compilation to explanations

Example

$E_1 = 0.4 :: fluffy : Cat$

$E_2 = 0.3 :: tom : Cat$

$E_3 = 0.6 :: Cat \sqsubseteq Pet$

$\exists hasAnimal.Pet \sqsubseteq NatureLover$

$(kevin, fluffy) : hasAnimal$

$(kevin, tom) : hasAnimal$

- $Q = kevin : NatureLover$ has two explanations:

$$\{ (E_1), (E_3) \}$$
$$\{ (E_2), (E_3) \}$$

- $P(Q) = 0.4 \times 0.6 \times (1 - 0.3) + 0.3 \times 0.6 = 0.348$

BUNDLE and TRILL

- BUNDLE performs inference over DISPONTE knowledge bases. It exploits an underlying ontology reasoner able to return all explanations for a query and applies knowledge compilation for computing the probability
- TRILL resolves the same problem of BUNDLE, but it is completely implemented in Prolog. It is part of a framework containing also TRILL^P and TORNADO, both based on TRILL.
  - Different approaches for collecting the explanations

TRILL

- Available online at `http://trill.ml.unife.it/`
- Pets example `http://trill.ml.unife.it/trill_on_swish/example/peoplePets.owl`

Combination of probabilistic logic programming and description logics

- Unified framework exploiting LPAD and DISPONTE Description Logics
- Allows the combination of different closure assumptions
- SPHERE: algorithm combining PITA and TRILL
  - Able to answer queries and computing their probability

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)
- The interpretations record the truth value of ground atoms, not of the choice variables
- Unseen data: relative frequency can't be used

Parameter Learning

- 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
- 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
- use of knowledge compilation to encode the explanations for each query Q

EDGE

- Em over bDds for description loGics paramEter learning
- EDGE is inspired to EMBLEM [Bellodi and Riguzzi, IDA 2013]
- Takes as input a DL theory and a number of examples that represent queries.
- EDGE computes the explanations of each example using BUNDLE
- EDGE^{MR} (EDGE powered by MapReduce) parallelizes EDGE:
  - The examples are divided in chunks and associated to different processes, since each example is independent of the others
  - The learning phase is spread on all the workers as well

EDGE: Experiments

	edu.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

Structure Learning for LPADs

- Given a trivial LPAD or an empty one, a set of interpretations (data)
- *Find the model and the parameters* that maximize the probability of the data (log-likelihood)
- SLIPCOVER: Structure LearnIng of Probabilistic logic program by searching OVER the clause space
- *Parameter learning* by means of EMBLEM [Riguzzi & Bellodi TPLP 2015]

Experiments - Area Under the PR Curve

System	HIV	UW-CSE	Mondial
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	-

Experiments - Area Under the PR Curve

System	Carcinogenesis	Mutagenesis	Hepatitis
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

LEAP

- LEArning Probabilistic description logics
- Learns the structure of a probabilistic KB by taking as input a DL KB and a number of positive and negative examples
- learns (acyclic) concept expressions C_i for one or more `Target` classes
- EDGE is run on the extended theory to compute the log-likelihood of the data LL and the updated parameters

- LEAP^{MR} (LEAP powered by MapReduce) parallelizes LEAP by exploiting EDGE^{MR}

LEAP: Experiments

Carcinogenesis	EDGE	LEAP	p-value
AUCPR	0.5340 ± 0.108	0.8006 ± 0.240	0.0603
AUCROC	0.4452 ± 0.051	0.7980 ± 0.246	0.0360

Conclusions

- Logics are useful in domains with complex relationships between entities
- Combination with probability allows the management of real world domains, with uncertain information
- PITA, MCINTYRE, EMBLEM, and SLIPCOVER allows reasoning on LPAD
  - Are all part of the framework cplint
- BUNDLE, TRILL (and its extensions), EDGE, and LEAP allows reasoning on DISPONTE description logics

THANKS FOR LISTENING AND ANY QUESTIONS ?

References

- Bellodi, E. and Riguzzi, F. (2013). Expectation Maximization over binary decision diagrams for probabilistic logic programs. Intelligent Data Analysis, 17(2).
- Meert, W., Struyf, J., and Blockeel, H. (2009). CP-Logic theory inference with contextual variable elimination and comparison to bdd based inference methods. In ILP 2009.
- Sato, T. (1995). A statistical learning method for logic programs with distribution semantics. In International Conference on Logic Programming, pages 715-729.
- Riguzzi, Fabrizio, et al. "Probabilistic description logics under the distribution semantics." Semantic Web 6.5 (2015): 477-501.