

Hidden Markov Models

HMMs

Professor Marie Roch

Huang et al.: 8, 8.1-8.2

Dynamic programming

- Efficient algorithms for hidden Markov models (HMMs) depend upon dynamic programming algorithms.
- Before looking at HMMs, we will examine dynamic programming in the context of a simpler algorithm called dynamic time warping.

Goal of dynamic time warping

- One of the early speech recognition algorithms attempts to match a test utterance to a reference template.
- The test utterance is compared to the reference, with each portion compared by a distortion (distance) metric.

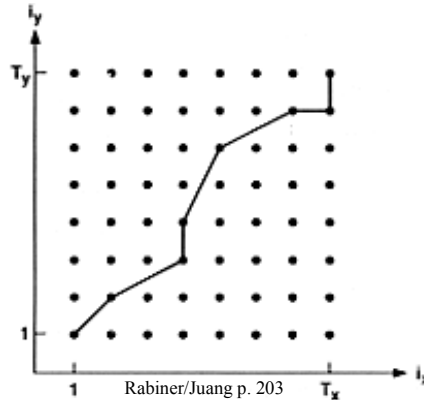
Time alignment

- Test utterances differ in length from reference template
- Before comparison, the test and reference must be aligned.
- Problems:
 - Nonuniform differences.
 - Simple compression/expansion will not work.

Time alignment

- In this example, the best matches are:
 - x_1, y_1 – x_4, y_4
 - x_2, y_2 – x_5, y_6
 - x_4, y_3 – etc.
- Where the best match for y_t is:

$$x_j : \min_{T_y \geq j \geq t} d(x_j, y_t)$$



Note: This diagram uses a slightly different version of DTW than in your text. The algorithm in your text only permits skips in the T_y direction, not in the T_x . This exact path would not be possible. 7

Time alignment

- Number of paths increases exponentially with the length of the utterances.
- Dynamic programming offers an efficient solution.

DTW

- In a serious attempt to use DTW, we would add a number of constraints, but for our purposes, we state the problem as follows:
- Find the path which minimizes the distortion between reference x and test y .

DTW data structures

- $D(i,j)$ will be used to denote the cumulative distortion between x_i and x_j and all pairs leading to that point.
- $B(i,j)$ will be called the backpointer, and will be used to recover the matching pairs. (More on this later).

DTW

- Initialization

- Assume that x and y start at the same place, i.e. x_1 and y_1 are the best match for each other.

- $D(1,1) = d(1,1)$, distortion between x_1 and y_1 .

- Make all other matches y_2, y_3, \dots for x_1 very costly:

- $D(1,j) = \infty$ for $j=2, \dots, T_y$

DTW

Iteration:

for $i=2, \dots, T_x$

for $j=1, 2, \dots, T_y$

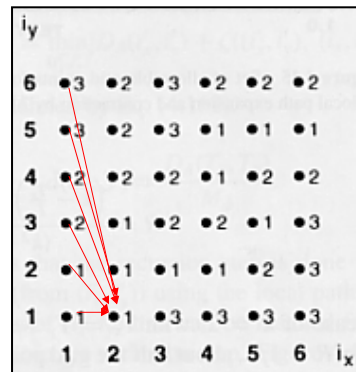
$D(i,j) = \min[D(i-1,p)+d(i,j)] \quad 1 \leq p \leq T_y$

$B(i,j) = \arg \min[D(i-1,p)+d(i,j)] \quad 1 \leq p \leq T_y$

Note: Your text authors use $d(p,j)$ which is incorrect. Typically, DTW uses a function $\zeta((i-1,p),(i,j))$ which includes the cost/penalty of moving from $(i-1,p)$ to (i,j) and $d(i,j)$.

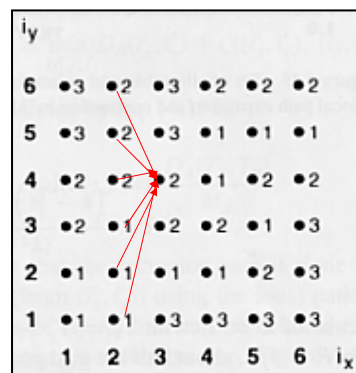
What's happening?

