Online Structure Learning for Markov Logic Networks

Tuyen N. Huynh and Raymond J. Mooney

Department of Computer Science
The University of Texas at Austin

ECML-PKDD-2011, Athens, Greece

Large-scale structured/relational learning

Citeseer Citation segmentation [Peng & McCallum, 2004]

D. McDermott and J. Doyle. Non-monotonic Reasoning I. Artificial Intelligence, 13: 41-72, 1980.

Craigslist ad segmentation [Grenager et al., 2005]

Modern, clean, quiet, $750 up--BIG pool, parking, laundry, elevator. Open viewing SAT/SUN, 10am-6pm, at 1720 12 Avenue, corner East 17 St. Other times call first: Sam, 510-534-0558.

Motivation

- Markov Logic Networks (MLNs) [Richardson & Domingos, 2006] are an elegant and powerful formalism for handling complex structured/relational data.
- All existing structure learning algorithms for MLNs are batch learning methods.
  - Effectively designed for problems that have a few "mega" examples.
  - Do not scale to problems with a large number of smaller structured examples.
- No existing online structure learning algorithms for MLNs.

The first online structure learner for MLNs

Outline

- ☑ Motivation
- Background
  - Markov Logic Networks
- OSL: Online structure learning algorithm
- Experiment Evaluation
- Summary

Background

Markov Logic Networks (MLNs)

[Richardson & Domingos, 2006]

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

10	InField(f,p1,c) ∧ Next(p1,p2) ⇒ InField(f,p2,c)
5	Token(t,p,c) ∧ IsInitial(t) ⇒ InField(Author,p,c) ∨ InField(Venue,p,c)

- Larger weight indicates stronger belief that the clause should hold.
- Probability of a possible world (a truth assignment to all ground atoms) x:

