Structured learning for sequence labeling
Part 3: Conditional Random Fields

Hakan Erdogan
Sabanci University

August 2-5, 2010
Enterface'10 workshop, Amsterdam

Outline

Generative vs Discriminative - HMM vs MEMM I

- Generative models (e.g. FLD) and logistic regression are generative-discriminative pairs
- MEMM is an attempt to get a discriminative version of HMM
- Depends on writing the conditional likelihood as

$$p(y_{1:T}|\mathbf{x}_{1:T}) = \prod_{t=1}^{T} p(y_t|y_{t-1}, \mathbf{x}_t)$$

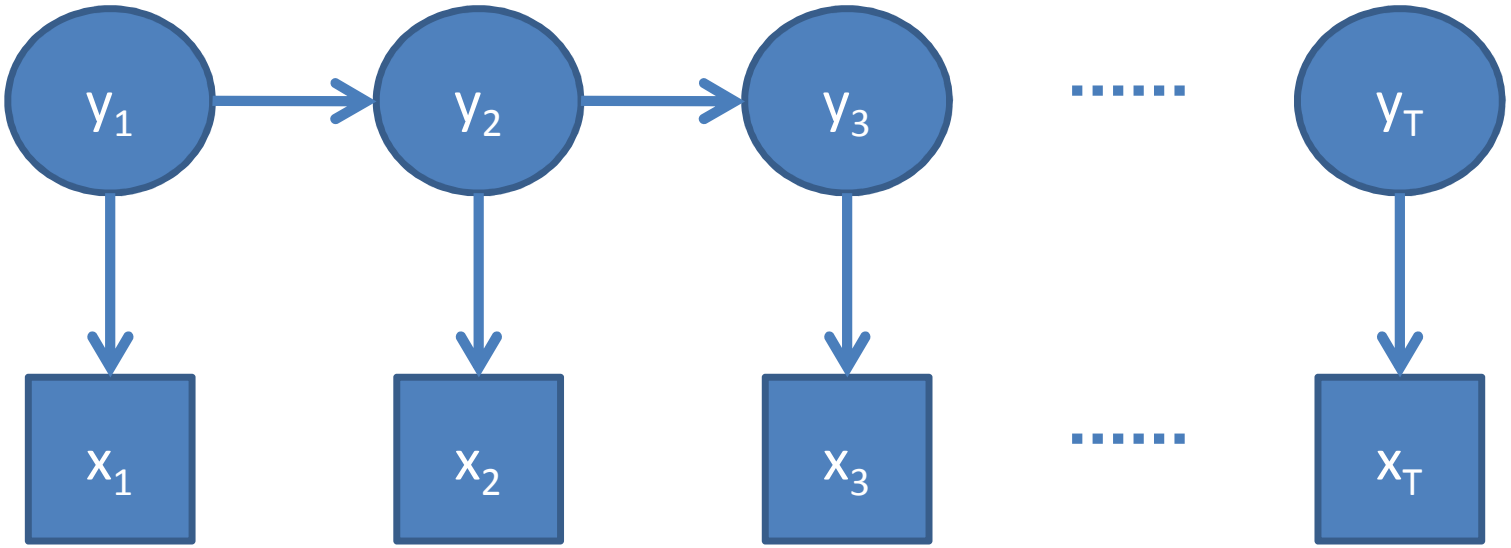
- This may not be a good assumption
- This turns out not to be a good way to obtain a discriminative model from HMM
- Leads to a problem called "label bias"

Generative vs Discriminative - HMM vs MEMM II

- Solution: Directly model $p(y_{1:T}|\mathbf{x}_{1:T})$ (conditional random fields)
  - without assuming probabilistic dependencies among y_t, y_{t-1} and $\mathbf{x}_t$
  - Directly use a log-linear model
  - But use only local features in the log-linear model that depend on y_t and y_{t-1} only! (to enable dynamic programming)

