

Learning and Inference in Markov Logic Networks

December 1, 2009
CS 486/686
University of Waterloo

Outline

- Markov Logic Networks
 - Parameter learning
 - Lifted inference

CS486/686 Lecture Slides (c) 2009 P. Poupart

2

Parameter Learning

- Where do Markov logic networks come from?
- Easy to specify first order formulas
- Hard to specify weights due to unclear interpretation
- Solution:
 - Learn weights from data
 - Preliminary work to learn first-order formulas from data

CS486/686 Lecture Slides (c) 2009 P. Poupart

3

Parameter tying

- Observation: first-order formulas in Markov logic networks specify templates of features with identical weights
- Key: tie parameters corresponding to identical weights
- Parameter learning:
 - Same as in Markov networks
 - But many parameters are tied together

CS486/686 Lecture Slides (c) 2009 P. Poupart

4

Parameter tying

- Parameter tying \rightarrow few parameters
 - Faster learning
 - Less training data needed
- Maximum likelihood: $\theta^* = \operatorname{argmax}_{\theta} P(\text{data}|\theta)$
 - Complete data: convex opt., but no closed form
 - Gradient descent, conjugate gradient, Newton's method
 - Incomplete data: non-convex optimization
 - Variants of the EM algorithm

CS486/686 Lecture Slides (c) 2009 P. Poupart

5

Grounded Inference

- Grounded models
 - Bayesian networks
 - Markov networks
- Common property
 - Joint distribution is a product of factors
- Inference queries: $\Pr(X|E)$
 - Variable elimination

CS486/686 Lecture Slides (c) 2009 P. Poupart

6

Grounded Inference

- Inference query: $\Pr(\alpha|\beta)?$
 - α and β are first order formulas
- Grounded inference:
 - Convert Markov Logic Network to ground Markov network
 - Convert α and β into grounded clauses
 - Perform variable elimination as usual
- This defeats the purpose of having a compact representation based on first-order logic... **Can we exploit the first-order representation?**

CS4386/686 Lecture Slides (c) 2009 P. Poupart

7

Lifted Inference

- **Observation: first order formulas in Markov Logic Networks specify templates of identical potentials.**
- Question: can we speed up inference by taking advantage of the fact that some potentials are identical?

CS4386/686 Lecture Slides (c) 2009 P. Poupart

8

Caching

- **Idea: cache all operations on potentials to avoid repeated computation**
- Rational: since some potentials are identical, some operations on potentials may be repeated.
- Inference with caching: $\Pr(\alpha|\beta)?$
 - Convert Markov logic network to ground Markov network
 - Convert α and β to grounded clauses
 - Perform variable elimination with caching
 - Before each operation on factors, check answer in cache
 - After each operation on factors, store answer in cache

CS4386/686 Lecture Slides (c) 2009 P. Poupart

9

Caching

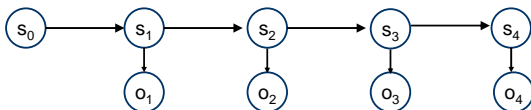
- How effective is caching?
- Computational complexity
 - Still exponential in the size of the largest intermediate factor
 - But, potentially sub-linear in the number of ground potentials/features
 - This can be significant for large networks
- Savings depend on the amount of repeated computation
 - Elimination order influences amount of repeated computation

CS4386/686 Lecture Slides (c) 2009 P. Poupart

10

Example: Hidden Markov Model

- Conditional distributions:
 - $\Pr(S_0), \Pr(S_{t+1}|S_t), \Pr(O_t|S_t)$
 - Identical factors at each time step



CS4386/686 Lecture Slides (c) 2009 P. Poupart

11

Hidden Markov Models

Markov Logic Network encoding

```

obs = { Obs1, ... , ObsN }
state = { St1, ... , StM }
time = { 0, ... , T }
  
```

```

State(state!, time)
Obs(obs!, time)
  
```

```

State(+s, 0)
State(+s, t) ^ State(+s', t+1)
Obs(+o, t) ^ State(+s, t)
  
```

CS4386/686 Lecture Slides (c) 2009 P. Poupart

12

State Prediction

- Common task: state prediction
 - Suppose we have a belief at time t : $\Pr(S_t | \mathcal{O}_{1:t})$
 - Predict state k steps in the future: $\Pr(S_{t+k} | \mathcal{O}_{1:t})$?
- $P(S_{t+k} | \mathcal{O}_{1:t}) = \sum_{S_t, \dots, S_{t+k-1}} P(S_t | \mathcal{O}_{1:t}) \prod_i P(S_{t+i+1} | S_{t+i})$
- In what order should we eliminate the state variables?

