Hidden Markov Model

Instructor: Tom Ko

Objectives

- HMM
- State alignment in ASR
- Why use phonemes as modeling unit?
- State tying in ASR

Let's apply what we have learnt

- Right now, we have already learnt feature extraction of speech and basic classification theory.
- Can we apply them in speech recognition?

Start from the most simplest case

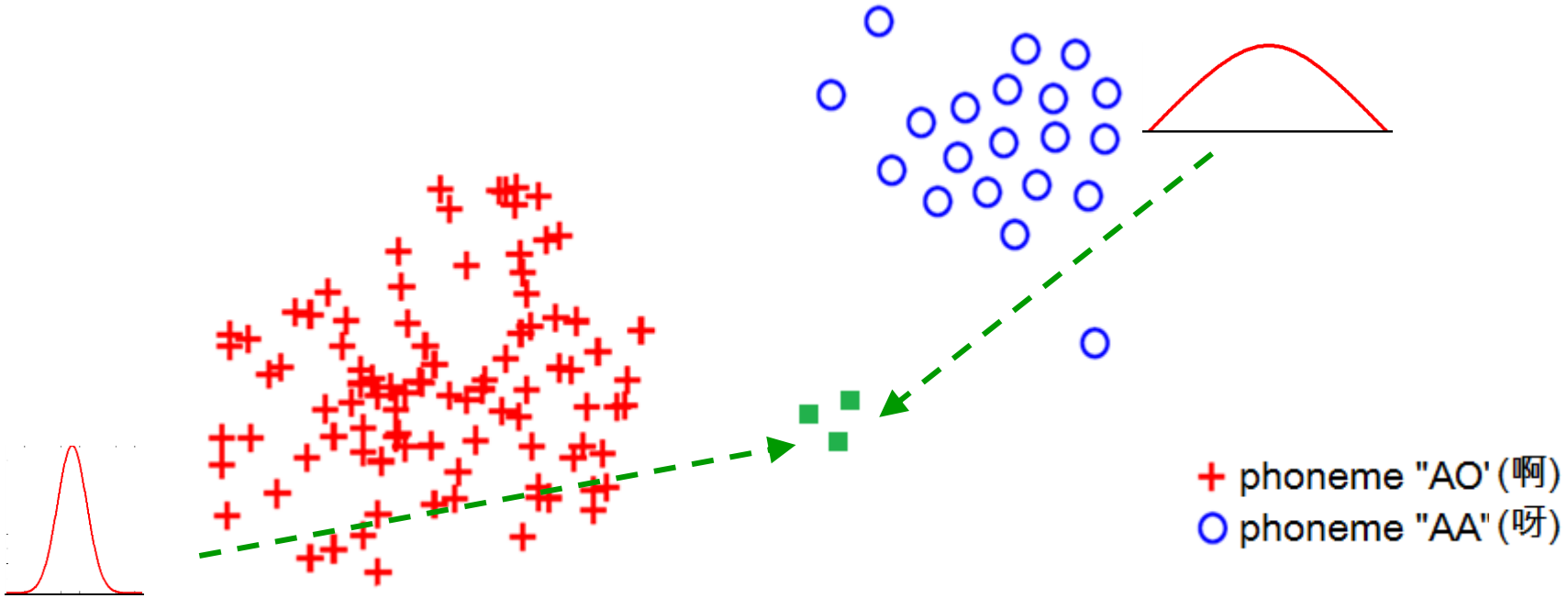
- Assume that there are only two sound units to classify: AO(啊) and AA(呀)
- Let x be one of the testing samples, we want to compute P(AO|x) and P(AA|x)

$$P(\omega_j|x) = \frac{p(x|\omega_j)P(\omega_j)}{p(x)},$$

- Thus we need P(x|AO)P(AO) and P(x|AA)P(AA)
- For the likelihood terms, we use Gaussian models:

$$p(x|\omega_1) = \frac{1}{\sqrt{2\pi}\sigma_1} \exp\left[-\frac{1}{2}\left(\frac{x-\mu_1}{\sigma_1}\right)^2\right]$$

