Discriminative Structure and Parameter Learning for Markov Logic Networks

Tuyen N. Huynh and Raymond J. Mooney

Machine Learning Group
Department of Computer Sciences
University of Texas at Austin

ICML'08, Helsinki, Finland

Motivation

- **New Statistical Relational Learning (SRL) formalisms combining logic with probability have been proposed:**
  - Knowledge-based model construction [Wellman et al., 1992]
  - Stochastic logic programs [Muggleton, 1996]
  - Relational Bayesian Networks [Jaeger 1997]
  - Bayesian logic programs [Kersting and De Raedt, 2001]
  - CLP(BN) [Costa et al. 03]
  - Markov logic networks (MLNs) [Richardson & Domingos, 2004]
  - etc …
- **Question**: Do these advanced systems perform better than pure first-order logic system, traditional ILP methods, on standard benchmark ILP problems?

In this work, we answer this question for MLNs, one of the most general and expressive models

Background

Markov Logic Networks

[Richardson & Domingos, 2006]

- An MLN is a weighted set of first-order formulas

1.98579	alk_groups(b,0) => less_toxic(a,b)
4.19145	ring_subst_3(a,c) ^ polar(c,POLAR2) => less_toxic(a,b)
10	less_toxic(a,b) ^ less_toxic(b,c) => less_toxic(a,c)

- The clauses are called the structure
- Larger weight indicates stronger belief that the clause should hold
- Probability of a possible world X:

$$P(X = x) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(x) \right)$$

Weight of formula i

No. of true groundings of formula i in x

Inference in MLNs

- **MAP/MPE inference:** find the most likely state of the world given the evidence
  - MaxWalkSAT algorithm [Kautz et al., 1997]
  - LazySAT algorithm [Singla & Domingos, 2006]
- **Computing the probability of a query:**
  - MC-SAT algorithm [Poon & Domingos, 2006]
  - Lifted first-order belief propagation [Singla & Domingos, 2008]

Existing learning methods for MLNs

- **Structure learning:**
  - MSL[Kok & Domingos 05], BUSL [Mihalkova & Mooney, 07]:
    - Greedily search for clauses which optimize a non-discriminative metric: Weighted Pseudo-Log Likelihood (**WPLL)**
- **Weight learning:**
  - **Generative** learning: maximize the pseudo-log likelihood [Richardson & Domingos, 2006]
  - **Discriminative** learning: maximize the Conditional Log Likelihood (CLL)
    - [Lowd & Domingos, 2007]: Found that the Preconditioned Scaled Conjugated Gradient (PSCG) performs best

Initial results

- Initial results:

Average accuracy

Data set	MLN1*	MLN2**	ALEPH
Alzheimer amine	50.1 ± 0.5	51.3 ± 2.5	81.6 ± 5.1
Alzheimer toxic	54.7 ± 7.4	51.7 ± 5.3	81.7 ± 4.2
Alzheimer acetyl	48.2 ± 2.9	55.9 ± 8.7	79.6 ± 2.2
Alzheimer memory	50 ± 0.0	49.8 ± 1.6	76.0 ± 4.9

*MLN1: MSL + PSCG
**MLN2: BUSL+ PSCG

- What happened: The existing learning methods for MLNs fail to capture the relations between the background predicates and the target predicate

→ New discriminative learning methods for MLNs

Generative vs Discriminative in SRL

- **Generative** learning:
  - Find the relations between all the predicates in the domain
  - Find a structure and a set of parameters which optimize a generative metric such as the log likelihood
- **Discriminative** learning:
  - Find the relations between a target predicate and other predicates
  - Find a structure and a set of parameters which optimize a discriminative metric such as the conditional log likelihood

Proposed approach

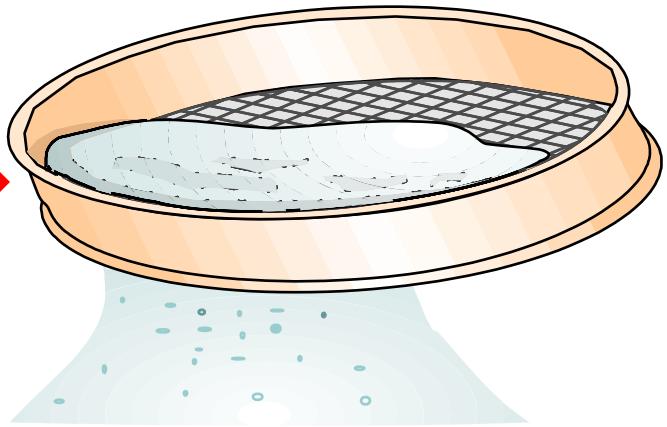
Proposed approach

Step 1

Clause Learner

Discriminative structure learning
(Generating candidate clauses)

Step 2



Discriminative weight learning
(Selecting good clauses)

Discriminative structure learning

- Goal: Learn the relations between background knowledge and the target predicate
- Solution: Use a variant of ALEPH [Srinivasan, 2001], called ALEPH++, to produce a larger set of candidate clauses:
  - Score the clauses by *m*-estimate [Dzeroski, 1991], a Bayesian estimate of the accuracy of a clause.
  - Keep all the clauses having an *m*-estimate greater than a pre-defined threshold (0.6), instead of the final theory produced by ALEPH.

