MAXIMUM ENTROPY MARKOV MODEL

100050052 – Dikkala Sai Nishanth
100050056 – Ashwin P. Paranjape
100050057 – Vipul Singh

ADAPTED FROM: HESHAAM FAILI
UNIVERSITY OF TEHRAN

INTRODUCTION

- Limitations of HMM
- An overview of HMM vs MEMM
- MEMM and the feature and weight vectors
- Linear and Logistic Regression (MEMM)
- Learning in logistic regression
- Why is it called Maximum Entropy?

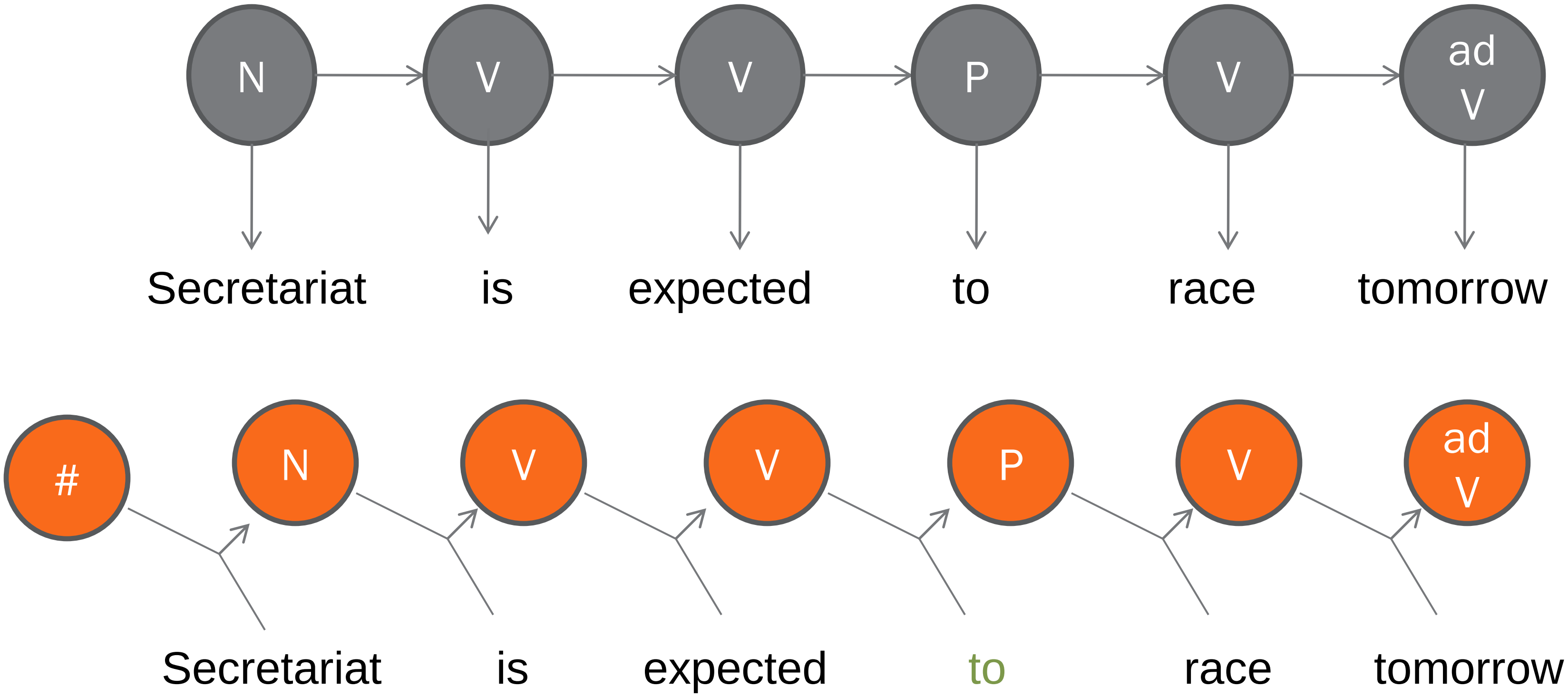
LIMITATIONS OF HMM

HMM – Tag and observed word both depend only on previous tag

Need to account for dependency of tag on observed word

Need to extract *“features”* from word & use

MEMM VS HMM OVERVIEW



MAXIMUM ENTROPY MODELS

- **Machine learning framework called Maximum Entropy modeling.**

- **Used for Classification**
  - The task of classification is to take a single observation, extract some useful features describing the observation, and then based on these features, to **classify** the observation into one of a set of discrete classes.

- **Probabilistic classifier: gives the probability of the observation being in that class**

- **Non-sequential classification**
  - in **text classification** we might need to decide whether a particular email should be classified as spam or not
  - In **sentiment analysis** we have to determine whether a particular sentence or document expresses a positive or negative **opinion.**
  - we'll need to classify a period character ('.') as either a sentence boundary or not