CS4386/686 Lecture Slides (c) 2009 P. Poupart

13

Common Elimination Orders

- Forward elimination
 - $P(S_{t+i+1} | \mathcal{O}_{1:t}) = \sum_{S_{t+i}} P(S_{t+i} | \mathcal{O}_{1:t}) P(S_{t+i+1} | S_{t+i})$
 - $P(S_{t+i} | \mathcal{O}_{1:t})$ is different for all i 's, so no repeated computation
- Backward elimination
 - $P(S_{t+k} | S_{t+i}) = \sum_{S_{t+i+1}} P(S_{t+k} | S_{t+i+1}) P(S_{t+i+1} | S_{t+i})$
 - $P(S_{t+k} | \mathcal{O}_{1:t}) = \sum_{S_t} P(S_{t+k} | S_t) P(S_t | \mathcal{O}_{1:t})$
 - $P(S_{t+k} | S_{t+i})$ is different for all i 's, so no repeated computation
- Any saving possible?

CS4386/686 Lecture Slides (c) 2009 P. Poupart

14

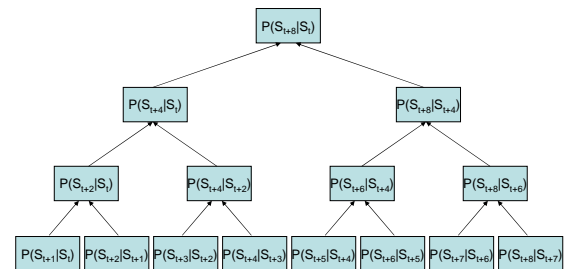
Pyramidal elimination

- Repeat until all variables are eliminated
 - Eliminate every other variable in order
- Example:
 - Eliminate $S_{t+1}, S_{t+3}, S_{t+5}, S_{t+7}, \dots$
 - Eliminate $S_{t+2}, S_{t+6}, S_{t+10}, S_{t+14}, \dots$
 - Eliminate $S_{t+4}, S_{t+12}, S_{t+20}, S_{t+28}, \dots$
 - Eliminate $S_{t+8}, S_{t+24}, S_{t+40}, S_{t+56}, \dots$
 - Etc.

CS4386/686 Lecture Slides (c) 2009 P. Poupart

15

Pyramidal elimination



CS4386/686 Lecture Slides (c) 2009 P. Poupart

16

Pyramidal elimination

- Observation: all operations at the same level of the pyramid are identical
 - Only one elimination per level needs to be performed
- Computational complexity:
 - $\log(k)$ instead of $\text{linear}(k)$

CS4386/686 Lecture Slides (c) 2009 P. Poupart

17

Automated elimination

- Question: how do we find an effective ordering automatically?
 - This is an area of active research
- Possible heuristic:
 - Before each elimination, examine operations that would have to be performed to eliminate each remaining variable
 - Eliminate variable that involves computation identical to the largest number of other variables (greedy heuristic)

CS4386/686 Lecture Slides (c) 2009 P. Poupart

18

Lifted Inference

- Variable elimination with caching still requires conversion of the Markov logic network to a ground Markov network, can we avoid that?
- Lifted inference:
 - Perform inference directly with first-order representation
 - Lifted variable elimination is an area of active research
 - Complicated algorithms due to first-order representation
 - Overhead due to the first-order representation often greater than savings in repeated computation
- Alchemy
 - Does not perform exact inference
 - Uses lifted approximate inference
 - Lifted belief propagation
 - Lifted MC-SAT (variant of Gibbs sampling)

Next Class

- Course wrap-up