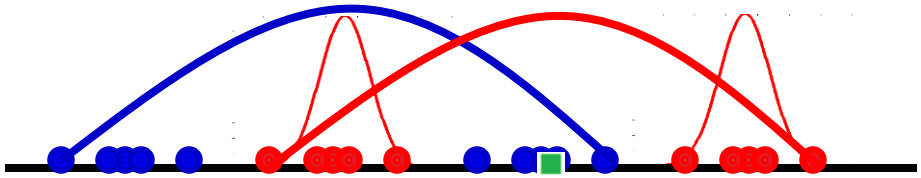
Start from the most simplest case



- Obtain P(AO) and P(AA) by simple counting
- Build a Gaussian model for each class: $N(\mu_1,\sigma_1^2)$ and $N(\mu_2,\sigma_2^2)$
- Compute P(x|AO) and P(x|AA)
- Make decision depending on P(AO|x) and P(AA|x)

Fitting the data

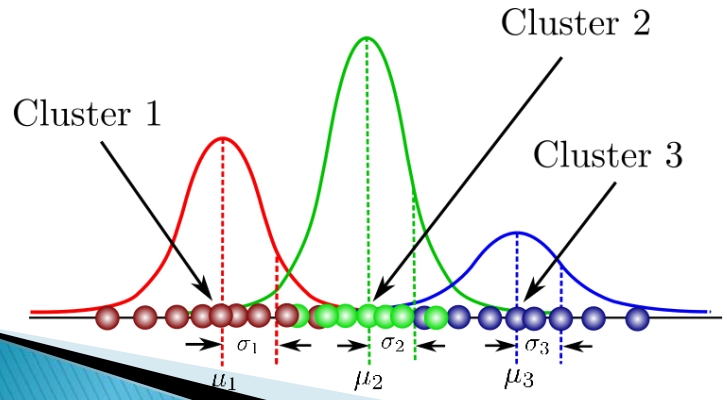
- In Bayesian approach, we learn the distribution of each class separately.
- We want to learn the truth distribution of the data to have a better estimate of likelihood.

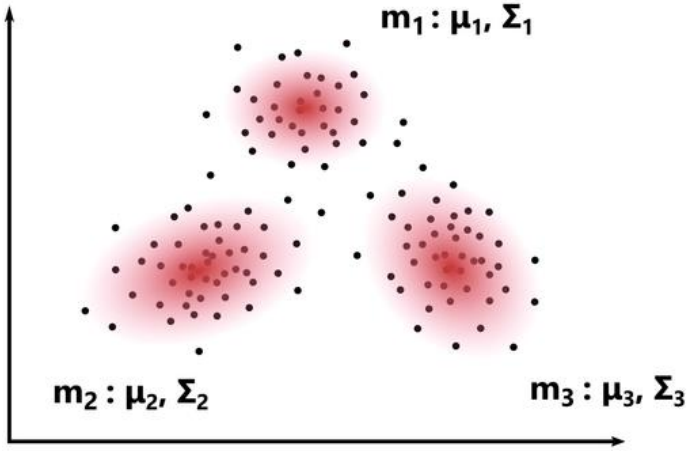


Gaussian mixture model (GMM)

- For most of the time, a single Gaussian is not enough to model the true distribution of the class.
- Thus we use Gaussian mixture model (GMM)
- The likelihood formulation becomes:

$$p(x|\omega_1) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\ x\ |\mu_k, \Sigma_k)$$





The problems

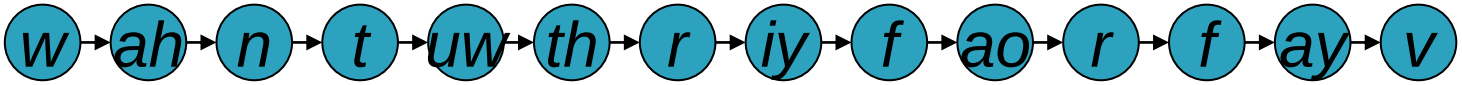
- If we are just doing single sound classification, that's all.
- How to group up class samples from a long sentence?
- How to recognize a word and a whole sentence?