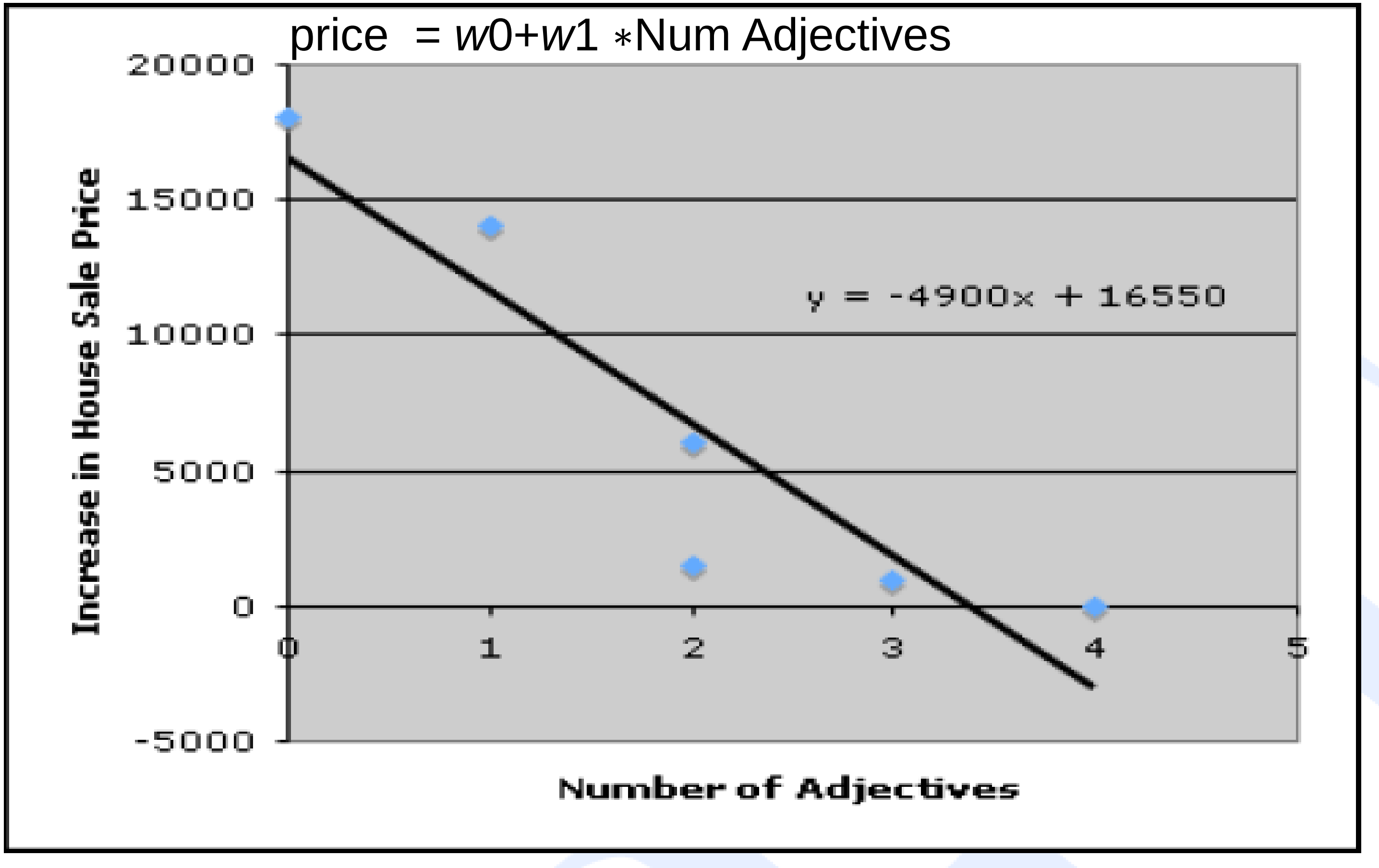
LINEAR REGRESSION

Given a set of real-valued observations, each observation associated with a set of features, linear regression formulates a linear expression which predicts the value of an outcome given its associated features.

E.g: Consider the following data which shows the relation between the number of vague adjectives used in the advertisement for a house and the amount the house fetched.

Number of vague adjectives	Amount sold for over asking price
4	$0
3	$1000
2	$1500
2	$6000
1	$14000
0	$18000

A GRAPH BETWEEN THE HOUSE SALE PRICE VS THE NUMBER OF VAGUE ADJECTIVES USED



MULTIPLE LINEAR REGRESSION

The true power of linear regression is visible when we have multiple features.

E.g:

price=w0+w1 ∗Num Adjectives+w2 ∗Mortgage Rate+w3 ∗Num Unsold Houses

$$\text{price} = w_0 + \sum_{i=1}^{N} w_i \times f_i$$

linear regression:

$$y = \sum_{i=0}^{N} w_i \times f_i$$

$$y = w \cdot f$$

LOGISTIC REGRESSION

Linear regression is what we want when we are predicting a real-valued outcome.

But somewhat more commonly in speech and language processing we are doing classification, in which the output *y we are trying to predict takes on one from a small set of* discrete values.

LOGISTIC REGRESSION

Furthermore, instead of just returning the 0 or 1 value, we'd like a model that can give us the probability that a particular observation is in class 0 or 1.

This is important because in most real-world tasks we're *passing the results of this classifier* onto some further classifier to accomplish some task.

Since we are rarely completely certain about which class an observation falls in, we'd prefer not to make a hard decision at this stage, ruling out all other classes.

LOGISTIC REGRESSION

Suppose we just tried to train a linear model to predict a probability as follows:

$$
\begin{aligned}
P(y = true|x) &= \sum_{i=0}^{N} w_i \times f_i \\
&= w \cdot f
\end{aligned}
$$

Output value need not be between 0 and 1. Consider the odds of the event on the left hand side instead of just the probability.

LOGISTIC REGRESSION

Now, the equation becomes:

$$\frac{p(y = true)|x}{1 - p(y = true|x)} = w \cdot f$$

Still the range of the LHS is only $[0, \infty)$. Consider the natural logarithm of the LHS:

$$\ln\left(\frac{p(y = true|x)}{1 - p(y = true|x)}\right) = w \cdot f$$

LOGISTIC REGRESSION

The logit function:

$$\operatorname{logit}(p(x)) = \ln\left(\frac{p(x)}{1 - p(x)}\right)$$

LOGISTIC REGRESSION

$$\ln\left(\frac{p(y=\text{true}|x)}{1-p(y=\text{true}|x)}\right) = w\cdot f$$

$$\frac{p(y=\text{true}|x)}{1-p(y=\text{true}|x)} = e^{w\cdot f}$$

$$p(y=\text{true}|x) = (1-p(y=\text{true}|x))e^{w\cdot f}$$

$$p(y=\text{true}|x) = e^{w\cdot f} - p(y=\text{true}|x)e^{w\cdot f}$$

$$p(y=\text{true}|x) + p(y=\text{true}|x)e^{w\cdot f} = e^{w\cdot f}$$

$$p(y=\text{true}|x)(1+e^{w\cdot f}) = e^{w\cdot f}$$

$$p(y=\text{true}|x) = \frac{e^{w\cdot f}}{1+e^{w\cdot f}}$$

$$p(y=false|x) = \frac{1}{1+e^{w\cdot f}}$$

LOGISTIC REGRESSION

$$p(y=\text{true}|x) = \frac{\exp(\sum_{i=0}^{N} w_i f_i)}{1+\exp(\sum_{i=0}^{N} w_i f_i)}$$

$$p(y=\text{false}|x) = \frac{1}{1+\exp(\sum_{i=0}^{N} w_i f_i)}$$

$$\begin{aligned} p(y=\text{true}|x) &= \frac{e^{w\cdot f}}{1+e^{w\cdot f}} \\ &= \frac{1}{1+e^{-w\cdot f}} \end{aligned}$$

LOGISTIC REGRESSION: CLASSIFICATION

$$p(y = true|x) > p(y = false|x)$$

$$\frac{p(y = true|x)}{p(y = false|x)} > 1$$

$$\frac{p(y = true|x)}{1 - p(y = true|x)} > 1$$

$$e^{w \cdot f} > 1$$

$$w \cdot f > 0$$

defines a half-space

LEARNING IN LOGISTIC REGRESSION

conditional maximum likelihood estimation.

Choose that weight vector which maximizes product of probabilities of obtaining the observed outputs given the inputs

$$\hat{w} = \underset{w}{\operatorname{argmax}} \prod_{i} P(y^{(i)} | x^{(i)})$$

$$\hat{w} = \underset{w}{\operatorname{argmax}} \sum_{i} \log P(y^{(i)} | x^{(i)})$$

$$\hat{w} = \underset{w}{\operatorname{argmax}} \sum_i \log \begin{cases} P(y^{(i)} = 1 | x^{(i)})) & \text{for } y^{(i)} = 1 \\ P(y^{(i)} = 0 | x^{(i)})) & \text{for } y^{(i)} = 0 \end{cases}$$

$$\hat{w} = \underset{w}{\operatorname{argmax}} \sum_i y^{(i)} \log P(y^{(i)} = 1 | x^{(i)})) + (1 - y^{(i)}) \log P(y^{(i)} = 0 | x^{(i)})$$

LEARNING IN LOGISTIC REGRESSION

$$\hat{w} = \underset{w}{\operatorname{argmax}} \sum_i y^{(i)} \log \frac{e^{-w \cdot f}}{1 + e^{-w \cdot f}} + (1 - y^{(i)}) \log \frac{1}{1 + e^{-w \cdot f}}$$

Convex Optimization

MAXENT

- MaxEnt belongs to the family of classifiers known as the exponential or log-linear classifiers
- MaxEnt works by extracting some set of *features* from the input, combining them linearly (meaning that we multiply each by a *weight* and then add them up), and then using this sum as an exponent
- Example: tagging
  - A feature for tagging might be *this word ends in -ing* or *the previous word was 'the'*

$$p(c|x) = \frac{1}{Z}\exp\left(\sum_i w_i f_i\right)$$

MAXIMUM ENTROPY MODELING

multinomial logistic regression(MaxEnt)

- Most of the time, classification problems that come up in language processing involve larger numbers of classes (part-of-speech classes)

y is a value take on C different value corresponding to classes C1,…,Cn

$$p(c|x) = \frac{1}{Z} \exp \sum_i w_i f_i$$

$$p(c|x) = \frac{\exp\left(\sum_{i=0}^{N} w_{ci} f_i\right)}{\sum_{c' \in C} \exp\left(\sum_{i=0}^{N} w_{c'i} f_i\right)}$$