- $D(1,1) = 1$
- $D(2,1) = \min$ of:
 - $D(1,1)+d(2,1)=1+1=2$
 - $D(1,2)+d(2,1)=\infty$
 - $D(1,3)+d(2,1)=\infty$
 - ...
- $D(2,1) = 2$
- $B(2,1) = 1$ ($p=1$ produced min)



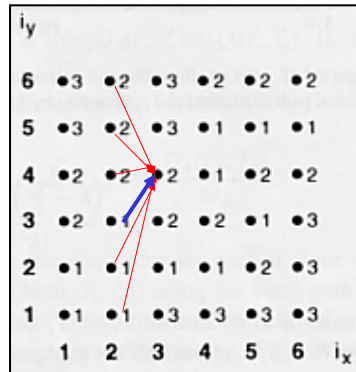
What's happening?

- For any given $i-1$, we know the cost of the shortest path to $(i-1, j')$ is $D(i-1, j')$.
- To determine the shortest path to (i, j) , we only need consider the distance from $D(i-1, j')$ + the cost of $d(i, j)$.



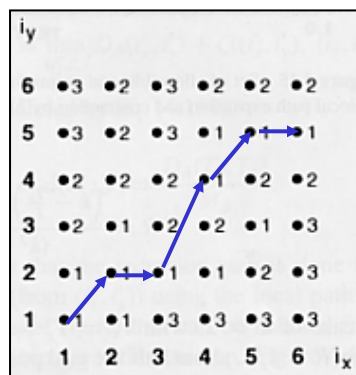
What's happening?

- When we have located the best path into $D(i,j)$, we note which j' produced that path.



Backtracking to recover the path

- When we have located the best path into $D(i,j)$, we note which j' produced that path.
- So here, the optimal distance, is $D(6,5)$.
 - $B(6,5) = 5$
 - $B(5,5) = 4$
 - $B(4,4) = 2$
 - $B(2,2) = 1$



Dynamic programming

- By incrementally performing our computation and retaining information, we have avoided a costly exponential problem.
- This technique will prove useful when dealing with hidden Markov models which we will now introduce.

A sequence of observations

- Let X be a sequence of random variables which we will call observations (vectors of features):

$$X = \{X_1, X_2, \dots, X_T\}$$

- The joint probability can be rewritten as:

$$\Pr(X_1, X_2, \dots, X_T) = \Pr(X_1) \prod_{i=2}^T \Pr(X_i | X_1, \dots, X_{i-1})$$

by repeated use of Bayes rule.

The Markov assumption

- Let us assume that X_i is only dependent upon X_{i-1} . That is:

$$\Pr(X_i | X_1, \dots, X_{i-1}) = \Pr(X_i | X_{i-1})$$

- then we can rewrite the probability as:

$$\Pr(X_1, X_2, \dots, X_T) = \Pr(X_1) \prod_{i=2}^T \Pr(X_i | X_{i-1})$$

- This is known as the Markov assumption

Markov chains

- Assume the random variable sequence can be associated with a finite set of states

$$\Omega = \{1, 2, \dots, N\}$$

- Let s_i denote the state associated with X_i .

$$s_i \in \Omega$$

Markov chains

- Sequence can be seen as moving from one state to another, dependent only upon the previous state:
 - $\Pr(\text{sunny} \mid \text{yesterday high, day before changing})$
 $= \Pr(\text{sunny} \mid \text{yesterday high})$



Low
pressure



Pressure
changing



High
pressure

State transition distribution

- Matrix A describes the state transition probabilities:

$$A = \begin{array}{c} \begin{array}{ccc} \text{transition to} \\ \text{low} & \text{changing} & \text{high} \\ \left[\begin{array}{ccc} \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{2} \end{array} \right] \\ \text{transition from} \\ \text{low} \\ \text{changing} \\ \text{high} \end{array} \end{array}$$

$$a_{ij} = \Pr(s_t = j \mid s_{t-1} = i)$$

$$\sum_{j=1}^N a_{ij} = 1$$

- $\Pr(\text{high} \mid \text{low}) = a_{13} = 1/4$

Initial state distribution

- The Markov chain has a probability of starting in an initial state, denoted by the vector π

$$\pi = \begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{bmatrix} \begin{matrix} \text{low} \\ \text{changing} \\ \text{high} \end{matrix}$$

- In this example, the starting state has a uniform distribution.