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

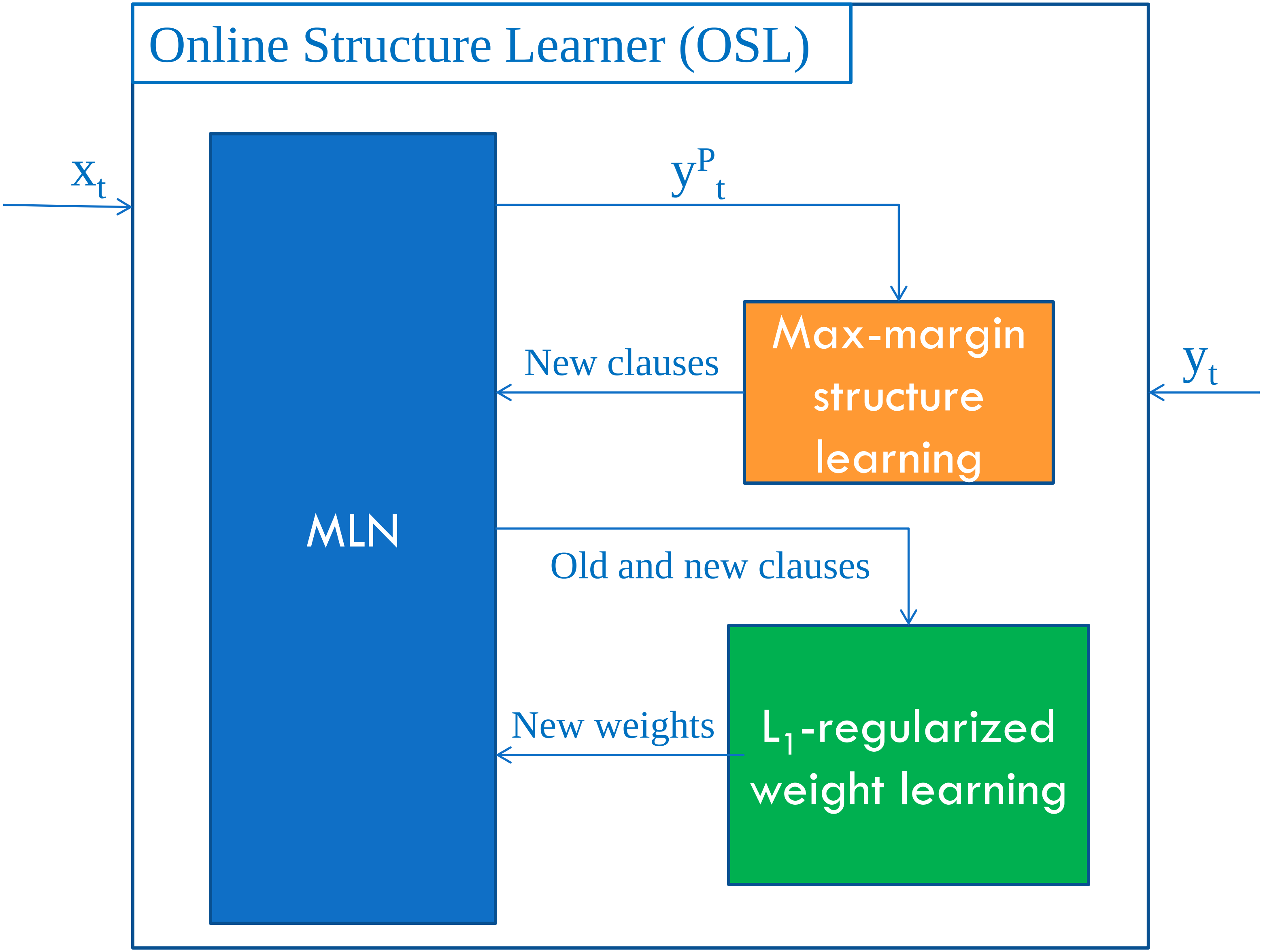
Weight of formula i

No. of true groundings of formula i in x

Existing structure learning methods for MLNs

- Top-down approach:
  - MSL[Kok & Domingos, 2005], DSL[Biba et al., 2008]
  - Start from unit clauses and search for new clauses
- Bottom-up approach:
  - BUSL[Mihalkova & Mooney, 2007], LHL[Kok & Domingos, 2009], LSM[Kok & Domingos , 2010]
  - Use data to generate candidate clauses

OSL: Online Structure Learner for MLNs



Online Structure Learner (OSL)
x_t
y^P_t
MLN
New clauses
Max-margin structure learning
y_t
Old and new clauses
New weights
L_1-regularized weight learning

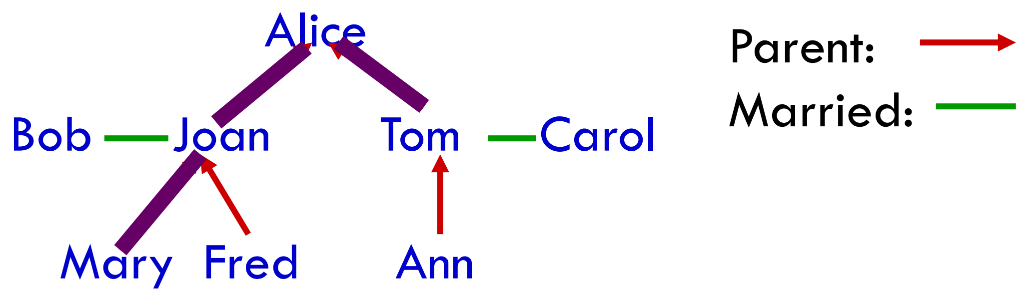
Max-margin structure learning

- Find clauses that discriminate the ground-truth possible world (x_t, y_t) from the predicted possible world (x_t, y_t^P)
  - Find where the model made wrong predictions
    $\Delta y_t = y_t \backslash \mathrm{y}_t^P$: a set of true atoms in y_t but not in y_t^P
  - Find new clauses to fix each wrong prediction in Δy_t
    - Introduce mode-guided relational pathfinding
      - Use mode declarations [Muggleton, 1995] to constrain the search space of relational pathfinding [Richards & Mooney, 1992]
  - Select new clauses that has more number of true groundings in (x_t, y_t) than in (x_t, y_t^P)
    - minCountDiff: $n_{nc}(x_t, y_t) - n_{nc}(x_t, y_t^P) \geq minCountDiff$

