10-708: Probabilistic Graphical Models 10-708, Spring 2018

10 : HMM and CRF

Lecturer: Kayhan Batmanghelich *Scribes: Ben Lengerich, Michael Kleyman*

1 Case Study: Supervised Part-of-Speech Tagging

We will examine the applications of Hidden Markov Model, Conditional Random Field, and Maximum Entropy Markov Model in the context of a supervised part-of-speech tagging. In this NLP task, we are given data of the form $\mathcal{D} = \{x^{(n)}, y^{(n)}\}_{n=1}^{N}$ where $x^{(n)}$ is a sequence of words and $y^{(n)}$ is a sequence of tags that correspond to parts of speech of the same length as $x^{(n)}$. Each sentence begins with the start token y_0. Our goal is to train a system which will automatically tag the parts of speech in new sentences.

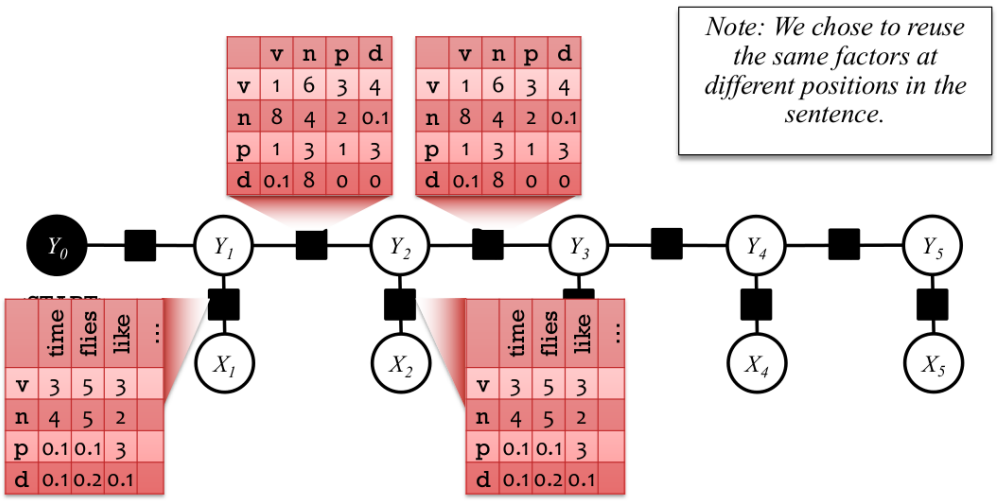

Figure 1: Factors for part-of-speech tagging.

As shown in Figure 1, we can construct a graph with two sets of factors: those between tags and words, and those between tags. To reduce the number of parameters, all members of the same set will reuse the same parameters. From these local opinions, the global probability can be calculated as the normalized product of local opinions:

$$p(n, v, p, d, n, time, flies, like, an, arrow) = \frac{1}{Z}(4 * 8 * 5 * 3 \ldots)$$

where Z is a normalization constant. Similarly, the probability of a sentence can be calculated by summing over all possible tag sequences. If this model is a Markov Random Field (MRF), the individual factors aren't necessarily probabilities and we must calculate the normalization constant Z separately, which would equal the sum of tag products of all possible permutations of tags for a sentence . If instead the model is a Bayesian Network (BN), then the factors represent conditional probabilities and each row in the factor table already sums to one. To look up a conditional probability in a factor table of a discrete case: $P(y_2|y_1)$, would equal the value in the column corresponding to y_2 and the row corresponding to y_1. In this case, $Z = 1$ and the

joint probability is simply:

$$p(n, v, p, d, n, time, flies, like, an, arrow) = (0.3 * 0.8 * 0.2 * 0.5 * \ldots)$$

1.1 Hidden Markov Model

A Hidden Markov Model (HMM) makes the Markov assumption that the probability of the current is only dependent on the state before it:

$$P(x_{1:n}, y_{1:n}) = \prod_{i=1}^{n} P(x_i|y_i)P(y_i|y_{i-1})$$

However, HMM models capture dependencies between each state and only its corresponding observation. This misses many valuables dependencies; for example, in a sentence segmentation task, each segmental state may depend not just on a single word (and the adjacent segmental stages), but also on the (non-local) features of the whole line.

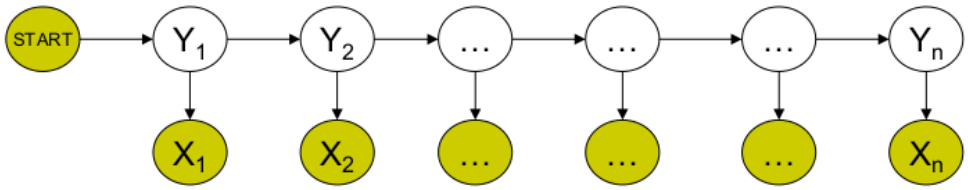

Figure 2: The structure of an HMM.