Facts

r _subst_1(A1,H)
r_subst_1(B1,H)
r _subst_1(D1,H)
x_subst(B1,7,CL)
x_subst(HH1,6,CL)
x _subst(D1,6,OCH3)
polar(CL,POLAR3)
polar(OCH3,POLAR2)
great_polar(POLAR3,POLAR2)
size(CL,SIZE1)
size(OCH3,SIZE2)
great_size(SIZE2,SIZE1)
alk_groups(A1,0)
alk groups(B1,0)
alk_groups(D1,0)
alk_groups(HH1,1)
flex(CL,FLEX0)
flex(OCH3,FLEX1)
less_toxic(A1,D1)
less_toxic(B1,D1)
less_toxic(HH1,A1)

ALEPH++

Candidate clauses

x_subst(d1,6,m1) ^ alk_groups(d1,1) => less_toxic(d1,d2)

alk_groups(d1,0) ^ r_subst_1(d2,H) => less_toxic(d1,d2)

x_subst(d1,6,m1) ^ polar(m1,POLAR3) ^ alk_groups(d1,1) => less_toxic(d1,d2)

….

They are all non-recursive clauses

Discriminative weight learning

- Goal: learn weights for clauses that allow accurate prediction of the target predicate.
- Solution: maximize CLL with L_1-regularization [Lee et al., 2006]
  - Use exact inference instead of approximate inferences
  - Use L_1-regularization instead of L_2-regularization

Exact inference

- Since the candidate clauses are non-recursive, the target predicate appears only once in each clause:
  - The probability of a target predicate atom being true or false only depends on the evidence.
  - The target atoms are independent.

$$\log P(Y = y|X = x) = \log \prod_{j=1}^{n} P(Y_j = y_j|X = x)$$

$$= \sum_{j=1}^{n} \log P(Y_j = y_j|X = x)$$

and,

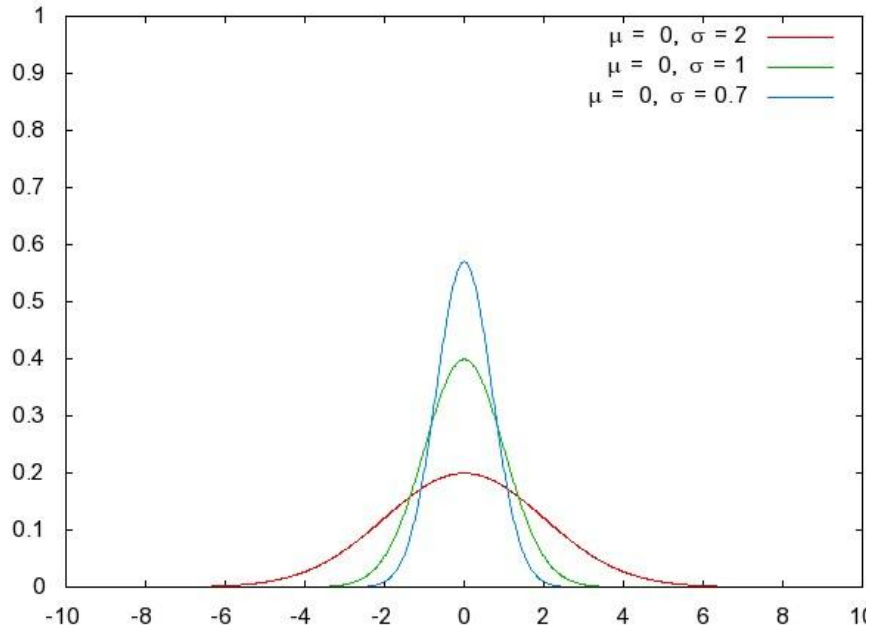
$$P(Y_j = y_j|X = x) = \frac{exp(\sum_{i \in \mathcal{F}_{Y_j}} w_i n_i(x, y_{[Y_j = y_j]}))}{exp(\sum_{i \in \mathcal{F}_{Y_j}} w_i n_i(x, y_{[Y_j = 0]})) + exp(\sum_{i \in \mathcal{F}_{Y_j}} w_i n_i(x, y_{[Y_j = 1]}))}$$

L_1-regularization

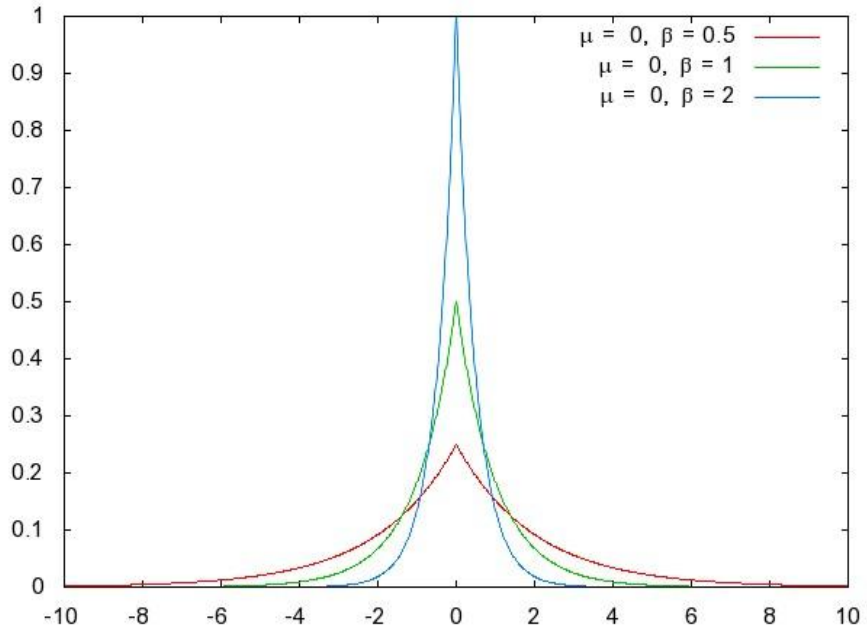
- Put a Laplacian prior with zero mean on each weight w_i