MAXIMUM ENTROPY MODELING

Indicator function: A feature that only takes on the values 0 and 1

$$Z = \sum_{C} p(c|x) = \sum_{c' \in C} \exp\left(\sum_{i=0}^{N} w_{c'i} f_i\right)$$

$$p(c|x) = \frac{\exp\left(\sum_{i=0}^{N} w_{ci} f_i(c,x)\right)}{\sum_{c' \in C} \exp\left(\sum_{i=0}^{N} w_{c'i} f_i(c',x)\right)}$$

MAXIMUM ENTROPY MODELING

$$f_1(c,x) = \begin{cases} 1 & \text{if } word_i = \text{“race”} \ \& \ c = \text{NN} \\ 0 & \text{otherwise} \end{cases}$$

$$f_2(c,x) = \begin{cases} 1 & \text{if } t_{i-1} = \text{TO} \ \& \ c = \text{VB} \\ 0 & \text{otherwise} \end{cases}$$

$$f_3(c,x) = \begin{cases} 1 & \text{if } \text{suffix}(word_i) = \text{“ing”} \ \& \ c = \text{VBG} \\ 0 & \text{otherwise} \end{cases}$$

$$f_4(c,x) = \begin{cases} 1 & \text{if } \text{is\_lower\_case}(word_i) \ \& \ c = \text{VB} \\ 0 & \text{otherwise} \end{cases}$$

$$f_5(c,x) = \begin{cases} 1 & \text{if } word_i = \text{”race”} \ \& \ c = \text{VB} \\ 0 & \text{otherwise} \end{cases}$$

$$f_6(c,x) = \begin{cases} 1 & \text{if } t_{i-1} = \text{TO} \ \& \ c = \text{NN} \\ 0 & \text{otherwise} \end{cases}$$

MAXIMUM ENTROPY MODELING

		f1	f2	f3	f4	f5	f6
VB	f	0	1	0	1	1	0
VB	w		.8		.01	.1	
NN	f	1	0	0	0	0	1
NN	w	.8					-1.3

Sample feature and weight values for "race"

$$P(NN|x) = \frac{e^{.8}e^{-1.3}}{e^{.8}e^{-1.3} + e^{.8}e^{.01}e^{.1}} = .20$$

$$P(VB|x) = \frac{e^{.8}e^{.01}e^{.1}}{e^{.8}e^{-1.3} + e^{.8}e^{.01}e^{.1}} = .80$$

- MaxEnt gives probability distribution over the classes.
- If we want to do a hard-classification, i.e., choose the single-best class, we can choose the class that has the highest probability

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|x)$$

THE OCCAM RAZOR

Adopting the least complex hypothesis possible is embodied in Occam's razor

The intuition of MaxEnt modeling : probabilistic model should follow whatever constraints we impose on it, but beyond these constraints it should follow Occam's Razor, i.e. make the fewest possible assumptions.

WHY DO WE CALL IT MAXIMUM ENTROPY?

- Information : NIL
- Output: 0.2 probability for each of N,V,A,R,O

- Information : 2 out of 5 occurrences as verb
- Output: 0.4 probability for V, 0.15 for each of the other 4

- From all of possible distributions, the equi-probable distribution has the maximum entropy.

- Recall : Entropy for distribution of a r.v. x is:

$$H(x) = -\sum_x P(x)\log_2 P(x)$$

MAXIMUM ENTROPY

"To select a model from a set C of allowed probability distributions, choose the model $p_* \in C$ with maximum entropy $H(p)$":

$$p_* = \underset{p \in C}{\operatorname{argmax}} H(p)$$


ARE WE ON COURSE?
OFF COURSE, WE ARE
memegenerator.net

MAXIMUM ENTROPY MARKOV MODELS

MaxEnt

- Not in itself a classifier for sequences
- Classify a single observation into one of a set of discrete classes
- Naïve classification possible using hard decision for every word.