State-dependent distributions

- For each state s , there is a probability of seeing an observation x : $b_s(x)$
- For our weather model example:

$$b_1(x) = \begin{cases} \frac{3}{4} & x = \text{rain} \\ \frac{1}{4} & x = \text{sun} \end{cases} \quad b_2(x) = \begin{cases} \frac{1}{2} & x = \text{rain} \\ \frac{1}{2} & x = \text{sun} \end{cases} \quad b_3(x) = \begin{cases} \frac{1}{4} & x = \text{rain} \\ \frac{3}{4} & x = \text{sun} \end{cases}$$



What are the odds of that?

$$\Pr(X_1 = \text{rain}, S_1 = \text{changing}, X_2 = \text{sun}, S_2 = \text{high}, X_3 = \text{sun}, S_3 = \text{high})$$

by chaining Bayes rule $\Pr(AB) = \Pr(A | B) \Pr(B) \dots$

$$= \Pr(s_1) \Pr(x_1 | s_1) \Pr(s_2 | s_1, x_1) \Pr(x_2 | s_2, s_1, x_1) \Pr(s_3 | x_2, s_2, s_1, x_1) \Pr(x_3 | s_3, x_2, s_2, s_1, x_1)$$

Recall x_i and x_j are assumed independent when $i \neq j$

$$= \Pr(s_1) \Pr(x_1 | s_1) \Pr(s_2 | s_1, x_1) \Pr(x_2 | s_2, s_1) \Pr(s_3 | x_2, s_2, s_1, x_1) \Pr(x_3 | s_3, s_2, s_1)$$

State transition is only dependent upon previous state (Markov property order 1)

$$= \Pr(s_1) \Pr(x_1 | s_1) \Pr(s_2 | s_1) \Pr(x_2 | s_2, s_1) \Pr(s_3 | s_2) \Pr(x_3 | s_3, s_2, s_1)$$

Observation probabilities depend on current state

$$= \Pr(s_1) \Pr(x_1 | s_1) \Pr(s_2 | s_1) \Pr(x_2 | s_2) \Pr(s_3 | s_2) \Pr(x_3 | s_3)$$

$$= \pi_2 b_2(\text{rain}) a_{23} b_3(\text{sun}) a_{33} b_3(\text{sun})$$

The hidden in hidden Markov model

- In our previous example, the barometer let us observe the state.
- What happens if we cannot observe what state we are in?
- Then many state sequences are possible, each of which have a probability of occurrence.

A bit of notation

$O = \{o_1, o_2, \dots, o_M\}$ – Set of observation symbols, i.e $O = \{\text{rain, sun}\}$

- Not to be confused with X which is a sequence of observations: $x_1 = o_3, x_2 = o_{14}, x_3 = o_2, \dots$
- As speech feature vectors are usually continuous, we map continuous feature vectors to discrete ones by using vector quantization.

A bit of notation

• $\Omega = \{1, 2, \dots, N\}$ – State set.

- s_t denotes state at time t .
- Thus to $s_1 = 1, s_2 = 3, s_3 = 3$ would represent:
 - start in state 1
 - transition to state 3
 - self transition to state 3

HMM

- Let $\Phi(A,B,\pi)$ denote a HMM where:
 - A – $N \times N$ state transition matrix. a_{ij} denotes the probability of transitioning from state i to j .
 - B – $\{b_j(k)\}$ – Set of state-dependent probability distributions. $1 \leq j \leq N$, k in O
 - π – Initial state distribution. π_j is the probability of starting in state j . $1 \leq j \leq N$

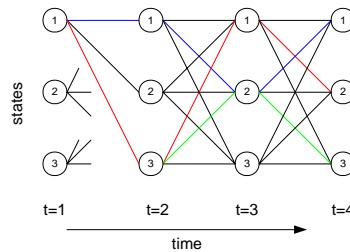
Top 3 List for HMMs

1. What is the probability of a given sequence X given model Φ ?
2. What state sequence is most likely to account for X in model Φ ?
3. How can we improve the parameters of Φ to better account for X ?

TOP 3 LIST: PROBLEM 1

Probability Evaluation

- Must evaluate all paths through model
- Naive approach is exponential!