$$P(w_i) = (\beta / 2) \cdot \exp(-\beta \mid w_i \mid)$$

Gaussian prior



Laplace prior



- L_1 ignores irrelevant features by setting many weights to zero [Ng, 2004]
- Larger value of β, the regularizing parameter, corresponds to smaller variance of the prior distribution
- Use the OWL-QN package [(Andrew & Gao, 2007] to solve the optimization problem

Facts

r _subst_1(A1,H)
r_subst_1(B1,H)
r _subst_1(D1,H)
x_subst(B1,7,CL)
x_subst(HH1,6,CL)
x _subst(D1,6,OCH3)
…

Candidate clauses

alk_groups(d1,0) ^ r_subst_1(d2,H) => less_toxic(d1,d2)

x_subst(d1,6,m1) ^ polar(m1,POLAR3) ^ alk_groups(d1,1) => less_toxic(d1,d2)

x_subst(d1,6,m1) ^ alk_groups(d1,1) => less_toxic(d1,d2)
….

L_1 weight learner

Weighted clauses

0 x_subst(v8719,6,v8774) ^ alk_groups(v8719,1) => less_toxic(v8719,v8720)

0.34487 alk_groups(d1,0) ^ r_subst_1(d2,H) => less_toxic(d1,d2)

2.70323 x_subst(d1,6,m1) ^ polar(m1,POLAR3) ^ alk_groups(d1,1) => less_toxic(d1,d2)
….

Experiments

Data sets

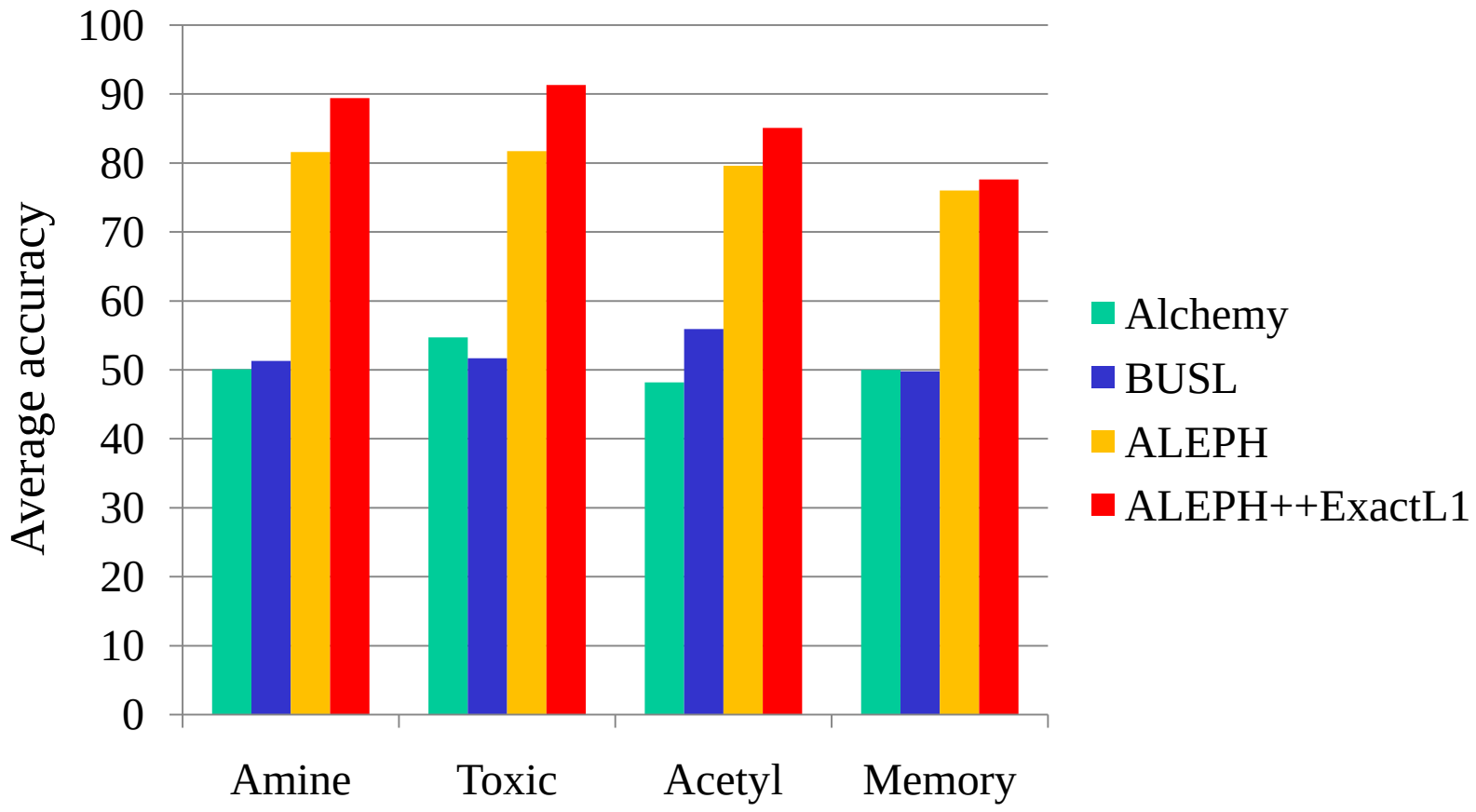
- ILP benchmark data sets about comparing drugs for Alzheimer's disease on four biochemical properties:
  - Inhibition of amine re-uptake
  - Low toxicity
  - High acetyl cholinesterase inhibition
  - Good reversal of scopolamine-induced memory