MEMM

- Combines HMM and MaxEnt
- MaxEnt - applied to assign a class to each element in a sequence

MEMM VS. HMM

Finding the most probable tag sequence

HMM:

MEMM:

$$\begin{aligned}
\hat{T} &= \underset{T}{\operatorname{argmax}} P(T|W) \\
&\overset{\text{Bayes}}{=} \underset{T}{\operatorname{argmax}} P(W|T)P(T) \\
&\overset{\text{Markov}}{=} \underset{T}{\operatorname{argmax}} \prod_i P(word_i|tag_i) \prod_i P(tag_i|tag_{i-1})
\end{aligned}$$

$$\begin{aligned}
\hat{T} &= \underset{T}{\operatorname{argmax}} P(T|W) \\
&= \underset{T}{\operatorname{argmax}} \prod_i P(tag_i|word_i, tag_{i-1})
\end{aligned}$$

MEMM VS. HMM

HMM

HMM model includes distinct probability estimates for each transition and observation

$$
\begin{aligned}
\hat{T} &= \operatorname*{argmax}_T P(T|W) \\
&= \operatorname*{argmax}_T P(W|T)P(T) \\
&= \operatorname*{argmax}_T \underbrace{\prod_i P(word_i|tag_i)}_{\textit{Likelihood}} \underbrace{\prod_i P(tag_i|tag_{i-1})}_{\textit{Prior}}
\end{aligned}
$$

