

# CS281A/Stat241A recitation 9: modeling, HMM relatives, CRFs

October 29, 2007

Percy Liang

## Revisiting HMMs

- Setup
  - Observations:  $\mathbf{x} = (x_1, \dots, x_T)$
  - Latent variables:  $\mathbf{z} = (z_1, \dots, z_T)$
  - Assume  $K$  states ( $z_t \in \{1, \dots, K\}$ ) and  $V$  observations ( $x_t \in \{1, \dots, V\}$ )
  - Parameters:  $\theta = (A, \eta)$ ,
    - \* Transitions  $A = \{a_{zz'} : 1 \leq z, z' \leq K\}$ :  $p(z_t | z_{t-1}) = a_{z_{t-1}, z_t}$
    - \* Emissions  $\eta = \{\eta_{zx} : 1 \leq z \leq K, 1 \leq x \leq V\}$ :  $p(x_t | z_t) = \eta_{z_t, x_t}$
  - Goal:  $\max_{\theta} p(\mathbf{x}; \theta)$
- We will derive the EM algorithm for HMMs intuitively in three stages. This methodology will allow us to quickly see the inference algorithm for a variety of models similar to the HMM without having to go through a formal derivation every time.
- Stage 1:
  - Assume that the latent variables are observed. The standard maximum likelihood estimate can be computed in closed form:

$$p(z' | z) = a_{z, z'} \leftarrow \frac{C_a(z, z')}{C_a(z, *)}$$

$$p(x | z) = \eta_{z, x} \leftarrow \frac{C_{\eta}(z, x)}{C_{\eta}(z, *)}$$

where

$$C_A(z, z') = \sum_{t=1}^T \mathbb{1}[z_{t-1} = z, z_t = z']$$

$$C_{\eta}(z, x) = \sum_{t=1}^T \mathbb{1}[z_t = z, x_t = x]$$

Intuitively, think of all the places in the model where a multinomial parameter is being used. Count the number of times that happens and normalize these counts to produce transition and emission probabilities.

- Stage 2 (Viterbi EM):

- When the latent variables are unobserved (latent), so we need to fill them in somehow using the current parameters that we have (this has a bootstrapping flavor).
- How do we fill them in? One natural possibility is to find the best sequence under the model (think k-means):

$$\mathbf{z}^* = \underset{\mathbf{z}}{\operatorname{argmax}} p(\mathbf{z} \mid \mathbf{x})$$

and then use counts based on that sequence.

$$C_A(z, z') = \sum_{t=1}^T \mathbb{1}[z_{t-1}^* = z, z_t^* = z'].$$

- Then we alternate between filling in  $z$  (the E-step) and optimizing  $\theta$  (the M-step). This algorithm can be viewed as coordinate-wise ascent of an appropriate objective. The objective function is nonconvex and we are subject to local optima.

- Stage 3 (EM):

- In stage 2, we are in some sense not using the model to its fullest because the model is capable of giving us an entire posterior distribution over  $p(\mathbf{z} \mid \mathbf{x})$ , not just a best  $\mathbf{z}^*$ .
- To use the model to its fullest, we consider creating all possible sequences  $\mathbf{z}_1, \dots, \mathbf{z}_{K^T}$ , each  $\mathbf{z}$  weighted by its posterior probability under the current parameters  $p(\mathbf{z} \mid \mathbf{x}; \theta)$ .
- We pretend we have  $p(\mathbf{z} \mid \mathbf{x}; \theta)$  occurrences of the fully observed  $(bx, \mathbf{z})$  (never mind that these are not integers). Then the counts should be:

$$C_A(z, z') = \sum_{\mathbf{z}} p(\mathbf{z} \mid \mathbf{x}; \theta) \sum_{t=1}^T \mathbb{1}[z_{t-1} = z, z_t = z'] = \sum_{t=1}^T p(z_{t-1} = z, z_t = z' \mid x).$$

### An HMM with one transition parameter

- Replace full transition matrix  $A$  with  $0 \leq \epsilon \leq 1$  and define

$$p(z_t = z' \mid z_{t-1} = z) = \begin{cases} \frac{\epsilon}{K-1} & \text{if } z \neq z' \\ 1 - \epsilon & \text{if } z = z' \end{cases}$$

- This model illustrates two things:

- We have fewer parameters now: as a result, it's easier to estimate the parameters but the model is less expressive. This tradeoff shows up a lot in machine learning.
- The graphical model structure does not tell us everything about the model. This HMM is parameterized very different from the ordinary HMM.