Data set	# Examples	% Pos. example	#Predicates
Alzheimer amine	686	50%	30
Alzheimer toxic	886	50%	30
Alzheimer acetyl	1326	50%	30
Alzheimer memory	642	50%	30

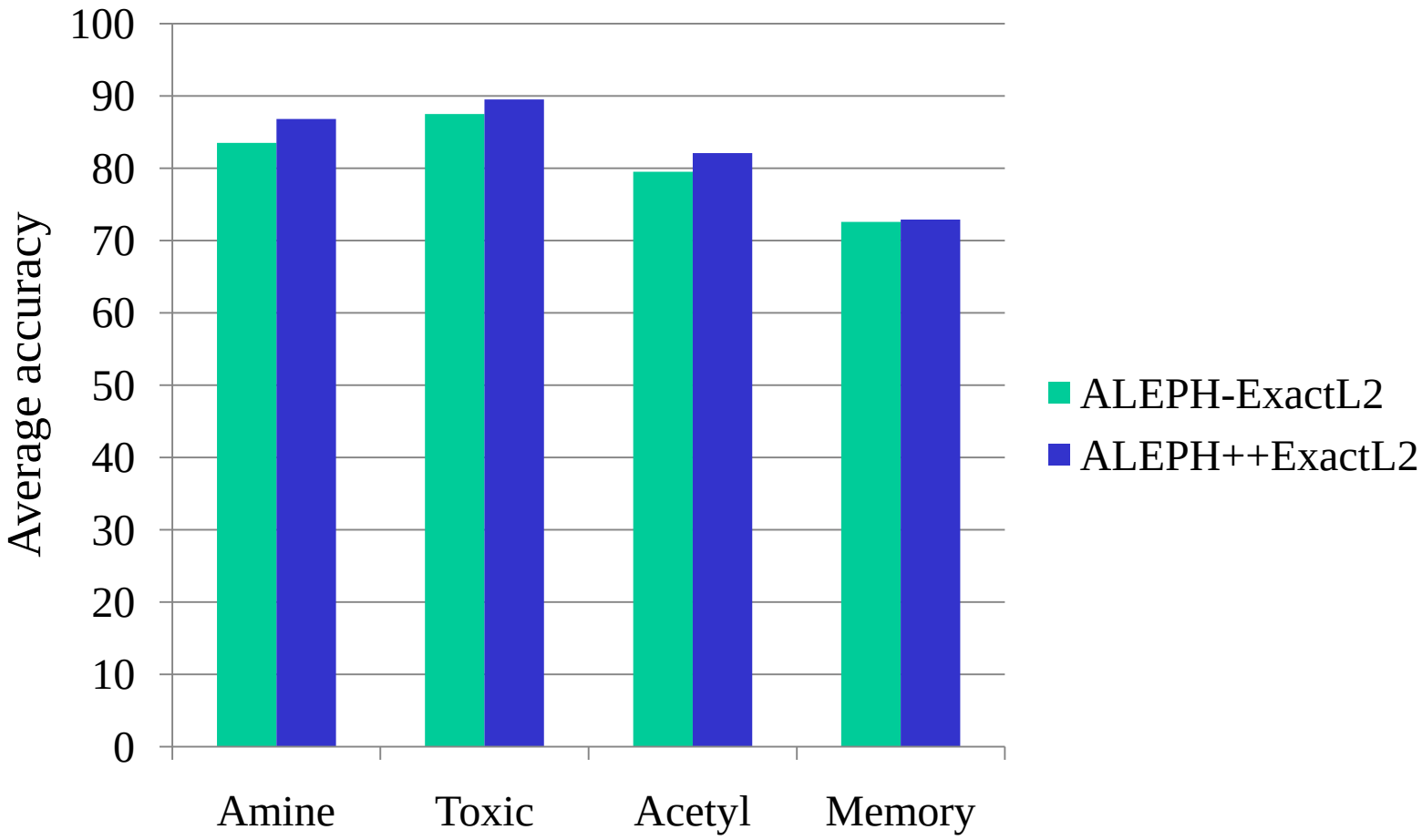
Methodology

- 10-fold cross-validation
- Metric:
  - Average predictive accuracy over 10 folds
  - Average Area Under the ROC curve over 10 folds