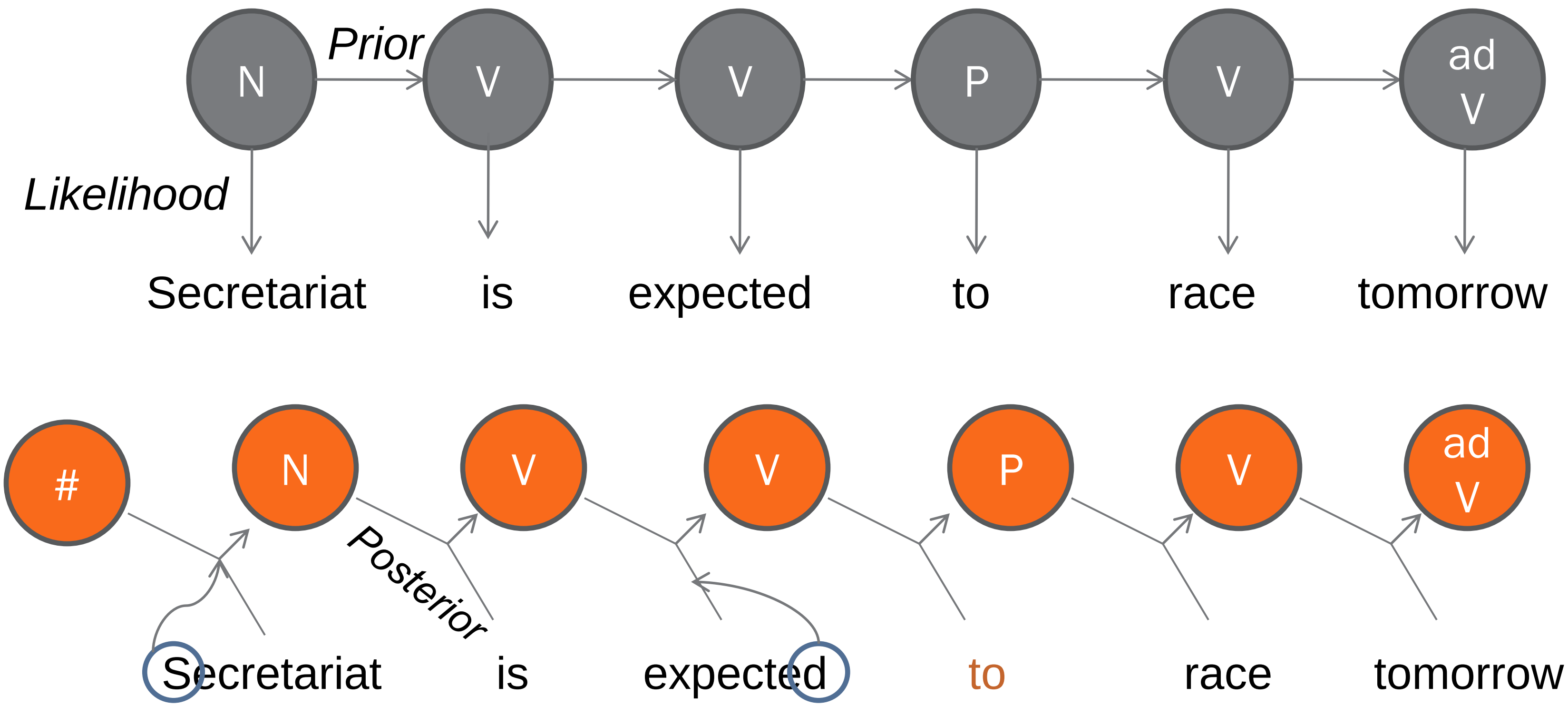
Generative

MEMM

MEMM gives one probability estimate per hidden state, which is the probability of the next tag given the previous tag and the observation.

Discr
$$
\begin{aligned}
\hat{T} &= \operatorname*{argmax}_T P(T|W) \\