Training of Acoustic Models

ONE TWO THREE FOUR FIVE.

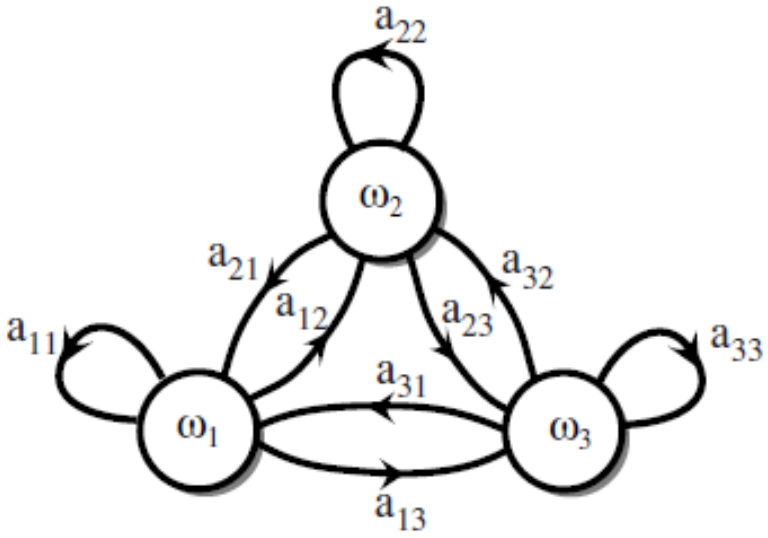
Dictionary	
ONE	*w ah n*
TWO	*t uw*
THREE	*th r iy*
FOUR	*f ao r*
FIVE	*f ay v*

w ah n t uw th r iy f ao r f ay v



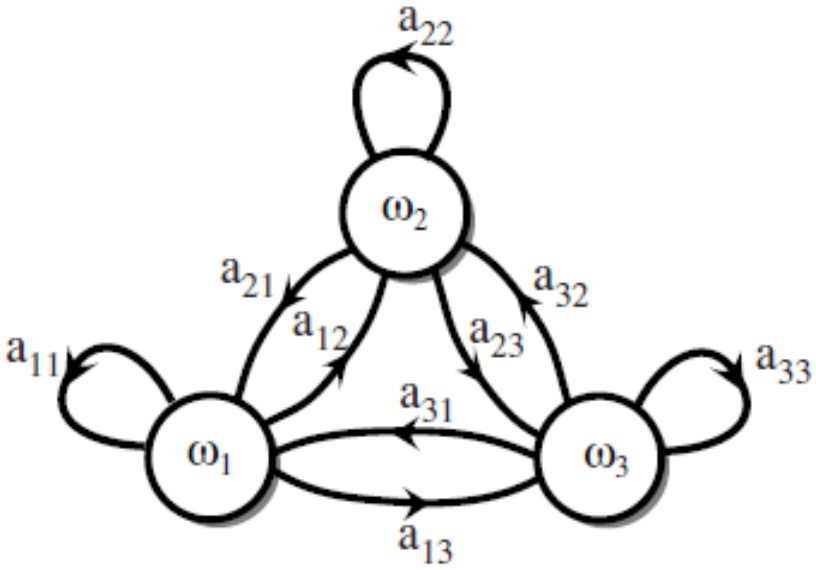
Markov Model

- We consider a sequence of states at successive times; the state at any time t is denoted $\omega(t)$.
- A particular sequence of length T is denoted by $\boldsymbol{\omega}^T = \{\omega(1), \omega(2), \ldots, \omega(T)\}$ as
  - for instance $\{\omega_1, \omega_3, \omega_2, \omega_1, \omega_2, \omega_3\}$.
  - A state can be revisited at different steps, including successive steps (self-loop)
  - Not every states need to be visited



