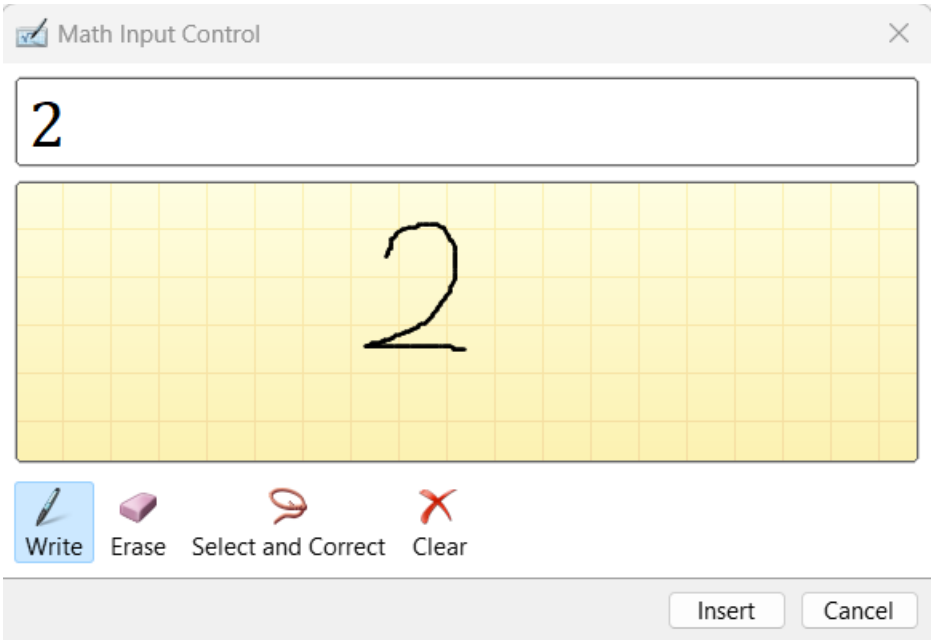
Idea

Simply the idea is practical and self-innovated and created from scratch. The whole idea and the conversion to normal examples we know.

There are many applications that use **online handwritten conversion to digital form** that computers can understand. This image is an example from **Microsoft Word.**



To keep the idea simple our domain will be number from 0 to 9, and you can convert single digit at one time, it can be enhanced in the future to capture English or Arabic letter, word, and sentences.

The dataset

We will create our HMM dataset. Each number is presented as a list of (x,y) points. All these points fall into the line that drawn that number. And the data is stored in JSON format.

Can be used to fine tune HMM model with baum-welch algorithm.

Output

Firstly, we train our model on our dataset, we can save both the model or dataset to use later.