&= \operatorname*{argmax}_T \prod_i \underbrace{P(tag_i|word_i, tag_{i-1})}_{\textit{Posterior}}
\end{aligned}
$$

MEMM VS HMM OVERVIEW



MEMM VS. HMM

State sequence Q, observation sequence O

HMM:

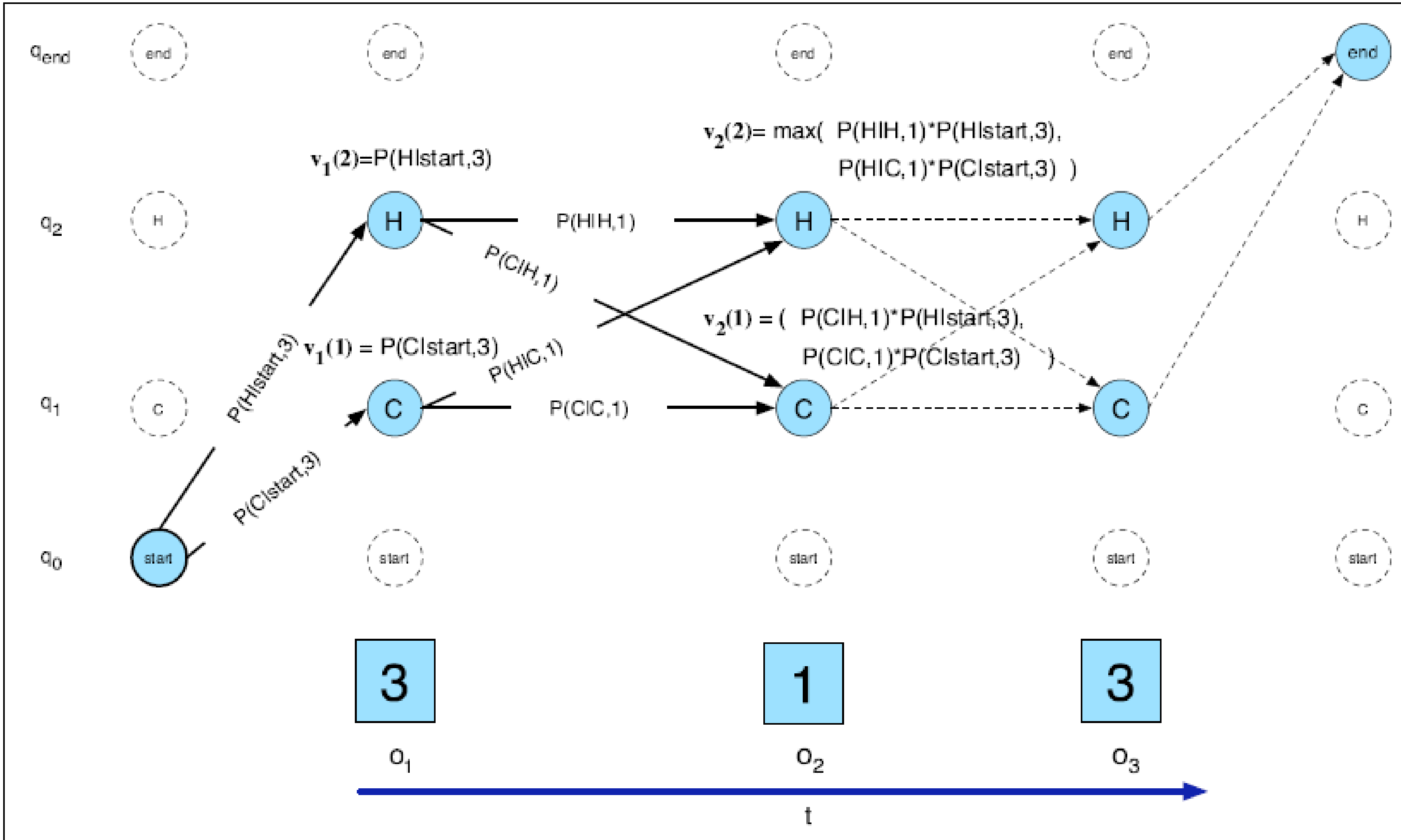
MEMM:

$$P(Q|O) = \prod_{i=1}^{n} P(o_i|q_i) \times \prod_{i=1}^{n} P(q_i|q_{i-1})$$

$$P(Q|O) = \prod_{i=1}^{n} P(q_i|q_{i-1}, o_i)$$

$$P(q|q', o) = \frac{1}{Z(o, q')} \exp\left(\sum_i w_i f_i(o, q)\right)$$

MaxEnt



q_{end}
q_2
q_1
q_0
end
start
H
C
$\mathbf{v_1(2)}$=P(H|start,3)
$\mathbf{v_1(1)}$ = P(C|start,3)
$\mathbf{v_2(2)}$= max(P(H|H,1)*P(H|start,3), P(H|C,1)*P(C|start,3))
$\mathbf{v_2(1)}$ = (P(C|H,1)*P(H|start,3), P(C|C,1)*P(C|start,3))
P(H|start,3)
P(C|start,3)
P(H|H,1)
P(C|H,1)
P(H|C,1)
P(C|C,1)
3
1
3
o_1
o_2
o_3
t

DECODE AND LEARN IN MEMM

- Viterbi for HMM:

- Viterbi for MEMM:

$$v_t(j) = \max_{1 \le i \le N-1} v_{t-1}(i)\, P(s_j|s_i)\, P(o_t|s_j); \quad 1 < j < N, 1 < t < T$$

$$v_t(j) = \max_{1 \le i \le N-1} v_{t-1}(i)\, P(s_j|s_i, o_t); \quad 1 < j < N, 1 < t < T$$

CONCLUSIONS

MaxEnt model is a classifier which assigns a class to an observation by computing a probability from an exponential function of a weighted set of features of the observation.

MaxEnt models can be trained using methods from the field of convex optimization.

A Maximum Entropy Markov Model or MEMM is a sequence model augmentation of MaxEnt which makes use of the Viterbi decoding algorithm.

REFERENCES

Jurafsky, Daniel and Martin, James H. (2006) *Speech and Language Processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Prentice-Hall.

Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Comput. Linguist.* 22, 1 (March 1996), 39-71.

Ratnaparkhi A. 1996. A Maximum Entropy Model for Part-of-Speech Tagging. *Proceedings of the Empirical Methods in Natural Language Processing* (1996), pp. 133-142

THANK YOU