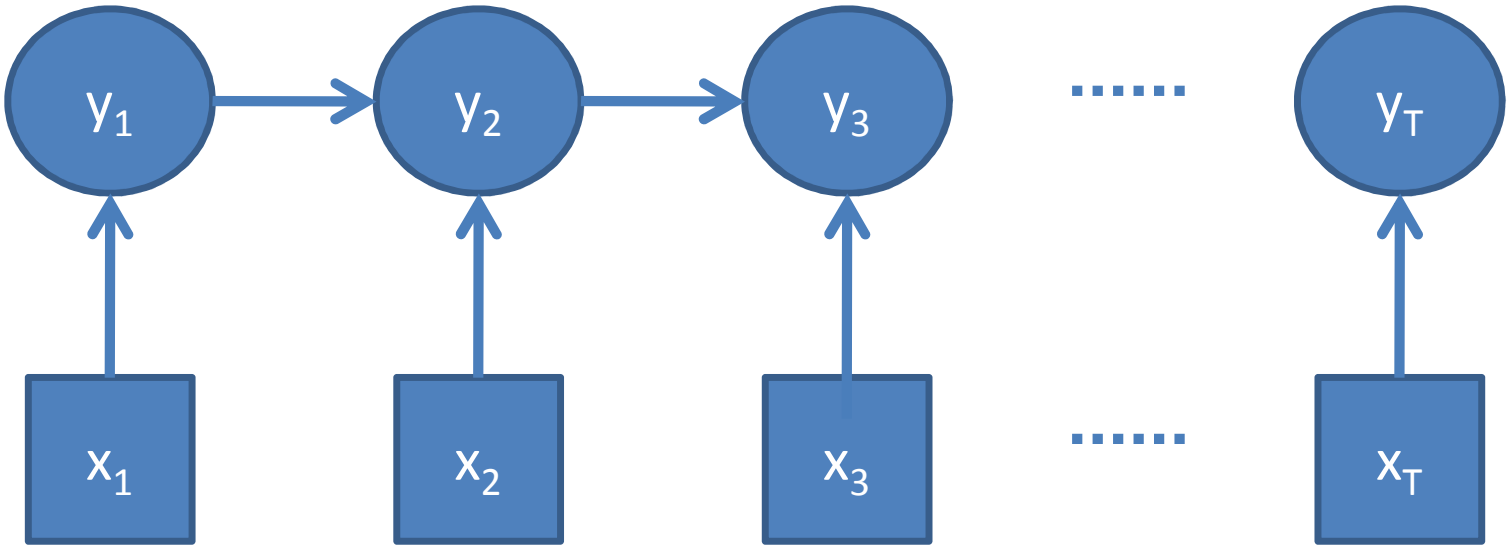
Graphical models I

HMM graphical model:



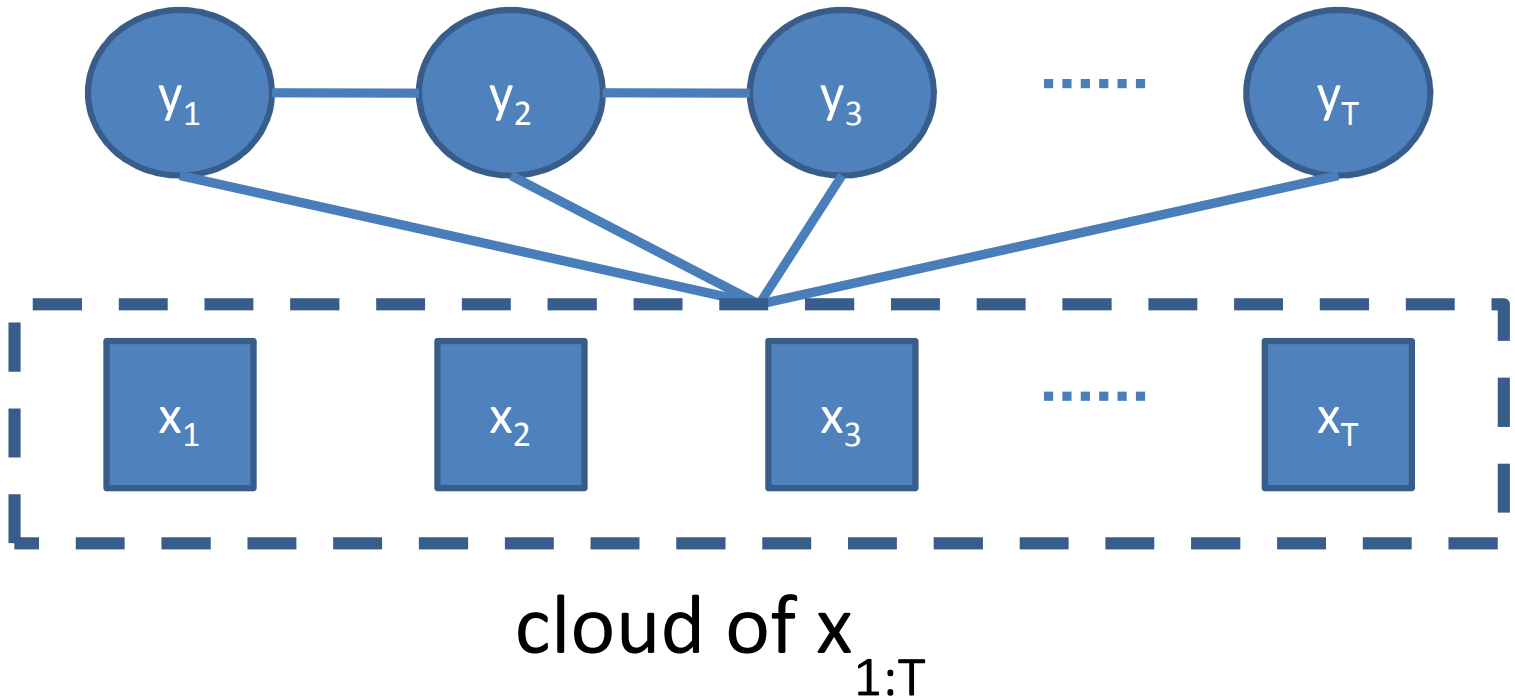
Graphical models II

MEMM graphical model:



Graphical models III

CRF graphical model:



Remembering problem setup I

- Given a sequence of features $\mathbf{x}_{1:T}$, find appropriate labels $y_{1:T}$ where each $y_t \in \mathcal{Y}$, we can assume wlog that $\mathcal{Y} = [M]$, a finite set
- This is a hard problem and the number of possible $y_{1:T}$ is too high, namely M^T and T is changeable
- We may need additional assumptions on output labels y_t, such as being Markov

Outline

Conditional Random Fields I

In CRF, we model the conditional probability of labels wrt observations as follows:

$$p(y_{1:T}|\mathbf{x}_{1:T}) = \frac{1}{Z(\mathbf{x}_{1:T}, \mathbf{w})} \exp\left\{\sum_{j=1}^{N_f} w_j F_j(\mathbf{x}_{1:T}, y_{1:T})\right\}$$

Key thing is to assume that the global feature functions $F_j(y_{1:T}, \mathbf{x}_{1:T})$ should be able to be written as a sum of local features

$$F_j(\mathbf{x}_{1:T}, y_{1:T}) = \sum_{t=1}^{T} f_j(y_{t-1}, y_t, \mathbf{x}_{1:T}, t)$$

- This assumption is necessary to be able to use dynamic programming algorithms in calculations
- Each local feature may depend on all $\mathbf{x}_{1:T}$ since they are given to us

Conditional Random Fields II

- This assumption yields a Markovian label sequence
- Local feature functions can specialize in depending on any combination of their inputs, they do not have to depend on all of their arguments
- For example, a transition feature will depend only on y_t and y_{t-1}

Problems of interest

To be able to solve inference problems in CRFs, we need to be able to compute the most likely label sequence:

$$y^*_{1:T} = \arg\max_{y'_{1:T}} p(y'_{1:T}|\mathbf{x}_{1:T}; \mathbf{w})$$

and for the learning problem, we need to calculate the partition function

$$Z(\mathbf{x}_{1:T}, \mathbf{w}) = \sum_{y'_{1:T}} \exp\left\{\sum_{j=1}^{N_f} w_j F_j(\mathbf{x}_{1:T}, y'_{1:T})\right\}$$