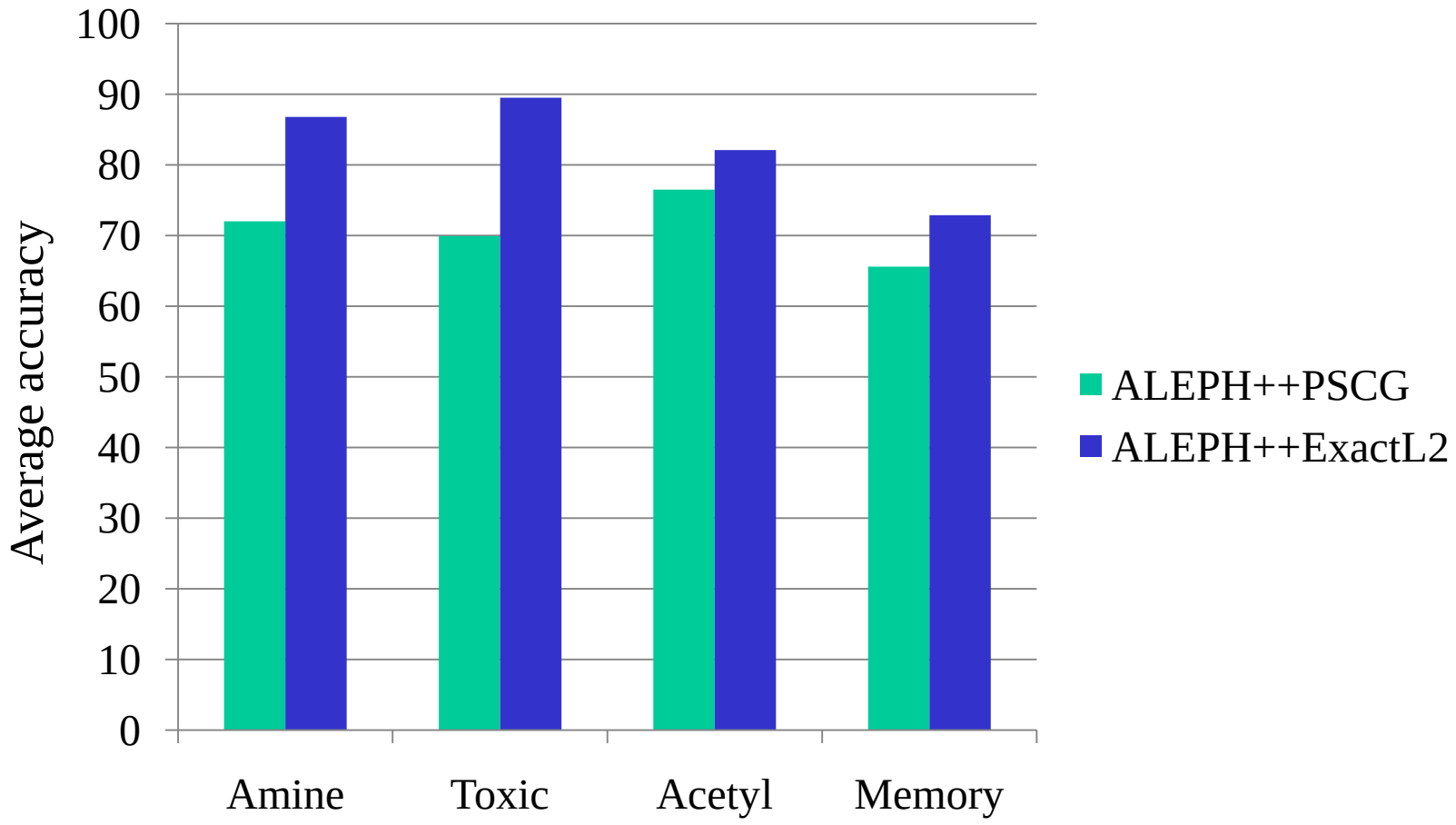
- Q1: Does the proposed approach perform better than existing learning methods for MLNs and traditional ILP methods?