Markov Model

- a_{ij} is the transition probability of transferring to state j given that it is currently at state i
  - No need to be symmetric (a_{ij} != a_{ji})
- When it is assumed that the probability of transferring to next state only depends on the current state, it is a first-order Markov model

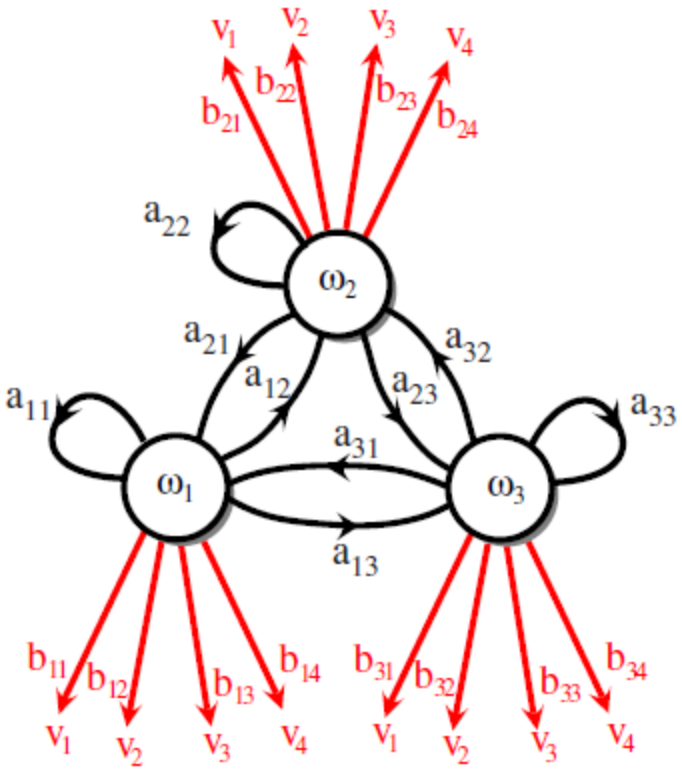


Markov Model

- Given a Markov model θ — that is, the full set of a_{ij}
- Let say the initial state is always ω_1, what is the probability of state sequence$\{\omega_1, \omega_3, \omega_2, \omega_1, \omega_2, \omega_3\}$?
- For the production of spoken words, the state sequence for the word "cat" would be {/k/, /a/, /t/}
- However, in speech recognition, we can not observe the state transition.
- Instead, we measure some properties of the emitted sound.

Hidden Markov Model (HMM)

- The state sequence $\boldsymbol{\omega}^T$ is not directly visible.
- The emission $\boldsymbol{V}^T$, dependent on the state is visible.
- Here, it is assumed V(t) are discrete observations with K choices.