Probability evaluation

- Dynamic programming can be used
- Two algorithms
 - Forward procedure

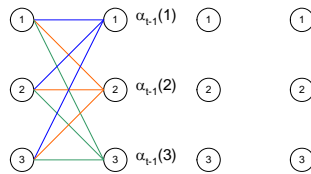
$$\alpha_t(i) = \Pr(x_1, x_2, \dots, x_t, s_t = i | \Phi)$$

- Backward procedure

$$\beta_t(i) = \Pr(x_{t+1}, x_{t+2}, \dots, x_T, s_t = i | \phi)$$

Forward algorithm

- Suppose we know the sum of all paths leading into each state j at time $t-1$:

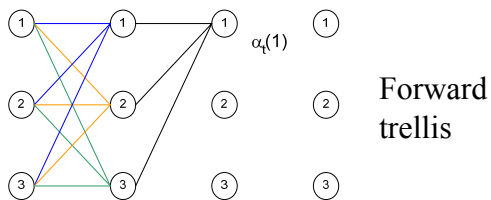


$$\alpha_{t-1}(j) = \Pr(x_1, x_2, \dots, x_{t-1}, s_{t-1} = j \mid \Phi)$$

Forward algorithm

- Then we can compute the probability of all paths leading into time t .

$$\alpha_t(i) = \Pr(x_1, x_2, \dots, x_t, s_t = i \mid \Phi)$$



Forward algorithm - $O(N^2T)$

- Initialization

$$\alpha_1(i) = \pi_i b_i(X_1) \quad 1 \leq i \leq N$$

- Induction

$$\alpha_t(j) = \left[\sum_{i=1}^N \alpha_{t-1}(i) a_{ij} \right] b_j(X_t) \quad \begin{array}{l} 1 \leq j \leq N \\ 2 \leq t \leq T \end{array}$$

- Termination

$$\Pr(X | \Phi) = \sum_{i=1}^N \alpha_T(i)$$

TOP 3 LIST: PROBLEM 2

Optimal State Sequence

- The forward algorithm finds all paths through a model.
- Sometimes, we are interested in the best path through the model:
 - Perhaps we are interested in determining which states are associated with which observations.
 - Frequently, most of the paths contribute very little to the overall probability.

Viterbi algorithm:

Determine optimal state sequence

- Another example of dynamic programming
- Finds the most likely path through the model
- Similar to the forward algorithm
 - Uses max instead of sum
 - Keeps extra information about the best path, similar to dynamic time warping.

Viterbi Algorithm

- Initialization
 - Probability to start in state i
- $$V_1(i) = \pi_i b_i(x_1) \quad 1 \leq i \leq N$$
- $B_t(i)$ – The previous state which transitioned into state i at time $t-1$. (0 indicates no previous state.)

$$B_1(i) = 0 \quad 1 \leq i \leq N$$

Viterbi Algorithm $O(N^2T)$

- Induction

- Find the best path leading into the current state and account for the probability of the observation

$$V_t(j) = \max_{1 \leq i \leq N} [V_{t-1}(i) a_{ij}] b_j(X_t) \quad \begin{array}{l} 1 \leq j \leq N \\ 2 \leq t \leq T \end{array}$$

- Record the previous node on the best path

$$B_t(j) = \arg \max_{1 \leq i \leq N} [V_{t-1}(i) a_{ij}] \quad \begin{array}{l} 1 \leq j \leq N \\ 2 \leq t \leq T \end{array}$$

Viterbi Algorithm

- Termination

$$\Pr^{BestPath}(X) = \max_{1 \leq i \leq N} [V_T(i)]$$

$$s_T^{BestPath} = \arg \max_{1 \leq i \leq N} [V_T(i)]$$

Note: Huang, Acero, and Hon have a typo on the termination step and write $\arg \max B_T$ instead of V_T (p. 388).

- Extracting the best path:

for $t = T - 1, T - 2, \dots, 1$

$$s_t^{BestPath} = B_{t+1}(s_{t+1}^{BestPath})$$

Log Domain Viterbi Implementation

- Operates in the log probability domain
- Multiplications are replaced with addition
- Not covered in text.

