Baum Welch Algorithm

For POS Tagging

Baum Welch Algorithm

Uses the well-known EM algorithm to find the maximum likelihood estimate of the parameters of a hidden markov model

Baum Welch Algorithm

Uses the well-known EM algorithm to find the maximum likelihood estimate of the parameters of a hidden markov model

Parameters of HMM

Let X_t be the random variable denoting hidden state at time t, and Y_t be the observation variable at time T. HMM parameters are given by $\theta = (A, B, \pi)$ where

- $A = \{a_{ij}\} = P(X_t = j | X_{t-1} = i)$ is the state transition matrix
- $\pi = \{\pi_i\} = P(X_1 = i)$ is the initial state distribution
- $B = \{b_j(y_t)\} = P(Y_t = y_t | X_t = j)$ is the emission matrix

Baum Welch Algorithm

Uses the well-known EM algorithm to find the maximum likelihood estimate of the parameters of a hidden markov model

Parameters of HMM

Let X_t be the random variable denoting hidden state at time t, and Y_t be the observation variable at time T. HMM parameters are given by $\theta = (A, B, \pi)$ where

- $A = \{a_{ij}\} = P(X_t = j | X_{t-1} = i)$ is the state transition matrix
- $\pi = \{\pi_i\} = P(X_1 = i)$ is the initial state distribution
- $B = \{b_j(y_t)\} = P(Y_t = y_t | X_t = j)$ is the emission matrix

Given observation sequences $Y = (Y_1 = y_1, Y_2 = y_2, \ldots, Y_T = y_T)$, the algorithm tries to find the parameters θ that maximise the probability of the observation.

The Algorithm

The basic idea is to start with some random initial conditions on the parameters θ, estimate best values of state paths X_t using these, then re-estimate the parameters θ using the just-computed values of X_t, iteratively.

The Algorithm

The basic idea is to start with some random initial conditions on the parameters θ, estimate best values of state paths X_t using these, then re-estimate the parameters θ using the just-computed values of X_t, iteratively.

Intuition

- Choose some initial values for $\theta = (A, B, \pi)$.
- *Repeat the following step until convergence:*
- Determine probable (state) paths $\ldots X_{t-1} = i, X_t = j \ldots$
- Count the expected number of transitions a_{ij} as well as the expected number of times, various emissions $b_j(y_t)$ are made
- Re-estimate $\theta = (A, B, \pi)$ using a_{ij} and $b_j(y_t)$s.

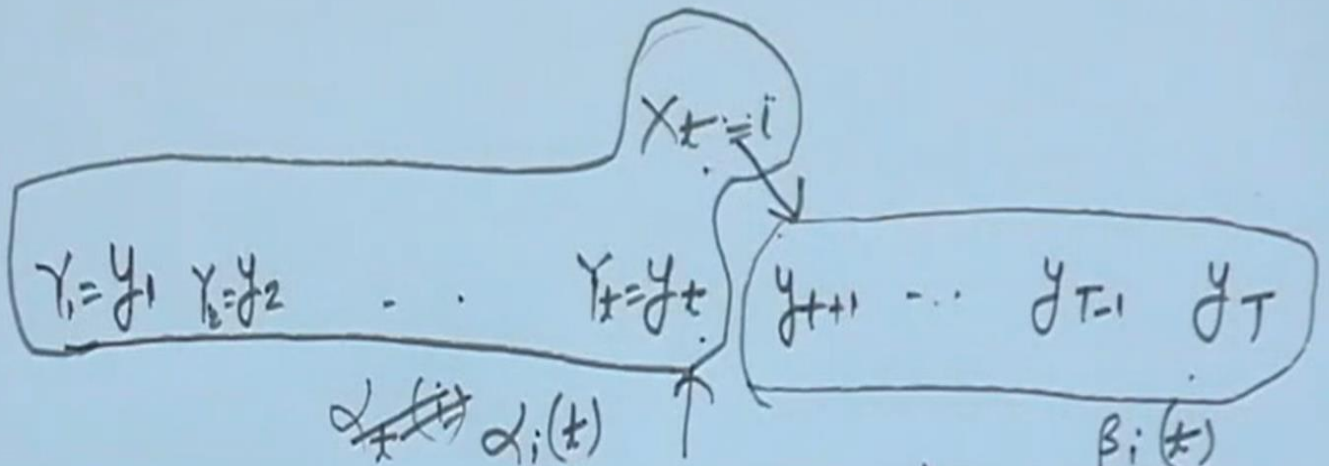
A forward-backward algorithm is used for finding probable paths.