Furthermore, there is a mismatch between the learning objective function and predictive objective function. The HMM learns a joint distribution of states and observations $P(Y, X)$ but in a prediction task, we are only interested in the conditional probability $P(Y|X)$ which is more computationally tractable to estimate.

1.2 Forward-Backward and Vertbi Algorithm

For details of the Forward-Backward Algorithm and Viterbi, see Notes from Feb 1. The Forward-Backward Algorithm is also known as the sum product algorithm for HMMs and computes marginals of cliques of variables. The Viterbi algorithm is also known as the max product algorithm for HMMs and computes that maximum assignment probability.

Aside: The message passing algorithm is a generalization of the forward backward algorithm and Vertbi, but we refer to the Forward-Backward Algorithm specifically for HMM, because it was published first

1.3 Conditional Random Field

What if our goal is simply to label a test sentence with parts-of-speech tags, rather than generating new sentences, why do we need to model the joint probability? This seems like wasted effort and leads us to consider conditional random fields. Unlike the HMM in a conditional random field formulation, we are concerned with simply modeling the conditional distribution over tags Y_i given words x_i, a task which computationally more tractable. The factors and Z are now specific to the sentence, and variables are "clamped" to their value in that particular sentence (Figure 3). This is equivalent to multiplying in an

"evidence potential" which is a point mass with all its weight on $X_i = x_i$. This is similar to the familiar concept in linear regression in which we assume $Y = X\beta + \epsilon$ and seek only to estimate $p_\theta(Y|X)$ rather than $p_\theta(Y, X)$.

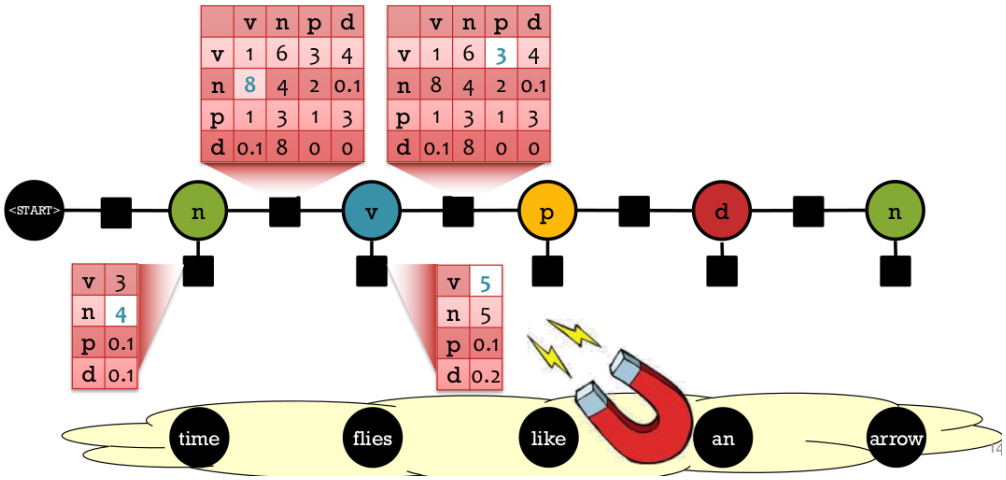

Figure 3: A CRF for part-of-speech tagging.

To identify parts-of-speech tags, our model must examine the context of each word (each word alone may have multiple applicable part-of-speech tags) using the forward-backward algorithm.

1.4 Maximum Entropy Markov Model

We can also overcome the problems seen in an HMM model by the Maximum Entropy Markov Model (MEMM), shown in Figure 4.

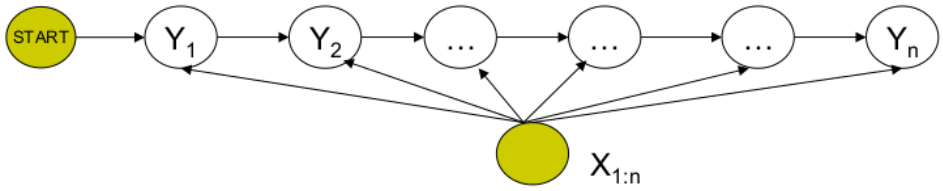

Figure 4: The structure of a MEMM.