TOP 3 LIST: PROBLEM 3

Parameter Estimation

- Application of the Expectation-
Maximization (EM) algorithm
 - If we had all the information
 - true state sequence
 - observations
 - then techniques such as maximum-likelihood
estimation could be used to improve our
parameter set

EM algorithm

- Problem: Some information is unknown
- Solution:
 - Use the expectation operator to determine the expected values of missing parameters
 - Determine new parameters
 - Repeat

EM Algorithm

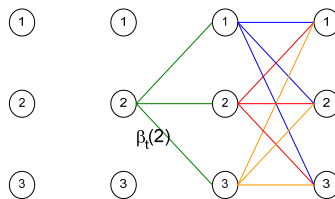
- Guaranteed to converge to a local maximum
- For speech applications, no more than 5-15 iterations are typically required
- For HMMs, the resulting formulas are known as the Baum-Welch reestimation equations.

Some needed concepts

- The backward algorithm – similar to the forward algorithm, but works from T down towards 1.
- $\gamma_t(i,j)$ – Given a model and observation sequence, the probability of transitioning from state i to j at time t given the model and acoustic evidence X

Backward algorithm

$$\beta_t(i) = \Pr(X_{t+1}, X_{t+2}, \dots, X_T \mid s_t = i, \Phi)$$



Note: $\beta_t(i)$ does not include the probability of observing X_t .

Backward Algorithm $O(N^2T)$

- Initialization

$$\beta_T(i) = 1 \quad 1 \leq i \leq N$$

Note: Huang, Acero, and Hon initialize to $1/N$ (p. 389) but most authors initialize to 1. In practice, it makes little difference.

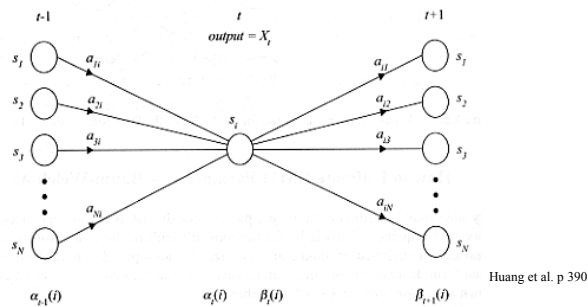
- Induction

$$\beta_t(i) = \sum_{j=1}^N a_{ij} b_j(x_{t+1}) \beta_{t+1}(j) \quad \begin{matrix} 1 \leq j \leq N \\ 1 \leq t \leq T-1 \end{matrix}$$

- Termination not needed, but possible

Forward-Backward relationship

- $\alpha_t(i)$ = all paths into $s_t=i$ and observing X_1, X_2, \dots, X_t .
- $\beta_t(i)$ = all paths out of $s_t=i$ and observing $X_{t+1}, X_{t+2}, \dots, X_T$.

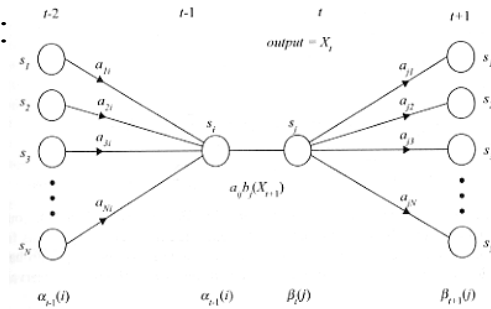


Constrained path probability

- Suppose we want to know the probability of all paths going through a specific transition from time t-1 to t:

$$\Pr(X, s_{t-1} = i, s_t = j | \Phi)$$

$$= \alpha_{t-1}(i) a_{ij} b_j(X_t) \beta_t(j)$$



Huang et al. p 391
Note: Left most column should read α_{t-2} , not α_{t-1} .
49



$\gamma_t(i, j)$

- $\gamma_t(i, j)$ – probability of a specific state transition at a specified time

$$\gamma_t(i, j) \stackrel{\Delta}{=} \Pr(s_{t-1} = i, s_t = j | X, \Phi)$$

$$= \frac{\Pr(s_{t-1} = i, s_t = j, X | \Phi)}{P(X | \Phi)} \quad \text{Bayes rule}$$

$$= \frac{\alpha_{t-1}(i) a_{ij} b_j(X_t) \beta_t(j)}{\sum_{k=1}^N \alpha_T(k)} \quad \text{previous slide, and}$$

$$P(X | \Phi) = \sum_{k=1}^N \alpha_T(k)$$



Special case: $\gamma_1(i, j)$

- For convenience, we will define $\alpha_0(i)=1$ and the transition between s_0 and s_1 as π_1 :

$$\begin{aligned}\gamma_1(i, j) &= \frac{\alpha_0(i)a_{ij}b_j(X_1)\beta_1(j)}{\sum_{k=1}^N \alpha_T(k)} \\ &= \frac{\pi_j b_j(X_1)\beta_1(j)}{\sum_{k=1}^N \alpha_T(k)}\end{aligned}$$