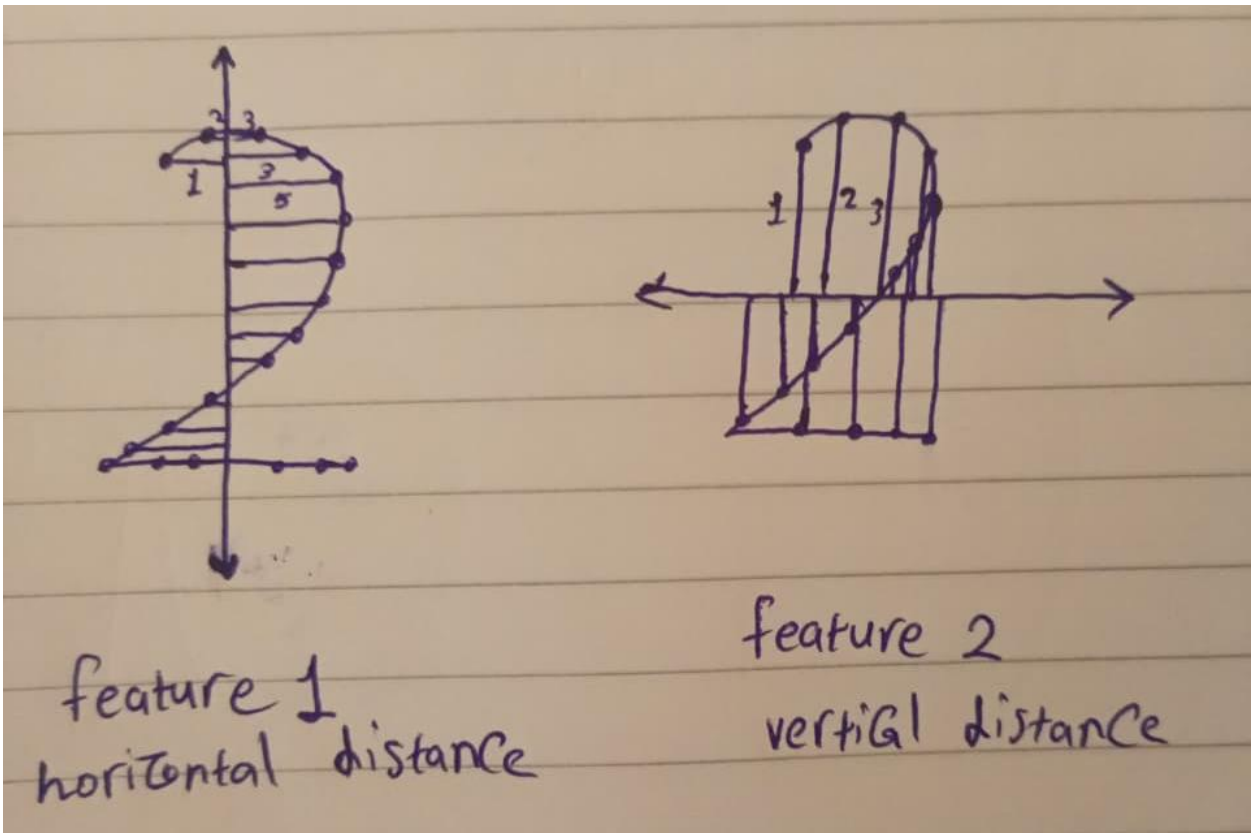
Then your draw a single number from 0 to 9 and the model will show the probabilities to be any number of 0 to 9 and highlights the highest probability as the expected number.

Baum-welch algorithm

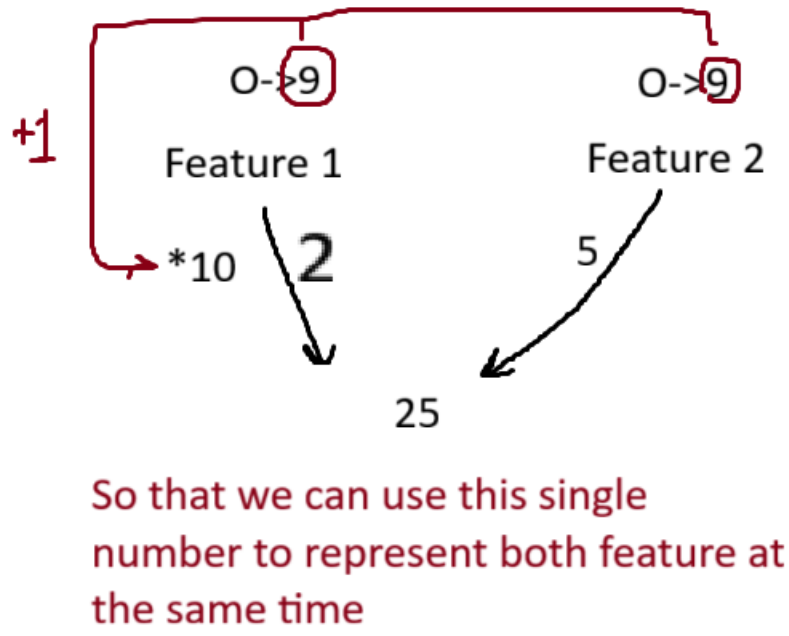
We need two things to train the model:

- Sequence of observations O_1, O_2, ..., O_n
  This will be pairs of **(distance from the central vertical line, distance from the central horizontal line).**
- Sequence of hidden states for these observations S_1, S_2, ..., S_n that we can not see.