Note that direct calculation of these two quantities is highly expensive due to exponential amount of all possible $y_{1:T}$ that is needed to be considered

Finding the most likely labeling - Viterbi algorithm I

It is easy to show that:

$$y^*_{1:T} = \arg\max_{y'_{1:T}} \sum_j w_j F_j(\mathbf{x}_{1:T}, y'_{1:T})$$

and after expanding the features

$$y^*_{1:T} = \arg\max_{y'_{1:T}} \sum_j w_j \sum_{t=1}^{T} f_j(y'_{t-1}, y'_t, \mathbf{x}_{1:T}, t)$$

Let $g_t(y_{t-1}, y_t) = \sum_j w_j f_j(y_{t-1}, y_t, \mathbf{x}_{1:T}, t)$ to simplify notation. Define partial maximums:

$$V(y, t) = \max_{y'_{1:t-1}} \left(\sum_{\tau=1}^{t-1} g_\tau(y'_{\tau-1}, y'_\tau) + g_t(y'_{t-1}, y) \right)$$

Finding the most likely labeling - Viterbi algorithm II

- Clearly, this leads to a recursion:

$$V(y,t) = \max_{y'} \left(V(y', t-1) + g_t(y', y) \right)$$

- Similar to HMMs, we need to hold a backpointer to the maximizer label (state) after each time step
- We can view this procedure in a trellis
- In the end we can trace back from $V(y_T^*, T)$ to obtain the most likely label sequence $y_{1:T}^*$.

Forward-backward algorithm for CRFs I

- Remember that:

$$Z(\mathbf{x}_{1:T}, \mathbf{w}) = \sum_{y'_{1:T}} \exp\left\{ \sum_{j=1}^{N_f} w_j F_j(\mathbf{x}_{1:T}, y'_{1:T}) \right\}$$

- We need to sum over exponentially many sequence labelings which is impractical
- Similar to forward-backward algorithm in HMMs, we can perform a dynamic programming algorithm like that to compute Z

Forward-backward algorithm for CRFs II

- We need to use the local features summed over time to do that

$$Z(\mathbf{x}_{1:T}, \mathbf{w}) = \sum_{y'_{1:T}} \exp\left\{ \sum_{\tau=1}^{T} \sum_{j=1}^{N_f} w_j f_j(y'_{\tau-1}, y'_{\tau}, \mathbf{x}_{1:T}, \tau) \right\}$$

$$Z(\mathbf{x}_{1:T}, \mathbf{w}) = \sum_{y'_{1:T}} \prod_{\tau=1}^{T} G_\tau(y'_{\tau-1}, y'_{\tau})$$

where we define $G_t(y_1, y_2) = \exp g_t(y_1, y_2)$, and $g_t(y_1, y_2) = \sum_j w_j f_j(y_1, y_2, \mathbf{x}_{1:T}, t)$ as defined earlier

- and define partial sums up to time t

$$\alpha(y, t) = \sum_{y'_{1:t-1}} \left(\prod_{\tau=1}^{t-1} G_\tau(y'_{\tau-1}, y'_{\tau}) G_t(y'_{t-1}, y) \right)$$

Forward-backward algorithm for CRFs III

- We can update $\alpha(y, t)$ by the following forward recursion

$$\alpha(y, t) = \sum_{y'} \alpha(y', t-1) G_t(y', y)$$

- Similarly we define backward partial sums

$$\beta(y, t) = \sum_{y'_{t+1:T}} \left(G_{t+1}(y, y'_{t+1}) \prod_{\tau=t+1}^{T} G_{\tau+1}(y'_\tau, y'_{\tau+1}) \right)$$

- which can be updated with the backward recursion

$$\beta(y, t) = \sum_{y'} \beta(y', t+1) G_{t+1}(y, y')$$

Forward-backward algorithm for CRFs IV

- Clearly

$$Z(\mathbf{x}_{1:T}, \mathbf{w}) = \sum_{y'} \alpha(y', T) = \sum_{y'} \beta(y', 1)$$