Relational pathfinding [Richards & Mooney, 1992]

- Learn definite clauses:
  - Consider a relational example as a hypergraph:
    - Nodes: constants
    - Hyperedges: true ground atoms, connecting the nodes that are its arguments
  - Search in the hypergraph for paths that connect the arguments of a target literal.

Uncle(Tom, Mary)



Parent(Joan,Mary) ∧ Parent(Alice,Joan) ∧ Parent(Alice,Tom) ⇒ Uncle(Tom,Mary)

Parent(x,y) ∧ Parent(z,x) ∧ Parent(z,w) ⇒ Uncle(w,y)

Relational pathfinding (cont.)

- We use a generalization of the relational pathfinding:
  - A path does not need to connect arguments of the target atom.
  - Any two consecutive atoms in a path must share at least one input/output argument.
- Similar approach used in LHL [Kok & Domingos, 2009] and LSM [Kok & Domingos , 2010].

→ Can result in an intractable number of possible paths

Mode declarations [Muggleton, 1995]

- A language bias to constrain the search for definite clauses.
- A mode declaration specifies:
  - The number of appearances of a predicate in a clause.
  - Constraints on the types of arguments of a predicate.

Mode-guided relational pathfinding

- Use mode declarations to constrain the search for paths in relational pathfinding:
  - Introduce a new mode declaration for paths, modep(r,p):
    - r (recall number): a non-negative integer limiting the number of appearances of a predicate in a path to r
      - can be 0, i.e don't look for paths containing atoms of a particular predicate
    - p: an atom whose arguments are:
      - Input(+): bound argument, i.e must appear in some previous atom
      - Output(-): can be free argument
      - Don't explore(.): don't expand the search on this argument

Mode-guided relational pathfinding (cont.)

- Example in citation segmentation: constrain the search space to paths connecting true ground atoms of two consecutive tokens
  - InField(field,position,citationID): the field label of the token at a position
  - Next(position,position): two positions are next to each other
  - Token(word,position,citationID): the word appears at a given position

modep(2,InField(.,–,.)) modep(1,Next(–, –)) modep(2,Token(.,+,.))

Mode-guided relational pathfinding (cont.)

Wrong prediction

InField(Title,**P09**,B2)

Hypergraph

P09 → {

Token(To,P09,B2),

Next(P08,P09),

Next(P09,P10),

LessThan(P01,P09)

…

}

Paths

{InField(Title,P09,B2),Token(To,P09,B2)}

Mode-guided relational pathfinding (cont.)

Wrong prediction

InField(Title,**P09**,B2)

Hypergraph

P09 → {
Token(To,P09,B2),
Next(P08,P09),
Next(P09,P10),
LessThan(P01,P09)
…
}

Paths

{InField(Title,P09,B2),Token(To,P09,B2)}
{InField(Title,P09,B2),Token(To,P09,B2),Next(P08,P09)}

Generalizing paths to clauses

Modes

modec(InField(c,v,v))
modec(Token(c,v,v))
modec(Next(v,v))
…

Paths

{InField(Title,P09,B2),Token(To,P09,B2),
Next(P08,P09),InField(Title,P08,B2)}
…

Conjunctions

InField(Title,p1,c) ∧ Token(To,p1,c) ∧ Next(p2,p1) ∧ InField(Title,p2,c)

Clauses

C1: ¬InField(Title,p1,c) ∨ ¬Token(To,p1,c) ∨ ¬Next(p2,p1) ∨ ¬ InField(Title,p2,c)

C2: InField(Title,p1,c) ∨ ¬Token(To,p1,c) ∨ ¬Next(p2,p1) ∨ ¬ InField(Title,p2,c)

Token(To,p1,c) ∧ Next(p2,p1) ∧ InField(Title,p2,c) ⇒ InField(Title,p1,c)

L_1-regularized weight learning

- Many new clauses are added at each step and some of them may not be useful in the long run.