How will the states be presented?



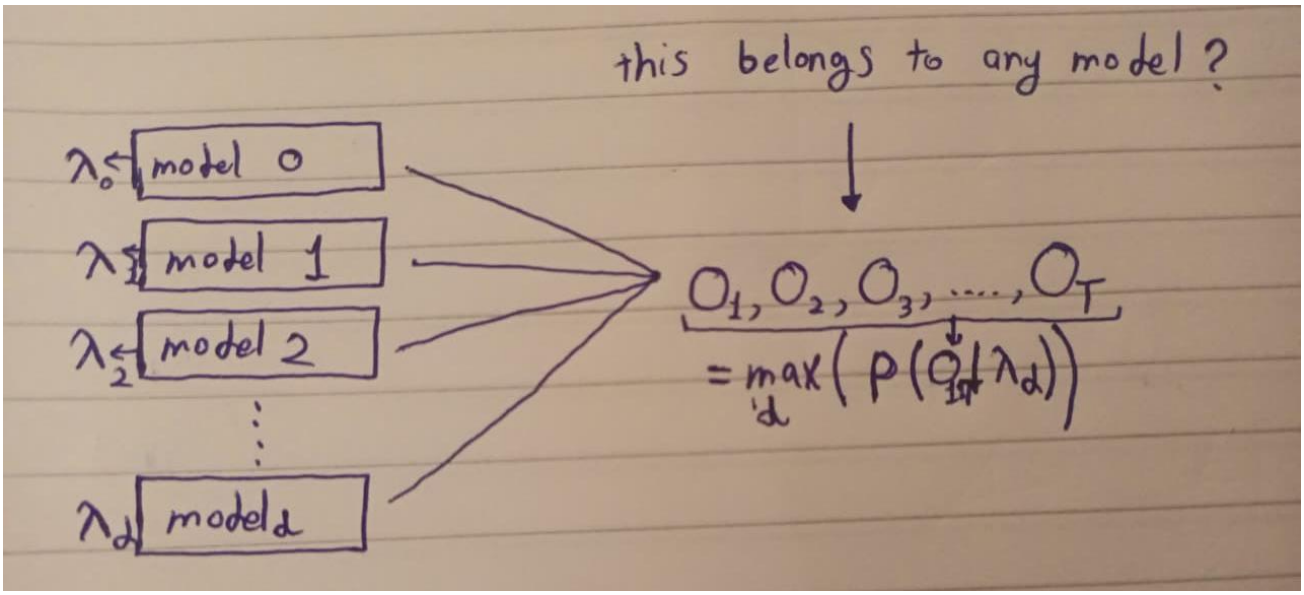
The features are discretized in one single feature (observable state). So that the model can capture more variance. This image explains this:



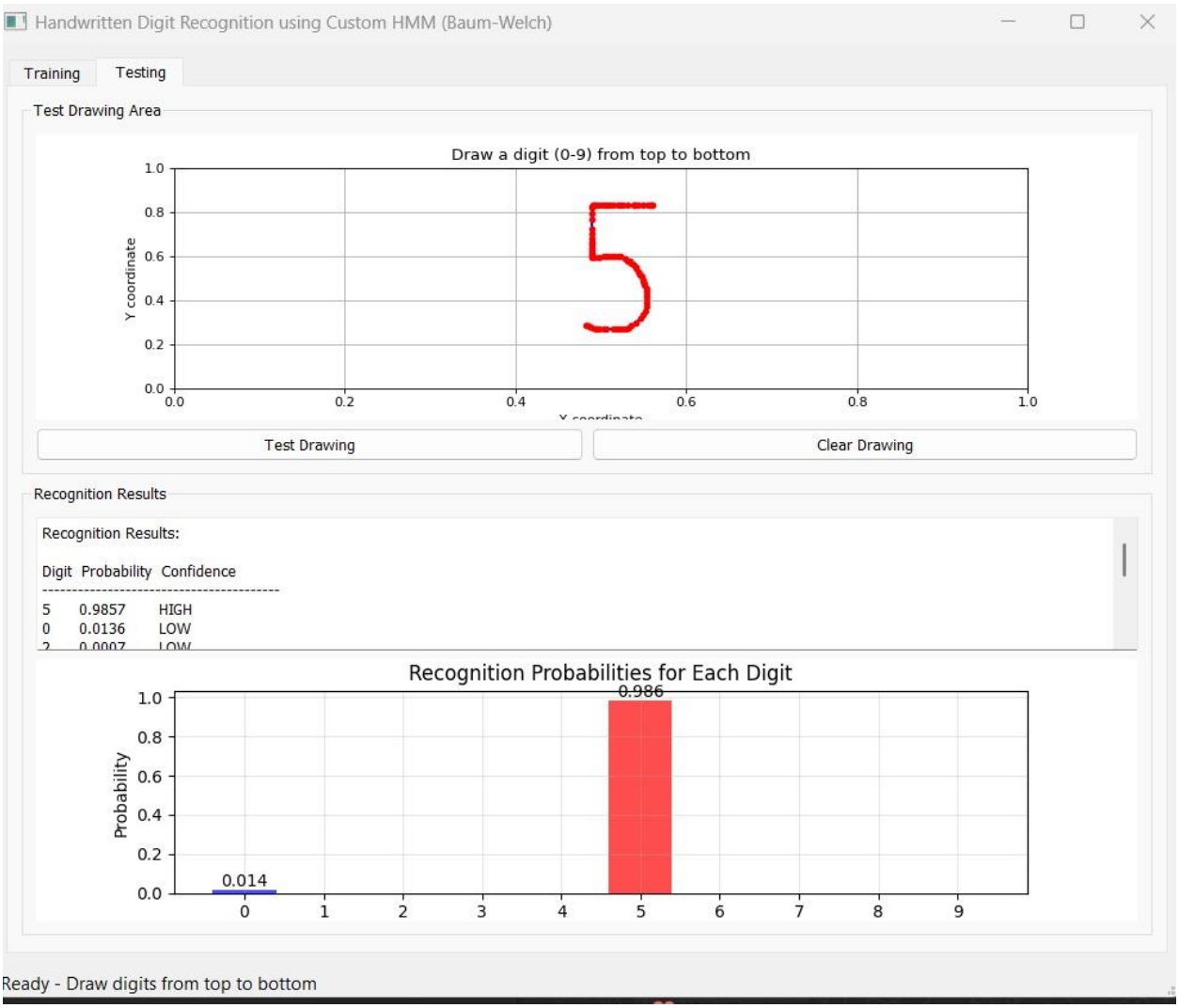
Here each number is stored as list of (X_i, Y_i) points and the points are numbers (From the drawing starts until it ends). Then we draw a vertical and horizontal lines that get across the center of the number. We can do this by:

$$\frac{\sum_i^n x_i}{n}$$

There is a model for each number. To know the model that the given observable sequence some from we will take the max probability among all models.



Output is like:



There are two main parameters to edit:

- Number of iterations.
- Number of components (states). The more it is the more details of number you can capture, but the more training data you must train with. Think of it as image quality.

The observed sequence differs from number to other based on its geometric shape. Now we can train our model.

The bam-welch output is:

- P (transition matrix).
- E (emission matrix).
- Π (initial state).

Expectation maximization

It is used to find values for P and E. We initialize P and E with any values and keep iterating until there is no enhancements in the parameters.