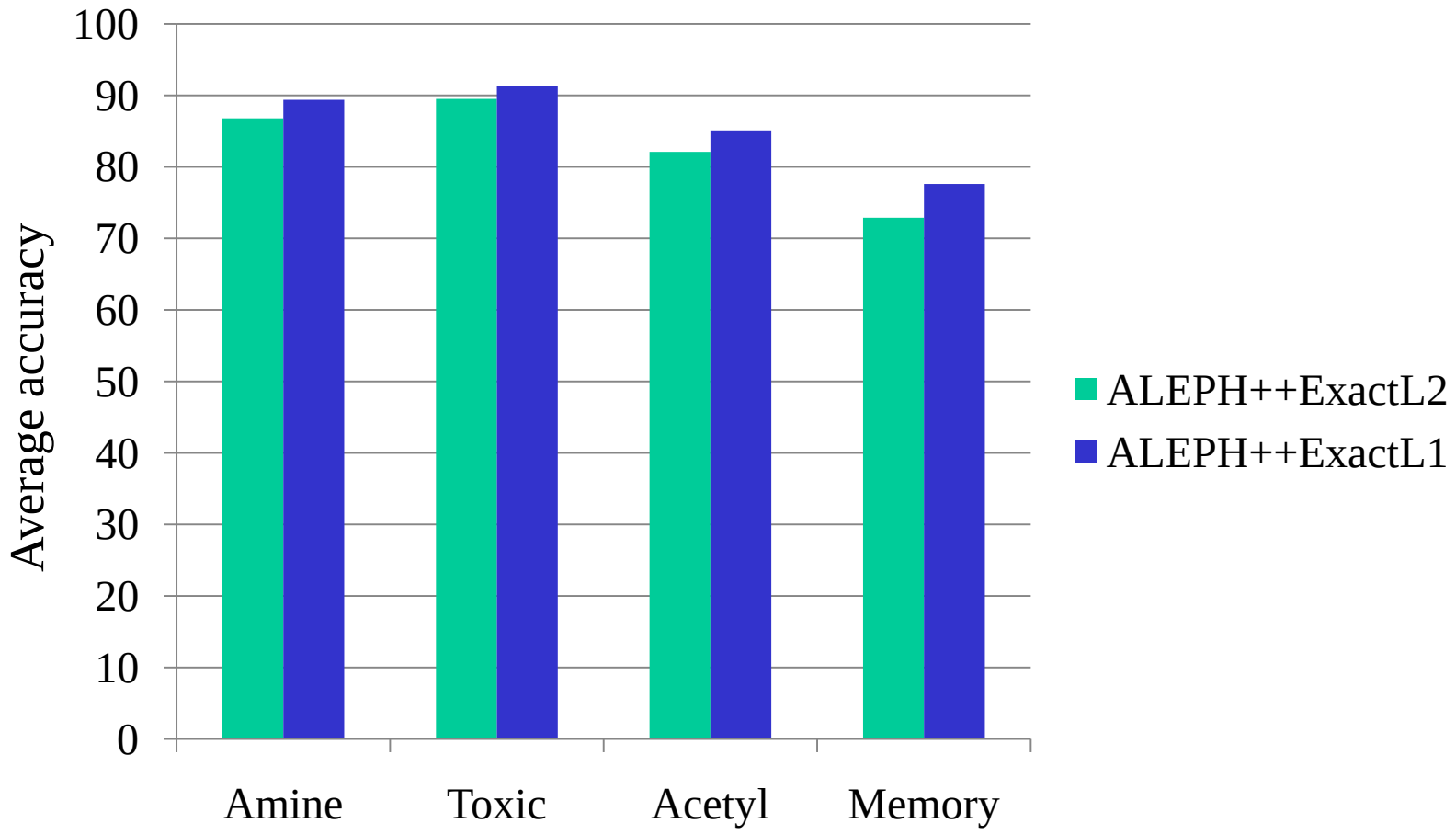
- Q2: The contribution of each component
  - ALEPH vs ALEPH++



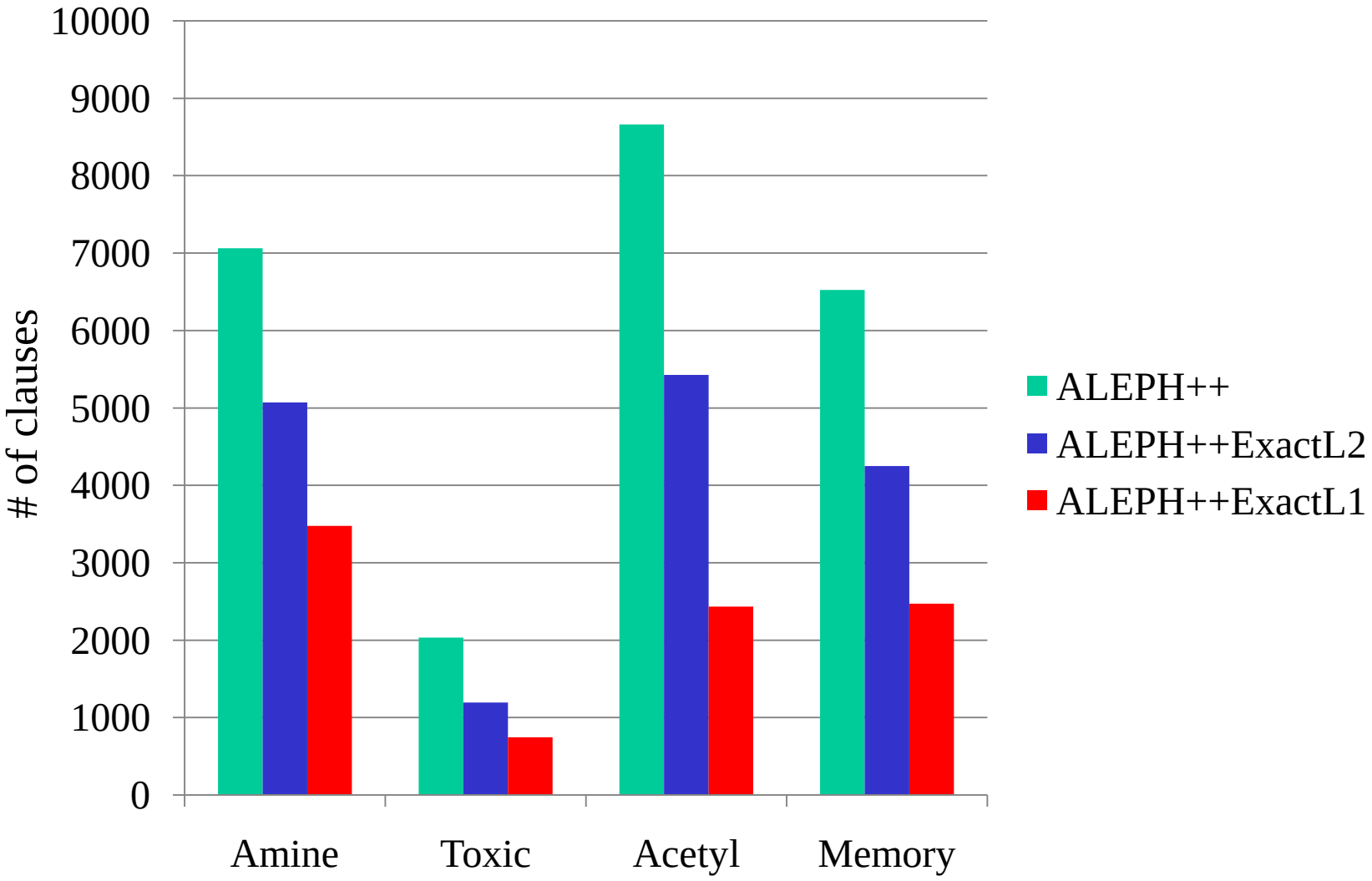
- Q2: The contribution of each component
  - Exact vs. approximate inference