→ Use L_1-regularization to zero out those clauses

- Use a state-of-the-art online L_1-regularized learning algorithm named ADAGRAD_FB [Duchi et.al., 2010], a L_1-regularized adaptive subgradient method.

Experiment Evaluation

- Investigate the performance of OSL on two scenarios:
  - Starting from a given MLN
  - Starting from an empty MLN
- Task: natural language field segmentation
- Datasets:
  - CiteSeer: 1,563 citations, 4 disjoint subsets corresponding 4 different research areas
  - Craigslist: 8,767 ads, but only 302 of them were labeled

Input MLNs

- A simple linear chain CRF (**LC_0**):
  - Only use the current word as features

    $$\text{Token}(+w,p,c) \Rightarrow \text{InField}(+f,p,c)$$

  - Transition rules between fields

    $$\text{Next}(p1,p2) \wedge \text{InField}(+f1,p1,c) \Rightarrow \text{InField}(+f2,p2,c)$$

Input MLNs (cont.)

- Isolated segmentation model (**ISM**) [Poon & Domingos, 2007], a well-developed MLN for citation segmentation :
  - In addition to the current word feature, also has some features that based on words that appear before or after the current word
  - Only has transition rules within fields, but takes into account punctuations as field boundary:

$\neg$HasPunc(p1,c) $\wedge$ InField(+f,p1,c) $\wedge$ Next(p1,p2) $\Rightarrow$ InField(+f,p2,c)

HasComma(p1,c) $\wedge$ InField(+f,p1,c) $\wedge$ Next(p1,p2) $\Rightarrow$ InField(+f,p2,c)