This the P that we all know:

$$P = \begin{bmatrix} p_{11} & p_{12} & p_{13} & \cdots & p_{1n} \\ p_{21} & p_{22} & p_{23} & \cdots & p_{2n} \\ p_{31} & p_{32} & p_{33} & \cdots & p_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ p_{n1} & p_{n2} & p_{n3} & \cdots & p_{nn} \end{bmatrix}$$

The emission matrix we also know

$$E = \begin{bmatrix} e_{11} & e_{12} & e_{13} & \cdots & e_{1k} \\ e_{21} & e_{22} & e_{23} & \cdots & e_{2k} \\ e_{31} & e_{32} & e_{33} & \cdots & e_{3k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ e_{n1} & e_{n2} & e_{n3} & \cdots & e_{nk} \end{bmatrix}$$

Number of hidden states = n, Number of observed states = k

forward probability $\alpha_i(t)$:

is the probability of being in state S_i at time t and having observed the sequence $O_1, O_2, \ldots, O_k$ so far.

We can calculate it as we do in section:

1. initialize

$$\alpha_1(j) = \pi_j b_j(y_1)$$

π_j: initial state distribution

$b_j(y_1)$: probability of observing y_1 given state j

2. For each time step

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(y_t)$$

$\sum_{i=1}^{N}$: sum over all states

a_{ij}: transition probabity

$b_j(y_t)$: probability of observing y_t given current state $= j$

$\alpha_{t-1}(i)$: probability of all previous observations give last state i

3. Result

$$P(Y|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

$P(Y|\lambda)$: all observations

$\alpha_T(i)$: final step

$\sum_{i=1}^{N}$: sum over all possible state

We know all the above till now from the lectures and sections content.

backward probabilities $\beta_{i+1}(t)$:

is the probability of seeing the future observations $O_{1+k}, O_{k+2}, \ldots, O_K$ given that we are currently in state S_i at time t. (reverse of the $\alpha_i(t)$).

$$\beta_i(T) = 1$$

$$\beta_i(t) = \sum_{j=1}^{N} a_{ij}\, b_j(y_{t+1})\, \beta_j\,(t+1) \quad \text{(For each time step from the end)}$$

$$P(Y|\lambda) = \sum_{j=1}^{N} \pi_j\, b_j(y_1)\, \beta_j(1)$$

The initial is 1 as it is the end, there is no chance to see more observations.

The last line of the above image is a bit tricky, As we sum the last probabilities in forward algorithm, we did that here:

Starting in state j, emitting y_1, and then going on to generate the entire future tail of the sequence [y_2, y_3, ..., y_T]. and so on in recursive approach.

Decoding (Viterbi algorithm):

We want to reverse the operation. This algorithm makes it. It shows us the most likely sequence of hidden states that produces the given sequence of observations.

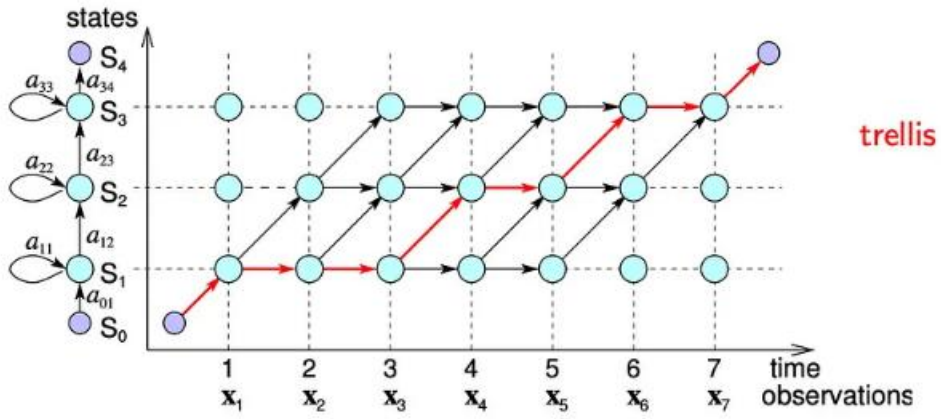
It is exactly the same calculations as forward algorithm but max instead of summation.