Forward-Backward Algorithm

Forward Procedure

$\alpha_i(t) = P(Y_1 = y_1, \ldots, Y_t = y_t, X_t = i|\theta)$ be the probability of seeing $y_1, \ldots, y_t$ and being in state i at time t. Found recursively using:

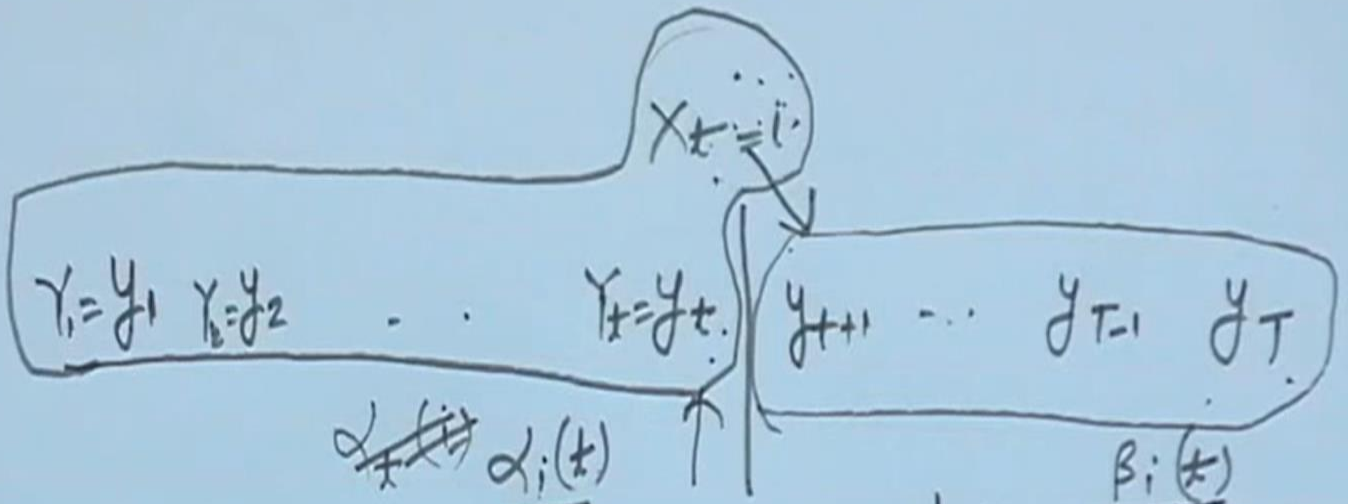
- $\alpha_i(1) = \pi_i b_i(y_1)$



$\alpha_i(t)$ $\beta_i(t)$

$= P(Y_1 = y_1, Y_2 = y_2 \cdots Y_t = y_t, X_t = i \mid \theta)$ $P(Y_{t+1} = y_{t+1} \cdots$

$Y_T = y_T \mid X_t = i, \theta)$



$= P(Y_1 = y_1, Y_2 = y_2 \cdots Y_t = y_t, X_t = i \mid \theta)$ $P(Y_{t+1} = y_{t+1} \cdots$

$Y_T = y_T \mid X_t = i, \theta)$

$\alpha_i(1) = P(Y_1 = y_1, X_1 = i \mid \theta)$

$= \pi_i b_i(y_1)$

$\alpha_j(t+1) = \sum_{i=1}^{N} \alpha_i(t)\, a_{ij}\, b_j(y_{t+1})$

Forward-Backward Algorithm

Forward Procedure

$\alpha_i(t) = P(Y_1 = y_1, \ldots, Y_t = y_t, X_t = i|\theta)$ be the probability of seeing $y_1, \ldots, y_t$ and being in state i at time t. Found recursively using:

- $\alpha_i(1) = \pi_i b_i(y_1)$
- $\alpha_j(t+1) = b_j(y_{t+1}) \sum_{i=1}^{N} \alpha_i(t) a_{ij}$

Backward Procedure

$\beta_i(t) = P(Y_{t+1} = y_{t+1}, \ldots, Y_T = y_T|X_t = i, \theta)$ be the probability of ending partial sequence $y_{t+1}, \ldots, y_T$ given starting state i at time t. $\beta_i(t)$ is computed recursively as:

- $\beta_i(T) = 1$

Forward-Backward Algorithm

Forward Procedure

$\alpha_i(t) = P(Y_1 = y_1, \ldots, Y_t = y_t, X_t = i|\theta)$ be the probability of seeing $y_1, \ldots, y_t$ and being in state i at time t. Found recursively using:

- $\alpha_i(1) = \pi_i b_i(y_1)$
- $\alpha_j(t+1) = b_j(y_{t+1}) \sum_{i=1}^{N} \alpha_i(t) a_{ij}$