Baum-Welch equations

- The EM algorithm can be used to derive the Baum-Welch reestimation equations.
- Computation of the expectations is done with the γ function.
- Once the expectation has been computed, a maximum likelihood estimate can be computed for the model.

Initial state distribution

- Percentage of time that we will be in state i at time $t=1$

$$\hat{\pi}_i = \sum_{k=1}^N \gamma_1(i, k)$$

- For many speech applications, we desire a starting state. In this case $\pi_{\text{start}}=1$. The reestimation formulas will not change this.

State transition

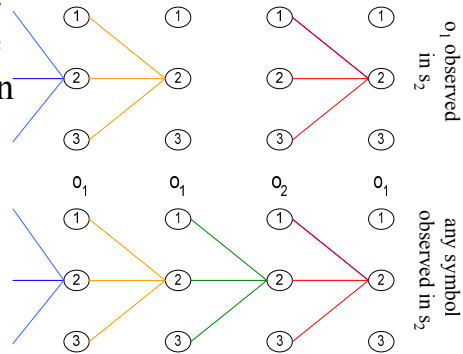
- We can think of this as the expected number of transitions from state i to j divided by the expected number of all transitions:

$$\hat{a}_{ij} = \frac{\sum_{t=2}^T \gamma_t(i, j)}{\sum_{t=2}^T \sum_{k=1}^N \gamma_t(i, k)}$$

Note: Huang, Acero, and Hon sum from 1 to N (eq. 8.40, p 392) which includes the initial state probability π . Most authors do not do this.

State-dependent pdfs

- For each time where symbol o_k is seen, we need to determine the probability of being in state j and seeing o_k .
- We divide this expectation by the expected probability of being in state j .



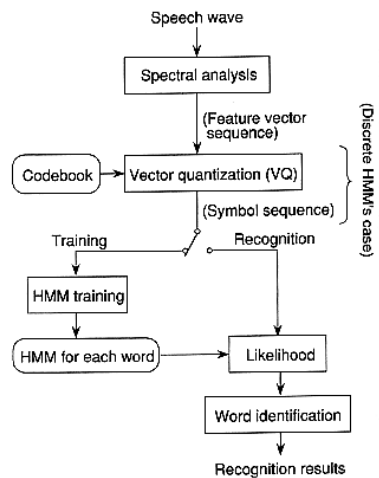
State-dependent pdfs

- Divide the expected number of transitions into state j where symbol o_k occurs by all transitions into state j :

$$\hat{b}_j(k) = \frac{\sum_{t=1}^T \sum_{i=1}^N \gamma_t(i, j) \delta(X_t, o_k)}{\sum_{t=1}^T \sum_{i=1}^N \gamma_t(i, j)}$$

$$= \frac{\sum_{t \text{ such that } X_t = o_k} \sum_{i=1}^N \gamma_t(i, j)}{\sum_{t=1}^T \sum_{i=1}^N \gamma_t(i, j)}$$

Isolated Word Recognizer



Furui 2001, p. 284

Multiple observations

- Single training token usually insufficient
- Use multiple tokens to capture diversity
- $X = [X^{(1)}, X^{(2)}, \dots, X^{(M)}]$ where

$$X^{(i)} = [x_1^{(i)}, x_2^{(i)}, \dots, x_{T^{(i)}}^{(i)}]$$

- Baum-Welch equations can be extended in a straight-forward manner.

Multiple observations

- Let $\gamma_t^m(i, j)$ denote the expectations associated with the m^{th} training sequence.

- Then:

$$\hat{a}_{ij} = \frac{\sum_{m=1}^M \sum_{t=2}^{T^m} \gamma_t^m(i, j)}{\sum_{m=1}^M \sum_{t=2}^{T^m} \sum_{k=1}^N \gamma_t^m(i, k)}$$

- Other equations extended similarly.