- M-step update:

$$\epsilon \leftarrow \frac{\sum_{t=1}^T \mathbb{1}[z_t \neq z_{t-1}]}{T}$$

Note that the sufficient statistics of the model are  $T(z, x) = \sum_{t=1}^T \mathbb{1}[z_t \neq z_{t-1}]$ .

- E-step: compute the expected sufficient statistics  $p(z_t \neq z_{t-1} | x)$ . These can be computed easily from  $p(z_{t-1}, z_t | x)$ , which is what the ordinary E-step of an HMM model yields.

## Applications of latent-variable models

- Density modeling
  - Recall our objective function:  $\max_{\theta} p(x; \theta)$ ; can use for anomaly detection.
  - Here, latent variables are just a means, not an end for the model.
- Prediction (unsupervised learning)
  - Naive Bayes for clustering (unsupervised classification):
    - \* Input is  $\mathbf{x} = (x_1, \dots, x_D)$ , output is  $y \in \{1, \dots, K\}$
    - \* The model is:  $p(\mathbf{x}, y) = p(y) \prod_{d=1}^D p(x_d | y)$
    - \* Normally, a Naive Bayes model is used for (supervised) classification.
    - \* Estimate the parameters given labeled data, we maximize the joint likelihood  $p(\mathbf{x}, y)$ .
    - \* But now, we have a way to estimate parameters when some variables in the model are not observed, so we could maximize  $p(\mathbf{x})$ , which does not require observing  $y$ . In general, this type of unsupervised learning is too good to be true. For example, many classification problems with different  $p(y | \mathbf{x})$  could have the same input distribution  $\mathbf{x}$ . How can the model read your mind?
    - \* But sometimes, if the model is designed properly, the latent states that are learned do have something to do with your labels. Sometimes the latent states can be interpretable, but not generally.
  - HMM for tagging:
    - \* Part-of-speech tagging: have sequence of words  $(x_1, \dots, x_T)$ , want to recover sequence of part-of-speech tags  $(y_1, \dots, y_T)$ .
    - \* What results? Latent states are a combination of syntax and semantics. 65% accuracy (contrast with supervised learning, which gets 95% accuracy).

- HMM for segmentation: Encode segmentation as a tag sequence: beginning of state or not

## Why use latent variables?

- Reinterpret the model to make parameter estimation friendlier
  - Two ways to look at it
    - \* No latent variables: complex interactions between observed variables
    - \* Latent variables: simple interactions between all variables
  - Of course we could use generic non-linear optimization methods but EM allows us to use the tools we already have (closed form optimization for the M-step)
  - Example: noisy-or model on the problem set, mixture models
- Increasing model capacity
  - We can have rich interdependencies between all the observed variables without introducing lots of new parameters, as the following example demonstrates.
  - Logistic regression: Assume the decision boundary is linear

$$p(x = 1 | x) = \frac{1}{1 + e^{-\beta^T x}}$$

Linearity could be a bad assumption.  $D$  parameters.

- Mixture of linear regression models:

$$p(x = 1 | x) = \sum_{q=1}^K \alpha_q \frac{1}{1 + e^{-\beta_q^T x}}$$

Is more expressive. Advantage: only  $KD$  parameters. Disadvantage: nonconvex optimization.

- Aside: another way to make the model more expressive is to introduce quadratic terms  $xx^T$ , but that involves  $D^2$  parameters; this optimization procedure is convex though. Advantage: convex optimization. Disadvantage:  $D^2$  parameters.

## Designing models

- We have talked about parameter estimation of latent variable models (use the EM algorithm). How do we get the model in the first place? That's not at all straightforward, and a bit of a black art.
  - All we have is the data to work with. So first, come up with a generative story (a graphical model) of how your data got there.

- The graphical model automatically imposes certain independence assumptions. Check if these are reasonable assumptions.
- Example: suppose we are modeling English sentences.
- Markov model
 
$$p(x_1, \dots, x_T) = \prod_{t=1}^T p(x_t | x_{t-1})$$
  - (Conditional) independence says that  $x_{t-1}$  and  $x_{t+1}$  are independent given  $x_t$ . This is a bad assumption:
    - \* Example: “the purpose of” versus “to purpose these”; clearly, the distribution of words after “purpose” depends on the word preceding “purpose.”
  - Can alleviate the brokenness of these independence assumptions using  $k$ -th order Markov model, but then we have  $O(V^{k+1})$  parameters, which is a lot.
  - Parameter estimation is easy: normalize counts (no latent variables)
- HMMs:
  - We no longer have Markov independence assumptions (marginalizing out the hidden states, all observations are coupled), but we make other assumptions. We often need fewer parameters when we use latent variable models.
  - Caveat: when designing models with latent variables, it’s tempting to interpret the variables as semantically meaningful. While it’s a useful for intuition, it can also be dangerous, because the algorithm is just going to look at the data and re-purpose latent variables for something else.
  - Parameter estimation: EM (Baum-Welsh)
- Factorial HMMs
  - HMM model is too coarse, meaning it’s either too simple (if have few latent states) or too complex (if have many latent states).
  - In sentences, there are many “signals” happening at the same time interacting to produce the sentence: syntax, topics, etc.
  - Parameter estimation
    - \* Sufficient statistics: transition and emission counts
    - \* EM with junction-tree
- Coupled HMMs
  - Observations: an English sentence and a French sentence which are translations of each other.