Backward Procedure

$\beta_i(t) = P(Y_{t+1} = y_{t+1}, \ldots, Y_T = y_T|X_t = i, \theta)$ be the probability of ending partial sequence $y_{t+1}, \ldots, y_T$ given starting state i at time t. $\beta_i(t)$ is computed recursively as:

- $\beta_i(T) = 1$
- $\beta_i(t) = \sum_{j=1}^{N} \beta_j(t+1) a_{ij} b_j(y_{t+1})$

Finding probabilities of paths

We compute the following variables:

- Probability of being in state i at time t given the observation Y and parameters θ

$$\gamma_i(t) = P(X_t = i|Y, \theta) = \frac{\alpha_i(t)\beta_i(t)}{\sum_{j=1}^{N} \alpha_j(t)\beta_j(t)}$$

- Probability of being in state i and j at time t and $t+1$ respectively given the observation Y and parameters θ

$$\zeta_{ij}(t) = P(X_t = i, X_{t+1} = j|Y, \theta) = \frac{\alpha_i(t)a_{ij}\beta_j(t+1)b_j(y_{t+1})}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i(t)a_{ij}\beta_j(t+1)b_j(y_{t+1})}$$

Finding probabilities of paths

We compute the following variables:

- Probability of being in state i at time t given the observation Y and parameters θ

$$\gamma_i(t) = P(X_t = i|Y,\theta) = \frac{\alpha_i(t)\beta_i(t)}{\sum_{j=1}^{N}\alpha_j(t)\beta_j(t)}$$

- Probability of being in state i and j at time t and $t+1$ respectively given the observation Y and parameters θ

$$\zeta_{ij}(t) = P(X_t = i, X_{t+1} = j|Y,\theta) = \frac{\alpha_i(t)a_{ij}\beta_j(t+1)b_j(y_{t+1})}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i(t)a_{ij}\beta_j(t+1)b_j(y_{t+1})}$$

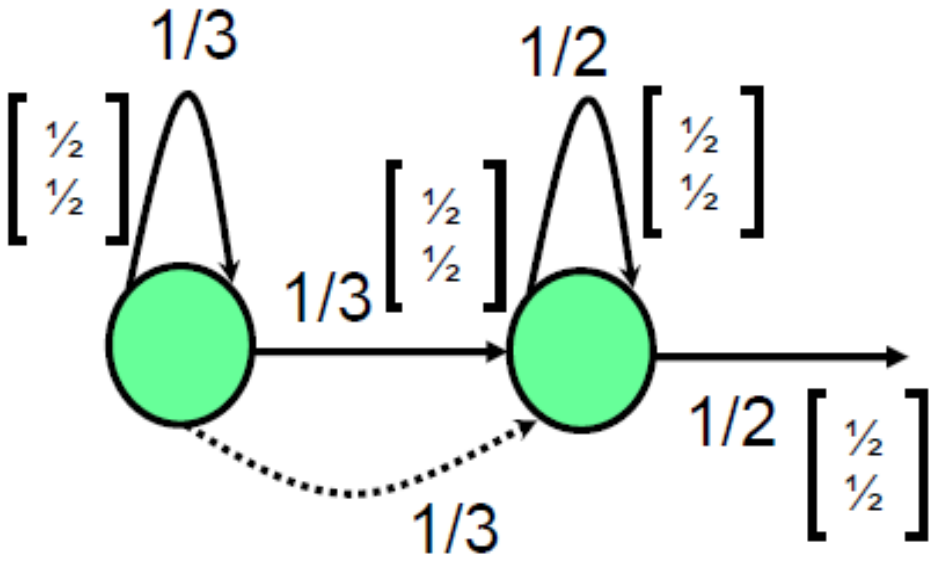
Updating the parameters

- $\pi_i = \gamma_i(1)$, expected number of times state i was seen at time 1
- $a_{ij} = \frac{\sum_{t=1}^{T} \zeta_{ij}(t)}{\sum_{t=1}^{T} \gamma_i(t)}$, expected number of transitions from state i to state j, compared to the total number of transitions away from state i
- $b_i(v_k) = \frac{\sum_{t=1}^{T} 1_{y_t = v_k} \gamma_i(t)}{\sum_{t=1}^{T} \gamma_i(t)}$ with $1_{y_t = v_k}$ being an indicator function, is the expected number of times the output observations are v_k while being in state i compared to the expected total number of times in state i.