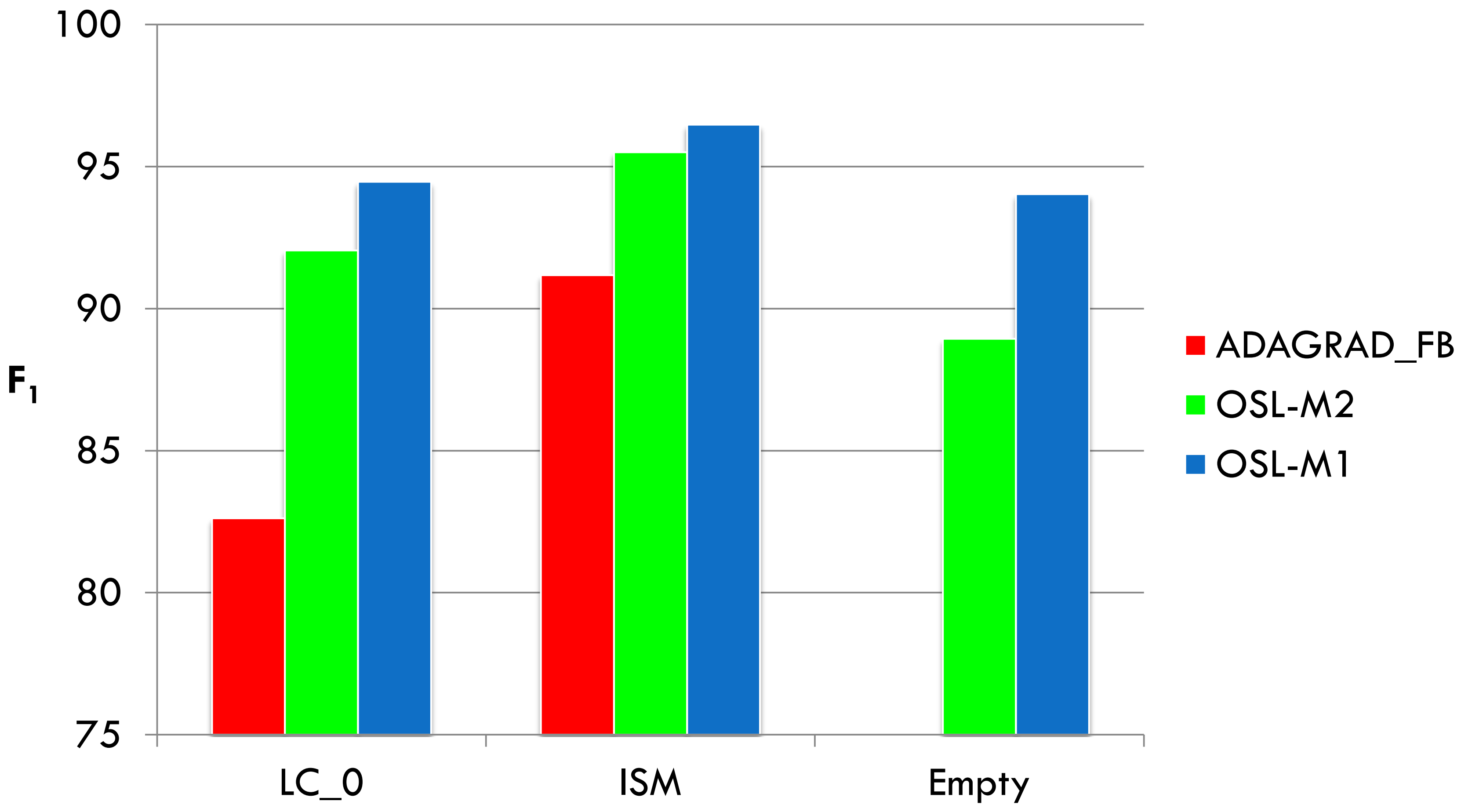
Systems compared

- **ADAGRAD_FB**: only do weight learning
- **OSL-M2**: a fast version of OSL where the parameter *minCountDiff* is set to 2
- **OSL-M1**: a slow version of OSL where the parameter *minCountDiff* is set to 1