- Note that if we use start and stop labels/states, we do not need the sums above and we get

$$Z(\mathbf{x}_{1:T}, \mathbf{w}) = \alpha(stop, T+1) = \beta(start, 0)$$

- Besides, we can calculate the following marginal posterior probabilities

$$p(y_t|\mathbf{x}_{1:T}) = \frac{\alpha(y_t, t)\beta(y_t, t)}{Z(\mathbf{x}_{1:T}, \mathbf{w})}$$

$$p(y_{t-1}, y_t|\mathbf{x}_{1:T}) = \frac{\alpha(y_{t-1}, t-1)G_t(y_{t-1}, y_t)\beta(y_t, t)}{Z(\mathbf{x}_{1:T}, \mathbf{w})}$$

Outline

CRF Training I

- We have seen that when $(\mathbf{x}_{1:T}, y_{1:T})$ were given, ML training for HMMs turned into simple counting
- For CRFs, even in that scenario, training is not that simple
- Consider conditional log-likelihood (CLL) for a single training sequence

$$\log p(y_{1:T}|\mathbf{x}_{1:T}; \mathbf{w}) = \mathbf{w}^T \mathbf{F}(\mathbf{x}_{1:T}, y_{1:T}) - \log Z(\mathbf{x}_{1:T}, \mathbf{w})$$

  where $\mathbf{F}$ denotes the vector of all N_f features
- For multiple training sequences, we need to sum the individual CLL's up

CRF Training II

- Gradient of the CLL for a single sequence is

$$\mathbf{F}(\mathbf{x}_{1:T}, y_{1:T}) - \sum_{y'_{1:T}} \mathbf{F}(\mathbf{x}_{1:T}, y'_{1:T}) p(y'_{1:T} | \mathbf{x}_{1:T})$$

$$\mathbf{F}(\mathbf{x}_{1:T}, y_{1:T}) - E_{y'_{1:T} \sim p(y'_{1:T} | \mathbf{x}_{1:T})} \left[\mathbf{F}(\mathbf{x}_{1:T}, y'_{1:T}) \right]$$

- When we obtain the maximizing $\mathbf{w}$, the gradient must be zero which corresponds to making the training data feature values to be the same as the expected values under the trained model

CRF Training III

- The expectation of a feature can be computed using the forward and backward variables as follows:

$$
\begin{aligned}
& E_{y'_{1:T} \sim p(y'_{1:T}|\mathbf{x}_{1:T})} \left[F_j(\mathbf{x}_{1:T}, y'_{1:T}) \right] \\
= \quad & E_{y'_{1:T} \sim p(y'_{1:T}|\mathbf{x}_{1:T})} \left[\sum_{t=1}^{T} f_j(y'_{t-1}, y'_t, \mathbf{x}_{1:T}, t) \right] \\
= \quad & \sum_{t=1}^{T} E_{y'_{t-1}, y'_t} [f_j(y'_{t-1}, y'_t, \mathbf{x}_{1:T}, t)] \\
= \quad & \frac{1}{Z} \sum_{t=1}^{T} \sum_{y_1, y_2} \alpha(t-1, y_1) f_j(y_1, y_2, \mathbf{x}_{1:T}, t) G_t(y_1, y_2) \beta(t, y_2)
\end{aligned}
$$

where $G_t(y_1, y_2) = \exp\{\sum_{j'} w_{j'} f_{j'}(y_1, y_2, \mathbf{x}_{1:T}, t)\}$

CRF Training IV

- The exact calculation of the expected value can be somewhat computationally complex
- It is possible to approximate the gradient calculation by
  - Considering only the best competitor's feature function instead of considering the average over all (expected value)
  - Performing Gibbs sampling to evaluate the expected value
- Regularized empirical risk (conditional log-likelihood plus a regularizing penalty term) is optimized in the case of CRFs as well
- Dropping dependence on $1 : T$, given training data (x^i, y^i) for $i \in [N]$ where each (x^i, y^i) is a training sequence, we get the following RER function to minimize

$$\sum_{i=1}^{N} \left(-\mathbf{w}^T \mathbf{F}(x^i, y^i) + \log Z(x^i, \mathbf{w}) \right) + \Omega(\mathbf{w})$$