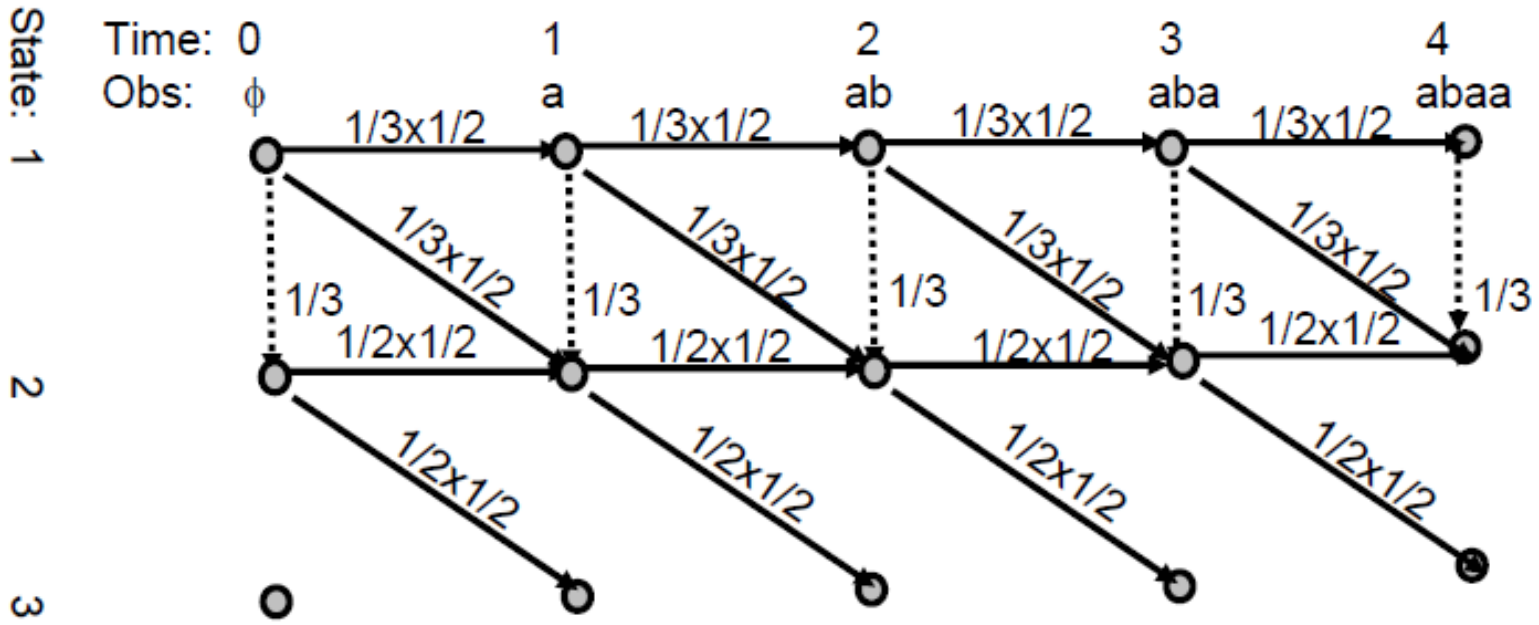
Example: Baum Welch (Forward Backward algorithm) for HMM

Step 1:

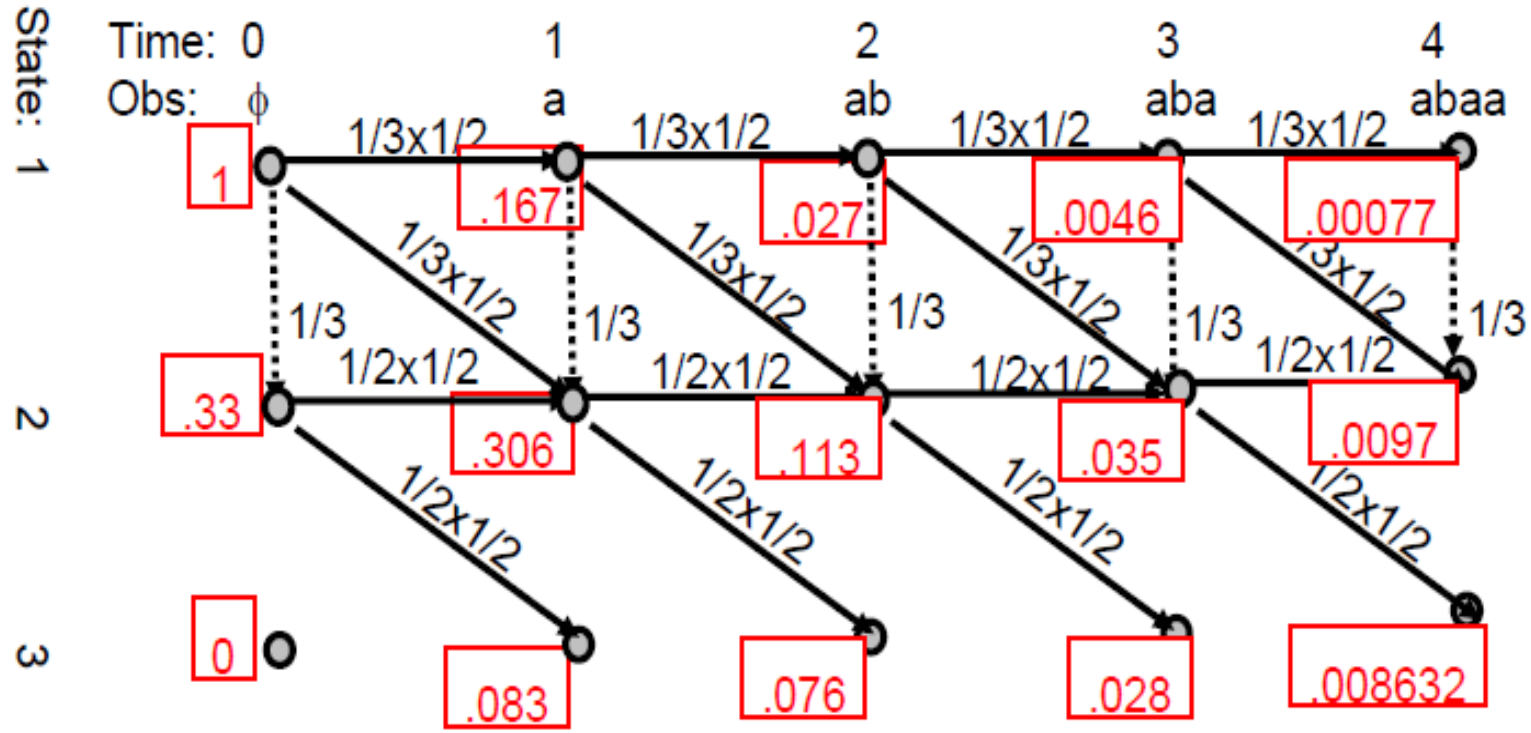
Problem for Forward Backward algorithm:



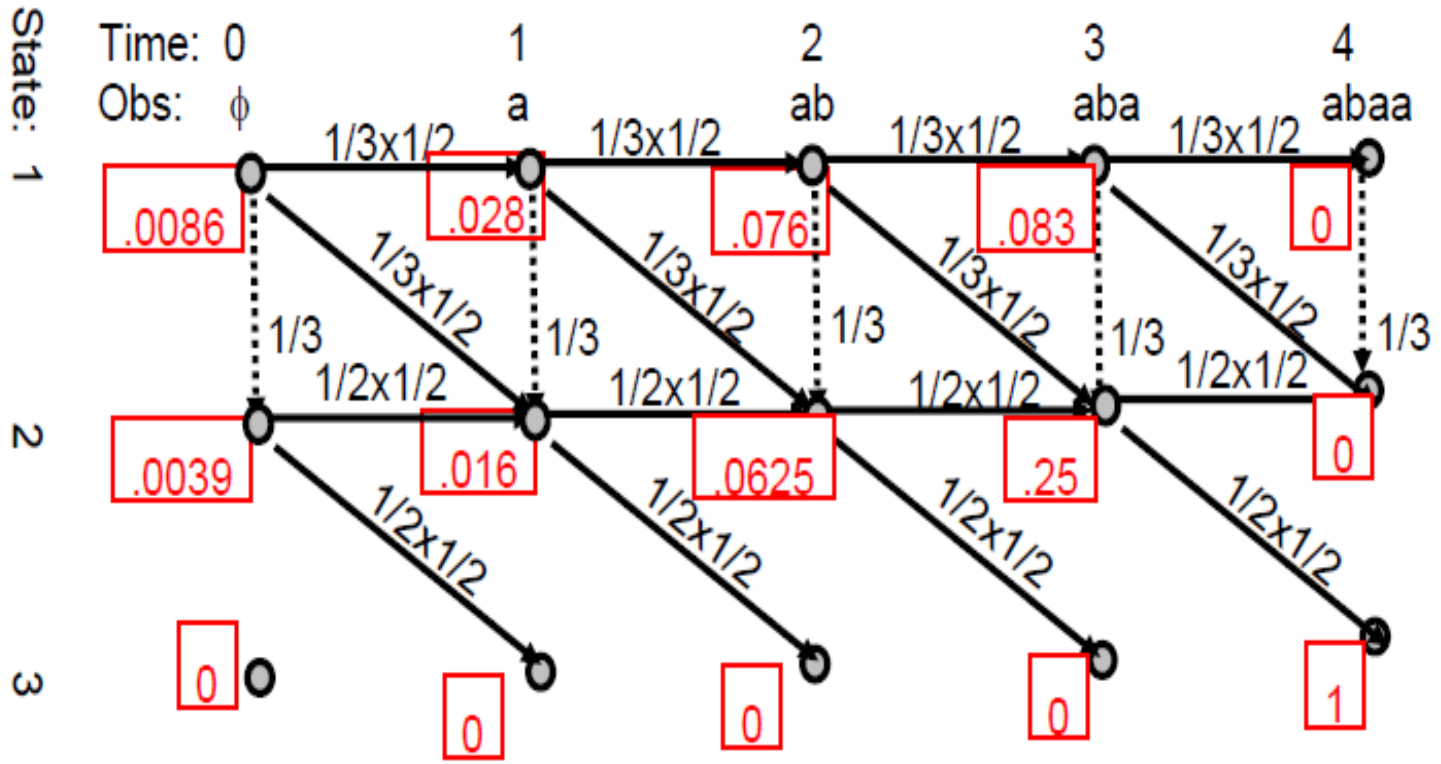
Step 2: Representation in Trellis



Step 3: Compute Alpha's



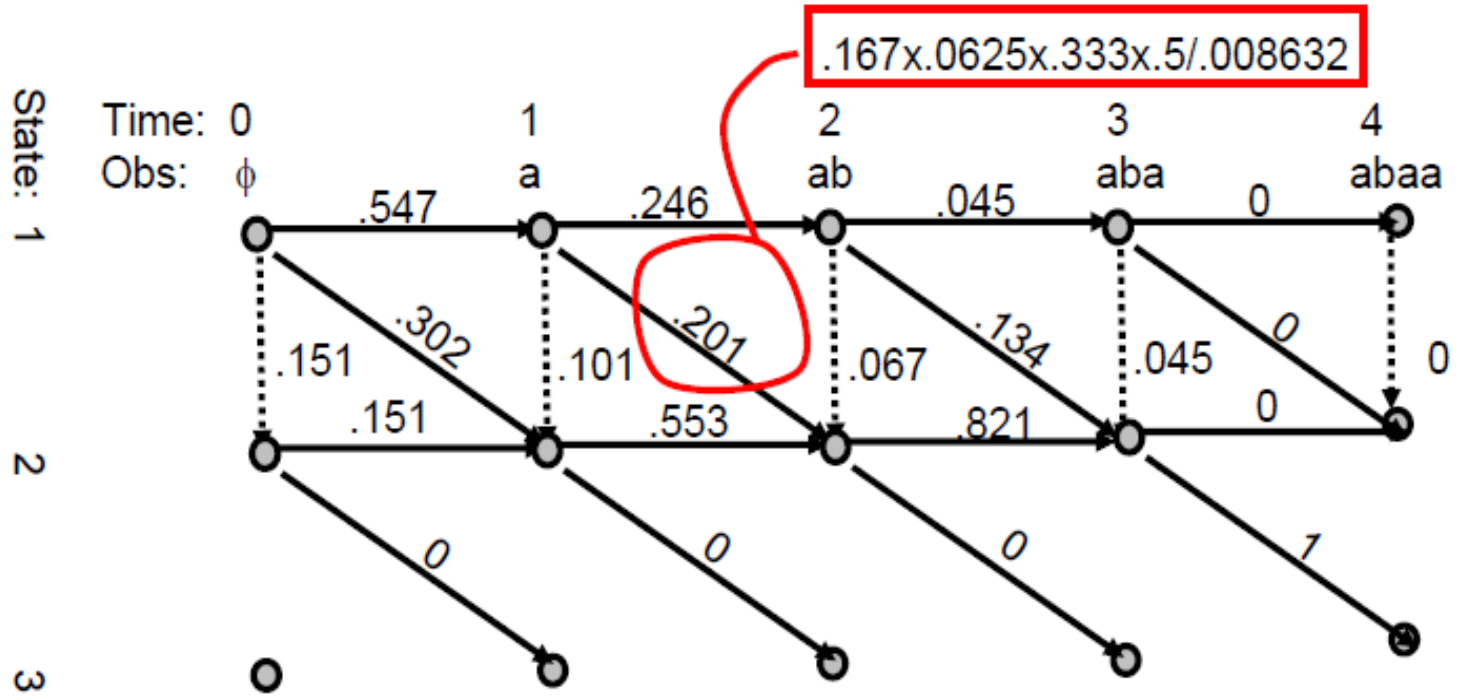
Step 4: Compute Beta's



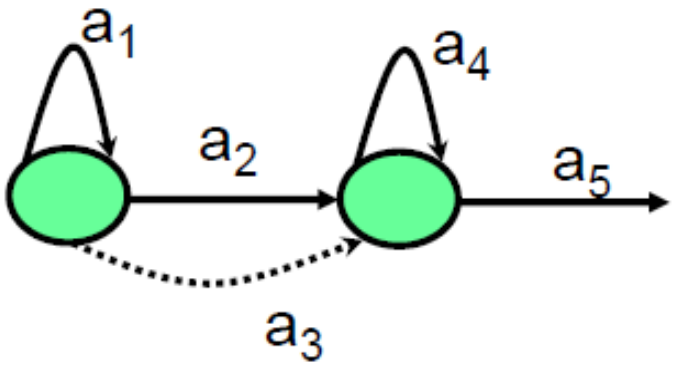
Step 5:

Compute counts. (a posteriori probability of each transition)

$$c_t(tr_{ij}|X) = \alpha_{t-1}(i)\, a_{ij}\, b_{ij}(x_t)\ \beta_t(j)/\ Pr(X)$$



Step 6: Compute Counts

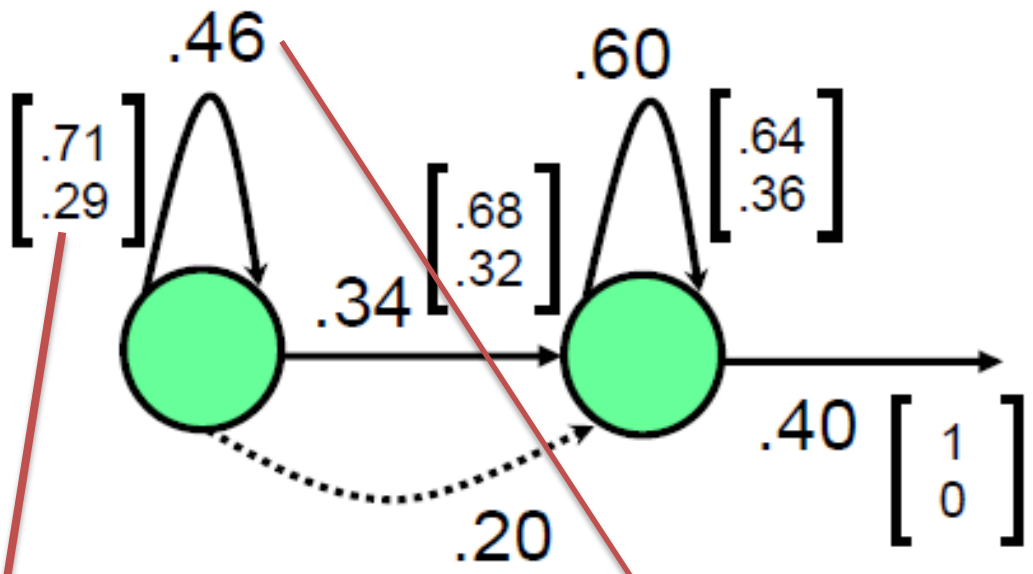


- $C(a_1)$=.547+.246+.045
- $C(a_2)$=.302+.201+.134
- $C(a_3)$=.151+.101+.067+.045
- $C(a_4)$=.151+.553+.821
- $C(a_5)$=1

- C(a1,'a')=.547+.045, C(a1,'b')=.246
- C(a2,'a')=.302+.134, C(a2,'b')=.201
- C(a4,'a')=.151+.821, C(a4,'b')=.553
- C(a5,'a')=1, C(a5,'b')=0

Step 7:

Normalize counts to get new parameter values.



(0.547 + 0.045) / (0.547 + 0.246 + 0.045) = 0.71

(0.246)/ (0.547 + 0.246 + 0.045) = 0.29

c(a1)/(c(a1) + c(a2) + c(a3)

= 0.838/(0.838+0.637+0.364)

= 0.46

Issues with Markov Model Tagging

Unknown Words

We do not have the required probabilities.

Possible solutions:

- Use morphological cues (capitalization, suffix) to assign a more calculated guess

Issues with Markov Model Tagging

Unknown Words

We do not have the required probabilities.

Possible solutions:

- Use morphological cues (capitalization, suffix) to assign a more calculated guess

Limited Context

- "is clearly **marked**" → verb, past participle
- "he clearly **marked**" → verb, past tense

Issues with Markov Model Tagging

Unknown Words

We do not have the required probabilities.
Possible solutions:

- Use morphological cues (capitalization, suffix) to assign a more calculated guess

Limited Context

- "is clearly **marked**" → verb, past participle
- "he clearly **marked**" → verb, past tense

Possible solution: Use higher order model, combine various n-gram models to avoid sparseness problem

Maximum Entropy Modeling: Discriminative Model

- We may identify a heterogeneous set of features which contribute in some way to the choice of POS tag of the current word.