- Q2: The contribution of each component
  - L1 vs. L2 regularization

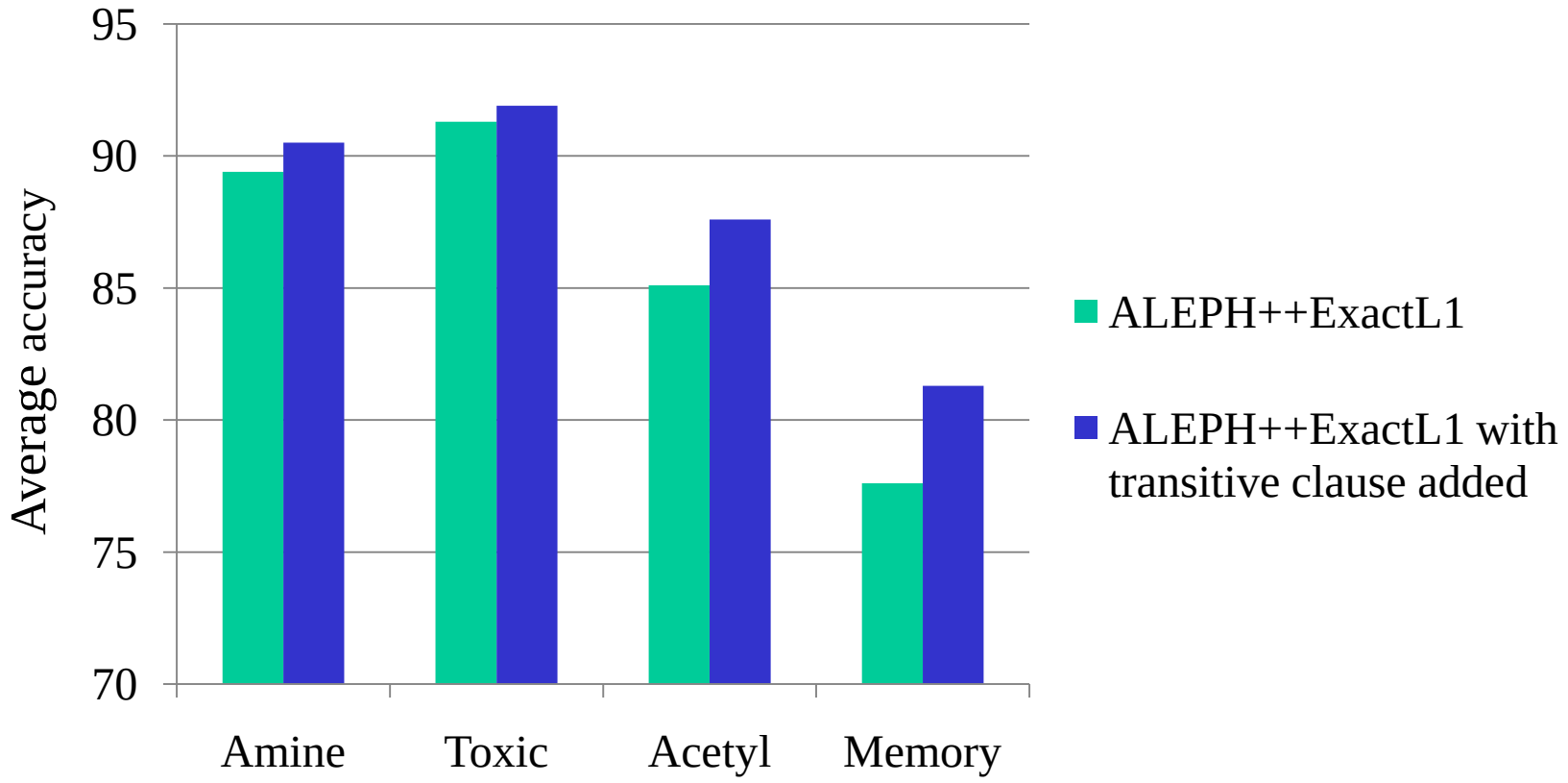


- Q3: The effect of L_1-regularization

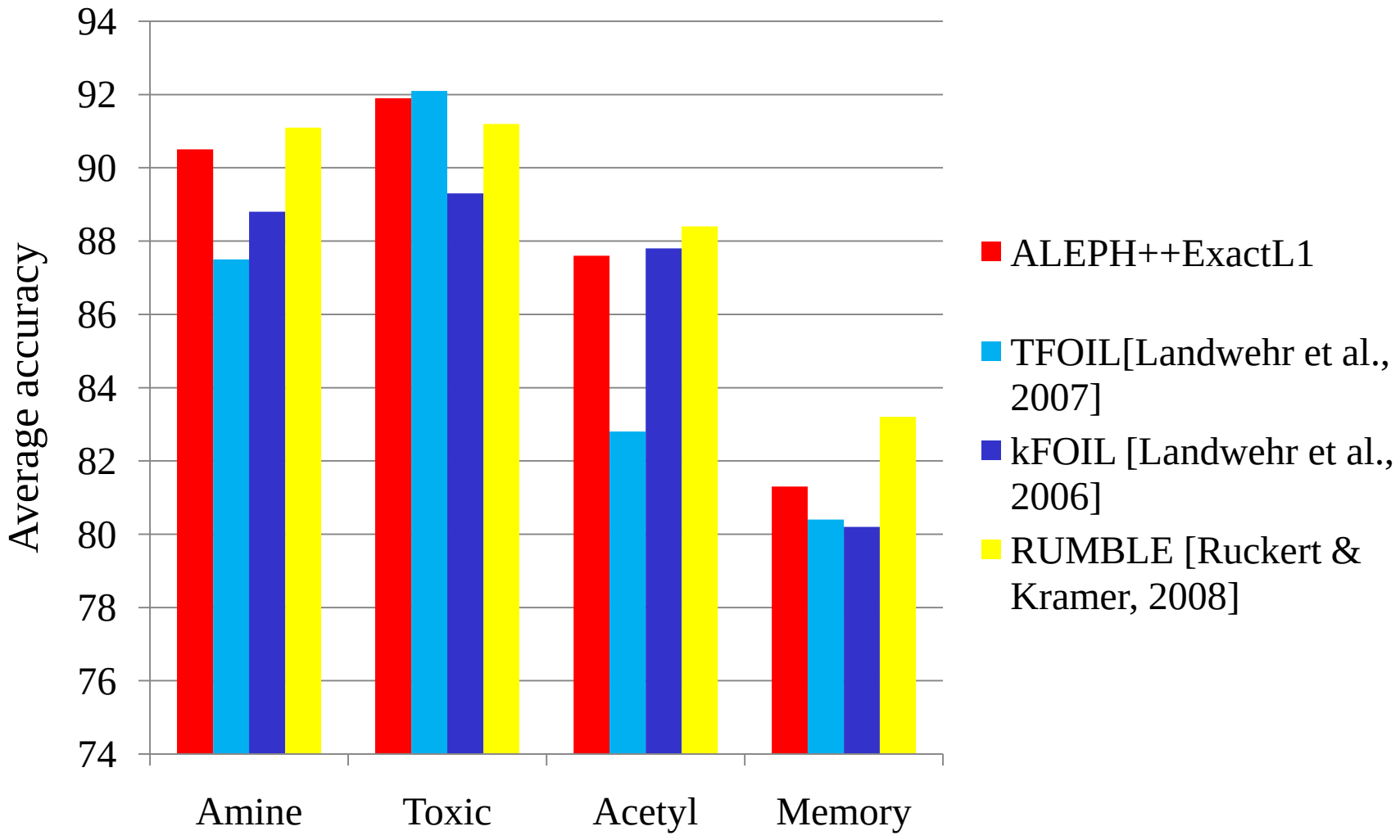


- Q4: The benefit of collective inference
  - Adding a transitive clause with infinite weight to the learned MLNs.

less_toxic(a,b) ^ less_toxic(b,c) => less_toxic(a,c).



- Q4: The performance of our approach against other "advanced ILP" methods



Conclusion

- Existing learning methods for MLNs fail on several benchmark ILP problems
- Our approach:
  - Use ALEPH++ for generating good candidate clauses
  - Use L1-regularization and exact inference to learn the weights for candidate clauses
- Our general approach can also be applied to other SRL models such as SLPs.
- Future work:
  - Integrate the discriminative structure and weight learning processes into one process

Thank you!
Questions?