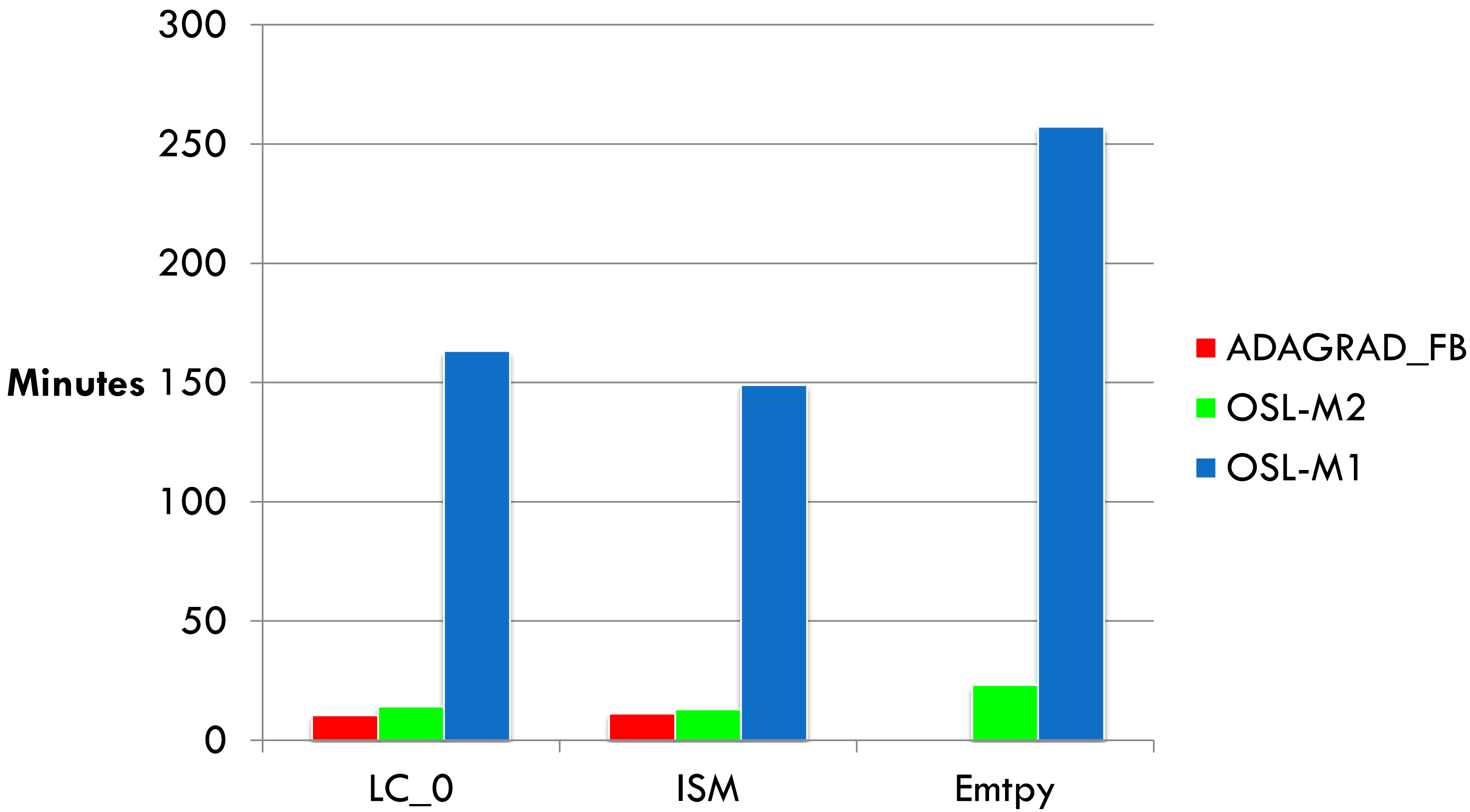
Experimental setup

- OSL: specify mode declarations to constrain the search space to paths connecting true ground atoms of two consecutive tokens:
  - A linear chain CRF:
    - Features based on current, previous and following words
    - Transition rules with respect to current, previous and following words
- 4-fold cross-validation
- Average F_1

Average F_1 scores on CiteSeer



Average training time on CiteSeer



Some good clauses found by OSL on CiteSeer

- OSL-M1-ISM:
  - The current token is a Title and is followed by a period then it is likely that the next token is in the Venue field

$$\mathrm{InField(Title,p1,c)} \wedge \mathrm{FollowBy(PERIOD,p1,c)} \wedge \mathrm{Next(p1,p2)} \Rightarrow \mathrm{InField(Venue,p2,c)}$$

- OSL-M1-Empty:
  - Consecutive tokens are usually in the same field

$$\mathrm{Next(p1,p2)} \wedge \mathrm{InField(Author,p1,c)} \Rightarrow \mathrm{InField(Author,p2,c)}$$

$$\mathrm{Next(p1,p2)} \wedge \mathrm{InField(Title,p1,c)} \Rightarrow \mathrm{InField(Title,p2,c)}$$

$$\mathrm{Next(p1,p2)} \wedge \mathrm{InField(Venue,p1,c)} \Rightarrow \mathrm{InField(Venue,p2,c)}$$

Summary

- The first online structure learner (OSL) for MLNs:
  - Can either enhance an existing MLN or learn an MLN from scratch.
  - Can handle problems with thousands of small structured training examples.
  - Outperforms existing algorithms on CiteSeer and Craigslist information extraction datasets.

Questions?

Thank you!