Maximum Entropy Modeling: Discriminative Model

- We may identify a heterogeneous set of features which contribute in some way to the choice of POS tag of the current word.
  - Whether it is the first word in the article
  - Whether the next word is *to*
  - Whether one of the last 5 words is a preposition, etc.
- MaxEnt combines these features in a probabilistic model

Maximum Entropy: The Model

$$p_\lambda(y|x) = \frac{1}{Z_\lambda(x)} exp\left(\sum_i \lambda_i f_i(x,y)\right)$$

where

- $Z_\lambda(x)$ is a normalizing constant given by

$$Z_\lambda(x) = \sum_y exp\left(\sum_i \lambda_i f_i(x,y)\right)$$

Maximum Entropy: The Model

$$p_\lambda(y|x) = \frac{1}{Z_\lambda(x)} exp\left(\sum_i \lambda_i f_i(x,y)\right)$$

where

- $Z_\lambda(x)$ is a normalizing constant given by

$$Z_\lambda(x) = \sum_y exp\left(\sum_i \lambda_i f_i(x,y)\right)$$

- λ_i is a weight given to a feature f_i
- x denotes an observed datum and y denotes a class

What is the form of the features?

Features in Maximum Entropy Models

- Features encode elements of the context x for predicting tag y
- Context x is taken around the word w, for which a tag y is to be predicted

Features in Maximum Entropy Models

- Features encode elements of the context x for predicting tag y
- Context x is taken around the word w, for which a tag y is to be predicted
- Features are binary values functions, e.g.,

$$f(x,y) = \left\{ \begin{array}{ll} 1 & \text{if } isCapitalized(w) \& y = NNP \\ 0 & otherwise \end{array} \right\}$$

Example Features

Example: Named Entities

- LOCATION (in Arcadia)
- LOCATION (in Québec)
- DRUG (taking Zantac)
- PERSON (saw Sue)

Example Features

Example: Named Entities

- LOCATION (in Arcadia)
- LOCATION (in Québec)
- DRUG (taking Zantac)
- PERSON (saw Sue)

Example Features

- $f_1(x, y) = [y = LOCATION \wedge w_{-1} = \text{“}in\text{”} \wedge isCapitalized(w)]$
- $f_2(x, y) = [y = LOCATION \wedge hasAccentedLatinChar(w)]$
- $f_3(x, y) = [y = DRUG \wedge ends(w, \text{“}c\text{”})]$

Tagging with Maximum Entropy Model

- $W = w_1 \ldots w_n$ - words in the corpus (observed)
- $T = t_1 \ldots t_n$ - the corresponding tags (unknown)

Tagging with Maximum Entropy Model

- $W = w_1 \ldots w_n$ - words in the corpus (observed)
- $T = t_1 \ldots t_n$ - the corresponding tags (unknown)

Tag sequence candidate $\{t_1, \ldots, t_n\}$ has conditional probability:

$$P(t_1, \ldots, t_n | w_1 \ldots, w_n) = \prod_{i=1}^{n} p(t_i | x_i)$$

- The context x_i also includes previously assigned tags for a fixed history.
- Beam search is used to find the most probable sequence

Beam Inference

Beam Inference

- At each position, keep the top k complete sequences
- Extend each sequence in each local way
- The extensions compete for the k slots at the next position

Beam Inference

Beam Inference

- At each position, keep the top k complete sequences
- Extend each sequence in each local way
- The extensions compete for the k slots at the next position

But what is a MaxEnt model?

Let's go to the basics now!

Maximum Entropy Model

Intuitive Principle

Model all that is known and assume nothing about that which is unknown. *Given a collection of facts, choose a model which is consistent with all the facts, but otherwise as uniform as possible.*

Maximum Entropy: Overview

- Suppose we wish to model an expert translator's decisions concerning the proper French rendering of the English word '*in*'.

Maximum Entropy: Overview

- Suppose we wish to model an expert translator's decisions concerning the proper French rendering of the English word '*in*'.
- Each French word or phrase f is assigned an estimate $p(f)$, probability that the expert would choose f as a translation of '*in*'.
- Collect a large sample of instances of the expert's decisions
- **Goal**: extract a set of facts about the decision-making process (first task) that will aid in constructing a model of this process (second task)

Maximum Entropy Model: Overview

First clue: list of allowed translations

- Suppose the translator always chooses among {dans, en, á, au cours de, pendant}.
- First constraint: p(dans)+p(en)+p(á)+p(au cours de)+p(pendant) = 1.
- Infinite number of models p for which this identity holds, the most intuitive model?
- *allocate the total probability evenly among the five possible phrases* $\rightarrow$ most uniform model subject to our knowledge.
- *Is it the most uniform model overall?*