In a MEMM,, and the entire word sequence is has an arrow to each node, rather than a single word as in the HMM. Because the directionality is reversed, the arrows no longer imply causality. Like the CRF introduced previously the variables are "clamped" to their value in that particular sentence. This part of the model can be viewed as regression from the sentence to the tags. Like an HMM however, there are still arrows between each of the tag variables. Finally, the overall normalizer is replaced by normalizing each factor:

$$P(y_{1:n}|x_{1:n}) = \prod_{i=1}^{n} P(y_i|y_{i-1}, x_{1:n}) = \prod_{I=1}^{n} \frac{exp(w^T f(y_i, y_{i-1}, x_{1:n})}{Z(y_{i-1}, x_{1:n})}$$

This is a discriminative model which saves modeling effort by completely ignoring $P(X)$.

1.5 Label Bias Problem

Unfortunately, both HMM and MEMM share a common problem: label bias. As an example, consider the transition probabilities depicted in Figure 5. Based on these transition probabilities, state 1 almost always prefers to go to state 2 and state 2 almost always prefers to stay in state 2. Thus, message passing algorithms select a path $1 \to 2 \to 2 \to 2$, but the most likely path is actually $1 \to 1 \to 1 \to 1$. This happens because state 1 has fewer transitions than state 2, and thus the average transition probability from state 2 is lower, resulting in the **label bias problem**: a preference for states with a lower number of transitions. This problem can be solved by replacing the transition probabilities with transition potentials, and leaving normalization the end.

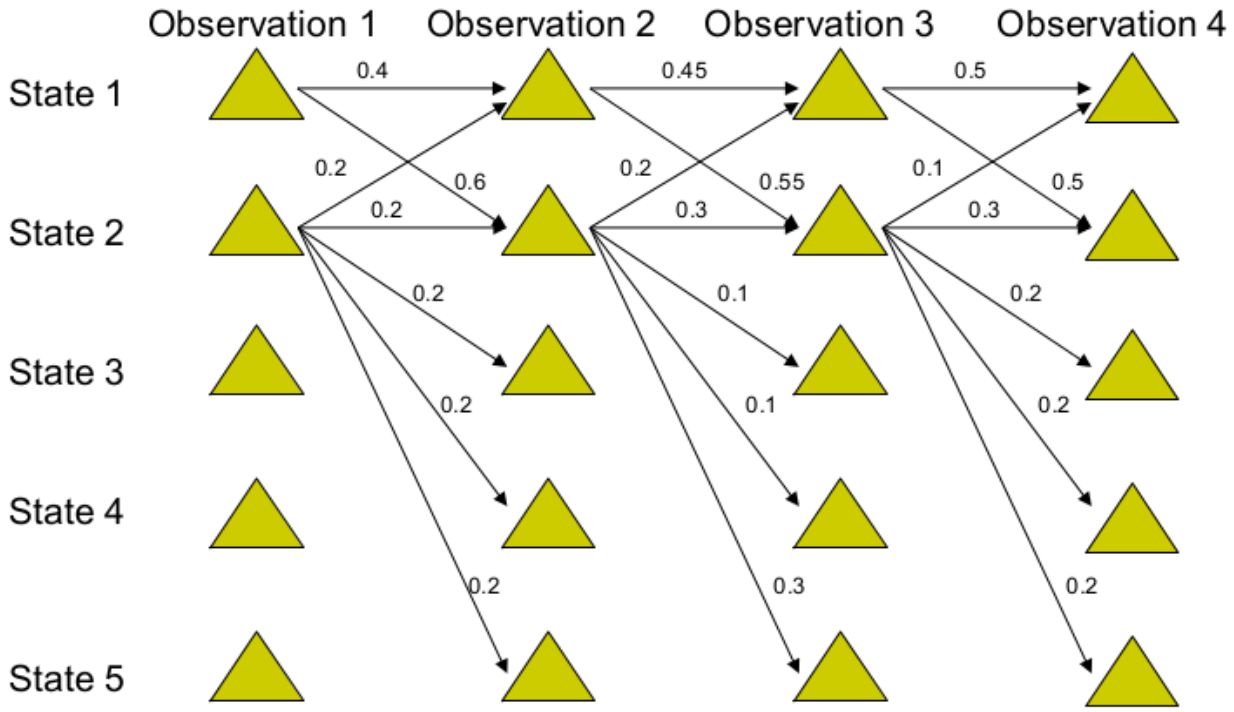

Figure 5: A state transition configuration which results in label bias.