CRF Training V

- and each entry of the gradient vector is

$$\sum_{i=1}^{N} \left(-F_j(x^i, y^i) + \sum_{y'} p(y'|x^i; \mathbf{w}) F_j(x^i, y') \right) + \frac{\partial \Omega}{\partial w_j}$$

Optimization methods

- Almost all methods require computation of the gradient whose exact computation requires forward-backward iterations, but this can be approximated through methods discussed above
  1. Iterative scaling (old one, slow)
  2. Conjugate gradient method
  3. L-BFGS (a Quasi Newton method)
  4. Stochastic gradient method: update parameters by moving in the direction of the gradient of one sequence at a time (easy and fast converging)

Stochastic Gradient Updates

- Stochastic gradient update for a single weight w_j is as follows:

$$w_j := w_j + k \left(F_j(x^i, y^i) - \sum_{y'} p(y'|x^i, \mathbf{w}) F_j(x^i, y') - \frac{\partial \Omega}{\partial w_j} \right)$$

  where k is a variable learning rate parameter (step size in the gradient direction)
- Usually k is chosen to decrease inversely proportional to the iteration number

For more information

- I have used the following papers/documents for this talk, it is beneficial to explore them all
- Papers and technical reports [Lafferty et al., 2001, Sutton and McCallum, 2006, Elkan, 2008, Gupta, 2005, Memisevic, 2006]
- See video lecture by Prof. Charles Elkan on videolectures.net

Max-margin training of CRFs I

- Also known as Structural SVM or max-margin Markov networks
- First proposed in [Altun et al., 2003, Taskar et al., 2003]
- We can formulate max-margin problems for estimating feature weights $\mathbf{w}$
- Problems can be formulated using the RER framework
- Enables using label-losses in margin rescaling or slack rescaling form
- [Tsochantaridis et al., 2005] proposes a cutting plane algorithm to solve the max-margin problem
  - Avoids forward backward
  - Only Viterbi algorithm needed to find the most offending label sequence
  - Multiple SVM problems need to be solved each time adding a new constraint
  - A recent single-slack formulation [Joachims et al., 2009] is even faster

Max-margin training of CRFs II

- [Taskar et al., 2003] proposes elegant reformulation of the problem as small linear programming problems, but may not scale too well to larger number of examples
- SVM-softmax approximation in the max-margin problem would give similar form as CRF training (and would require to compute the marginal probabilities) but enables using label-loss functions which was not possible in CRFs
- Need more detailed discussion

Toolkits

- Hidden markov models
  - HTK by Cambridge (Young et.al.)
  - Sphinx, Julius
  - Matlab statistics toolbox implements discrete HMMs
- Condition random fields
  - CRF++ (for NL problems)
  - CRFSGD
  - Mallet
- Max-margin structured learning
  - Svm-struct (includes SVM for sequence labeling) by Joachims

THANK YOU!

THANK YOU!
Questions?

References I

Y Altun, I Tsochantaridis, and T Hoffman. Hidden Markov support vector machines. In *International Conference on Machine Learning*, 2003.

Charles Elkan. Log-linear models and conditional random fields, notes for a tutorial at CIKM'08, 2008.

R Gupta. Conditional random fields. Technical report, IIT Bombay, 2005.

T Joachims, T Finley, and C N J Yu. Cutting-plane training of structural SVMs. *Machine Learning*, 77(1):27–59, October 2009.

J Lafferty, A McCallum, and F Pereira. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*, 2001.

R Memisevic. An introduction to structured discriminative learning. Technical report, University of Toronto, 2006.

References II

C Sutton and A McCallum. An Introduction to Conditional Random Fields for Relational Learning. In Lise Getoor and Ben Taskar, editors, *Introduction to statistical relational learning*. MIT Press, 2006.

Ben Taskar, Carlos Guestrin, and Daphne Koller. Max-margin Markov Networks. In *NIPS*, 2003.

I Tsochantaridis, T Joachims, T Hoffman, and Y Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–84, December 2005.