But why is that:

- Forward find the total probability from the beginning.
- Viterbi finds only the single best path that maximizes the probability to get the given sequence of observations. (If you love solving coding problems treat it as dp problem).

$$v_t(j) = \max_{s_0, s_1 \ldots s_{t-1}} P(s_0, s_1 \ldots s_{t-1}, x_1, x_2 \ldots x_t, S_t = s_j | \lambda)$$

$$v_t(j) = \max_{i=1}^{N} v_{t-1}(i)\, a_{ij}\, b_j(y_t)$$



$$p(\mathbf{X}, \text{path}_\ell \,|\, \boldsymbol{\lambda}) = p(\mathbf{X} \,|\, \text{path}_\ell, \boldsymbol{\lambda}) P(\text{path}_\ell \,|\, \boldsymbol{\lambda})$$

likelihood: $\sum_{\{\text{path}_\ell\}} p(\mathbf{X}, \text{path}_\ell \,|\, \boldsymbol{\lambda})$

decode: $\max_{\text{path}_\ell} p(\mathbf{X}, \text{path}_\ell \,|\, \boldsymbol{\lambda})$

This is how we will know the probability of the number if we have a sequence of hidden states.

Suddenly a new two stage variable comes from the sky:

- **$\xi_{ij}(t)$**-probability of transition from hidden state i to hidden state j at time t given observations:

$$\xi_{ij}(t) = P(X_t = i, X_{t+1} = j | Y, \theta)$$

$$= \frac{P(X_t = i, X_{t+1} = j, Y | \theta)}{P(Y|\theta)}$$

$$= \frac{\alpha_i(t) a_{ij} \beta_j(t+1) b_j(y_{t+1})}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i(t) a_{ij} \beta_j(t+1) b_j(y_{t+1})}$$

The yellow part as old. And the cyan to consider future observations. (AND operation = multiplication).

a = p, and b = e

- γ_i: the probability of being in state I given observations.

$$\gamma_i(t) = P(X_t = i|Y, \theta) = \frac{P(X_t = i, Y|\theta)}{P(Y|\theta)}$$

$$= \frac{\alpha_i(t)\beta_i(t)}{\sum_{j=1}^{N} \alpha_j(t)\beta_j(t)}$$

Why multiply B and alpha????
Now we are in the middle we are restricted with previous observations and future observations, so the must take past and future into consideration both.

But the doctor said "it depends on the present only"!!
The answer is that the future observations occur depending on the current present state (remember the question from the midterm exam and revision sheet).

Can also be written as:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i, j)$$

The hard part

Now we want to keep improving our P, E.

To update our transition matrix

$$\bar{a}_{ij} = \frac{\text{expected number of transitions from } S_i \text{ to } S_j}{\text{expected number of transitions from } S_i}$$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

But why this ??

From a_ij definition, it is the probability of moving from state I to state j in one unit of time, correct?

Now we are calculating normal probability by dividing expected number of transitions from I to j (**$\xi_{ij}(t)$ as described above**) by the total number of transitions starting from state I itself. The other states do mean anything here to us we must start from state I only.