- We could model them separately using two HMMs. Independence violated: the sentences are translations of each other.
- We could use the same latent sequence for both sentences. Model is too coarse: it's either too simple (if have few latent states) or too complex (if have many latent states).
- Solution: introduce two latent sequences but couple them.
- But then if we have  $m$  sequences, we have to introduce massive coupling between the  $m$  sequences, but number of parameters grows exponentially with  $m$ .
- An alternative model is to introduce a mother latent variable for each time slice  $t$ , which generates the  $m$  latent variables independently, which in turn generate the  $m$  observed data points at time  $t$ .
- Parameter estimation: EM with junction tree for E-step
- Dynamic Bayesian networks: all of these models we have talked about so far fall under the umbrella of DBNs.
- Phylogenetic trees
  - Start out with a tree model on each DNA site.
  - Different sites change at different rates, so there's a latent variable for that.
  - Phylo-HMM: But there is dependence between the rates of adjacent sites, so attach Markov dependencies.

### Conditional random fields (CRFs)

- Let's work in the supervised scenario (there are no latent variables).
- Generative models: Naive Bayes  $\rightarrow$  Hidden Markov models (where  $\mathbf{z}$  is observed)
- Discriminative models: logistic regression  $\rightarrow$  conditional random fields
- Generative versus discriminative
  - Input:  $\mathbf{x} = (x_1, \dots, x_T)$
  - Output:  $\mathbf{y} = (y_1, \dots, y_T)$
  - Generative:  $\max_{\theta} p(\mathbf{x}, \mathbf{y})$  (HMM with observed  $\mathbf{z}$ )
  - Discriminative:  $\max_{\theta} p(\mathbf{y} | \mathbf{x})$  (CRF)
- All these are exponential family models, so let's define the CRF in terms of exponential families:

$$p(\mathbf{y} | \mathbf{x}; \theta) = \exp\{\theta^T T(\mathbf{x}, \mathbf{y}) - A_{\mathbf{x}}(\theta)\},$$

where the log-partition function depends on  $\mathbf{x}$  and sums over only  $\mathbf{y}$ :

$$A_{\mathbf{x}}(\theta) = \log \sum_{\mathbf{y}} \exp\{\theta^T T(\mathbf{x}, \mathbf{y})\}$$

What makes it a CRF is that all components of  $T(\mathbf{x}, \mathbf{y})$  (the features) conform to the linear graph structure defined by  $\mathbf{y}$ , as in an HMM. For example, features could be of the form  $\sum_{t=1}^T \mathbb{1}[y_{t-1} = y, y_t = y']$  and  $\sum_{t=1}^T \mathbb{1}[y_t = y, x_t = x]$ , which are exactly the ones for an HMM. Differences: (1) we are maximizing a discriminative objective and (2) there are no constraints on  $\theta$  while there are sum-to-one constraints for  $A$  and  $\eta$ .

- Parameter estimation

- Gradient: 
$$\underbrace{T(\tilde{\mathbf{x}}, \tilde{\mathbf{y}})}_{\text{empirical sufficient statistics}} - \underbrace{\mathbb{E}_{\theta} T(\mathbf{x}, \mathbf{y})}_{\text{expected sufficient statistics}}$$

- Use any iterative optimization procedure. Objective is convex, so converge to global optimum. More computationally expensive than estimating HMM parameters (when  $\mathbf{z}$  is observed).
- How to compute  $\mathbb{E}_{\theta} T(\mathbf{x}, \mathbf{y})$  where the expectation is taken with respect to the posterior  $p(\mathbf{y} | \mathbf{x})$ . For features of the type  $\sum_{t=1}^T \mathbb{1}[y_{t-1} = y, y_t = y']$ , this just requires computing  $p(y_{t-1} = y, y_t = y' | \mathbf{x})$ , the exact same computation required for the E-step of HMM training!