1.6 Linear-chain CRF

A linear-chain CRF is created from a MEMM by removing directionality from connections in Y variables, and moving normalization outside the factors:

$$P(y_{1:n}|x_{1:n}) = \frac{1}{Z(x_{1:n})} \prod_{i=1}^{n} \phi(y_i, y_{i-1}, x_{1:n}) = \frac{1}{Z(x_{1:n})} \prod_{i=1}^{n} exp(w^T f(y_i, y_{i-1}, x_{1:n}))$$

An example of a CRF is shown in Figure 6. This is a partially directed model, and unlike MEMM, each factor is not normalized locally.

The differences between the three models we have discussed in the discrete case can be seen in Figure 7. As shown for all three models Marginal Inference and Map Inference just uses the message passing algorithm,

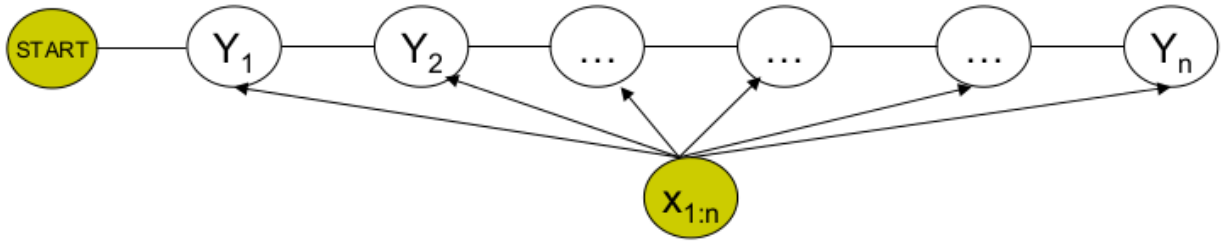

Figure 6: A CRF example.

while the learning algorithm differs depneding on the model

	Learning	Marginal Inference	MAP Inference
HMM	Just counting	Forward-backward	Viterbi
MEMM	Gradient based – decomposes and doesn't require inference (GLM)	Forward-backward	Viterbi
Linear-chain CRF	Gradient based – doesn't decompose because of $Z(x)$ and requires marginal inference	Forward-backward	Viterbi

Figure 7: Table comparing HMM,MEMM, and Chain CRF

1.7 When to choose CRF vs HMM?

Liang and Jordan (ICML 2008) compared HMM and CRF models with identical features. In a real dataset, the CRF outperforms the HMM, but in a simulated dataset, the HMM outperforms. This illustrates a key point: if the model is mis-specified, the discriminative model is better. If the model is close to the truth, the generative model does better. Since most data is generated by a complex process that is not captured according to simple Markov assumptions, a CRF typically outperforms HMM on real data.

1.8 Features that can be used in an MEMM or CRF

Because an MEMM and Linear Chain CRF have arrows from all of X to each tag variable, we can add some more complex features to X to add more information to our learning problem. Features that have been used in the past for speach tagging include:

- Count of tag P as the tag for a specific word, weight of this feature is like the log of an emission probability
- Count of tag P, prior probability
- Count of tag P in the middle of the third sentence
- Count of tag bigram of Verb adjective
- Last 2 chars of word i (will be ed for most past-tense verbs)
- Whether word i appears in thesaurus entry e (one attribute per e)
- Shape of word i (lowercase/capitalized/all caps/numeric/)

1.9 Minimum Bayes Risk Decoding

Often we don't want to weight errors equally. If we are given a loss function $\ell(y', y)$, we seek to minimize the Bayes risk, that it is instead of selecting the assignment which maximizes the log-likelihood $logp(y|x)$, we select

$$h_\theta(x) = argmin\mathbf{E}_{y\sim p_\theta(\cdot|x)}[\ell(\hat{y}, y)]$$

This allows us to reweight incorrect labeling as we see fit. Examples include:

a. The 0-1 loss function, which returns 1 only if the tag assignments to a sentence is identical to the true tag assignment, and 0 otherwise.

$$\ell(\hat{y}, y) = 1 - \mathbb{I}(\hat{y}, y)$$

This loss leads to the following Minimum Bayes Risk decoder:

$$h_\theta(x) = argmin_{\hat{y}} \sum_y p(y|x)(1 - \mathbb{1}(\hat{y}, y)) = argmax_{\hat{y}} p(\hat{y}|x)$$

This is equivalent to the MAP inference problem.

b. The Hamming loss corresponds to a the number of incorrect variable assignments:

$$\ell(\hat{y}, y) = \sum_i^V (1 - \mathbb{I}(\hat{y}_i, y_i)$$

This leads to the Minimum Bayes Risk decoder:

$$h_\theta(x) = argmax_{\hat{y}_i} p(\hat{y}_i|x)$$