To update the emission matrix's elements

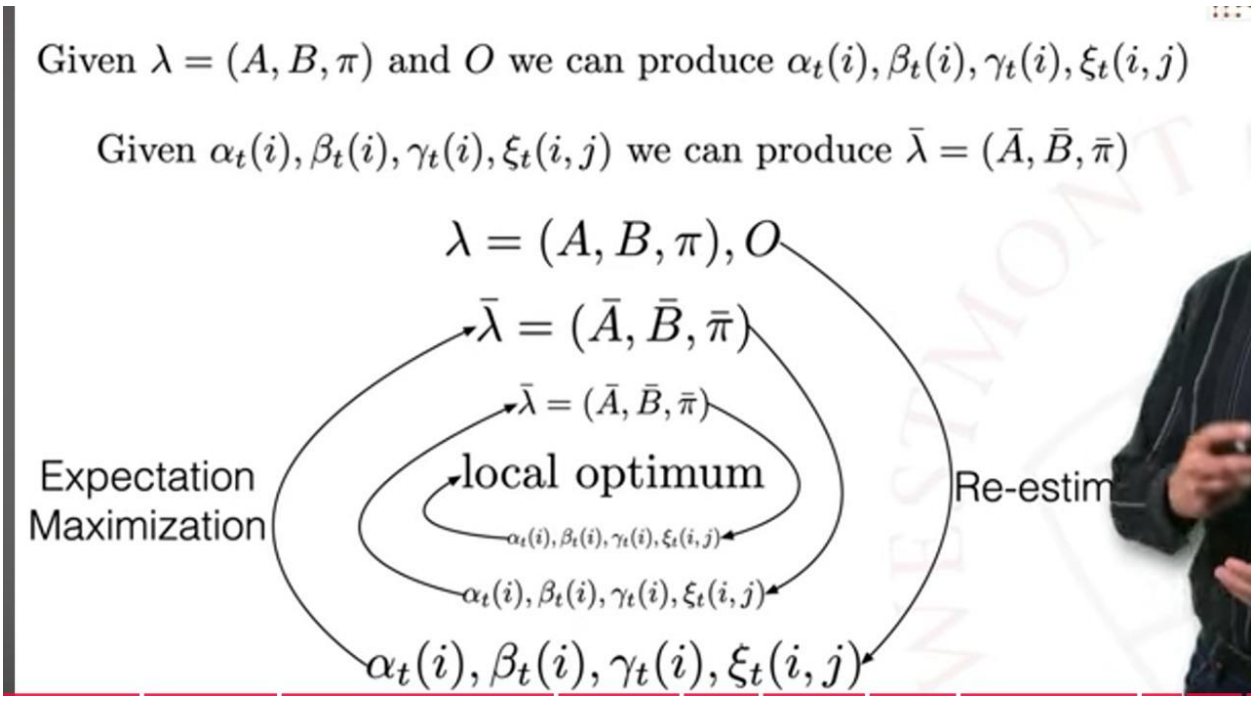
$$\bar{b}_j(k) = \frac{\text{expected number of times in state } j \text{ and observing } v_k}{\text{expected number of times in state j}}$$

$$\bar{b}_j(k) = \frac{\sum_{\substack{t=1 \\ s.t. O_t = v_k}}^{T} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)}$$

The same idea as above, but add condition on the summation above, as we will observe only the specified observation.

Imagine the value in the emission matrix.

Keep iterating



Summary

Estimation step

Forward

1. $\alpha_i(1) = \pi_i b_i(y_1)$,
2. $\alpha_i(t+1) = b_i(y_{t+1}) \sum_{j=1}^{N} \alpha_j(t) a_{ji}$.

Backward

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

Update

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

equals α (forward) × β (backward) for state i at time t

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

equals α for state i at time t × tranistion prob. between i and j × β for state j at time $t+1$ × observe y_{t+1} for state j

Maximization step

$$\pi_i^* = \gamma_i(1)$$

$$a_{ij}^* = \frac{\sum_{t=1}^{T-1} \xi_{ij}(t)}{\sum_{t=1}^{T-1} \gamma_i(t)}$$

$\xi_{ij}(t) = P(X_t = i, X_{t+1} = j|Y,\theta)$

$\gamma_i(t) = P(X_t = i|Y,\theta)$

$$b_i^*(v_k) = \frac{\sum_{t=1}^{T} 1_{y_t = v_k} \gamma_i(t)}{\sum_{t=1}^{T} \gamma_i(t)}$$

(sum γ over all time steps where the observation y_t is the same as v_t at time t)

where

$$1_{y_t = v_k} = \begin{cases} 1 & \text{if } y_t = v_k, \\ 0 & \text{otherwise} \end{cases}$$