The formulation

$$a_{ij} = P(\omega_j(t+1)|\omega_i(t))$$

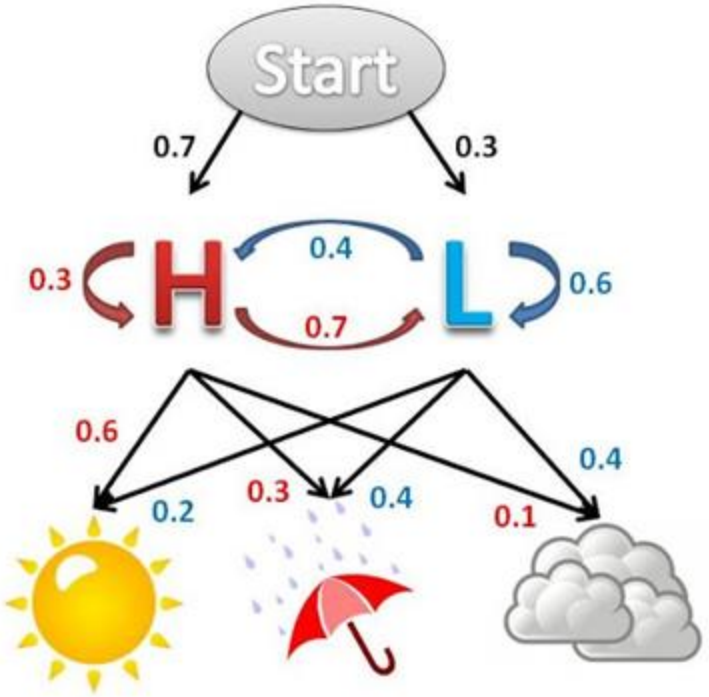
$$b_{jk} = P(v_k(t)|\omega_j(t))$$

$$\sum_j a_{ij} = 1 \text{ for all } i$$

$$\sum_k b_{jk} = 1 \text{ for all } j$$

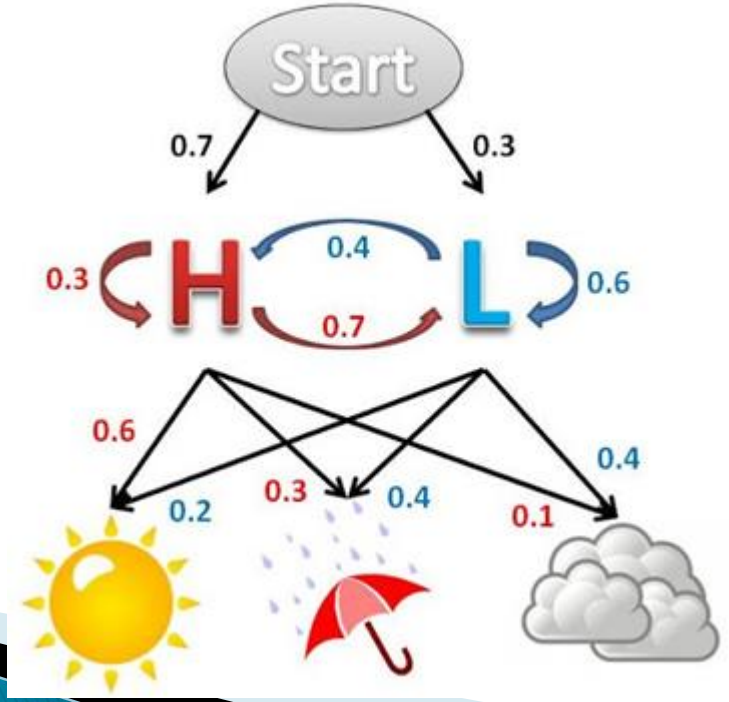
The weather example

- The pressure in the air, either high or low, is hidden.
- The weather conditions are visible.
- Each time step represents a day.



The weather example

- Given the following HMM model with its parameters known, what is the probability of having 5 sunny days in a row?
- If the underline pressure sequence is HLHHL, the probability will be
  - 0.6 x 0.2 x 0.6 x 0.6 x 0.2 = 0.00864
- We have only considered one particular sequence.
- We have to consider all possible sequences, how to do that?



Hidden Markov Model

- **The Evaluation problem.**
  - Suppose we have an HMM with all its parameters known. Determine the probability that a particular sequence of visible states V^T was generated by that model.
- **The Decoding problem.**
  - Suppose we have an HMM as well as a set of observations V^T. Determine the most likely sequence of *hidden* states ω^T that led to those observations.
- **The Learning problem.**
  - Suppose we are given the structure of a model (the number of states and the number of visible states) but *not* the probabilities a_{ij} and b_{jk}. Given a set of training observations of visible symbols, determine these parameters.

The evaluation problem of HMM

- Based on the first-order assumption,

$$P(\mathbf{V}^T) = \sum_{r=1}^{r_{max}} P(\mathbf{V}^T|\omega_r^T)P(\omega_r^T)$$

$$P(\omega_r^T) = \prod_{t=1}^{T} P(\omega(t)|\omega(t-1))$$

$$P(\mathbf{V}^T|\omega_r^T) = \prod_{t=1}^{T} P(v(t)|\omega(t))$$

$$P(\mathbf{V}^T) = \sum_{r=1}^{r_{max}} \prod_{t=1}^{T} P(v(t)|\omega(t))P(\omega(t)|\omega(t-1))$$

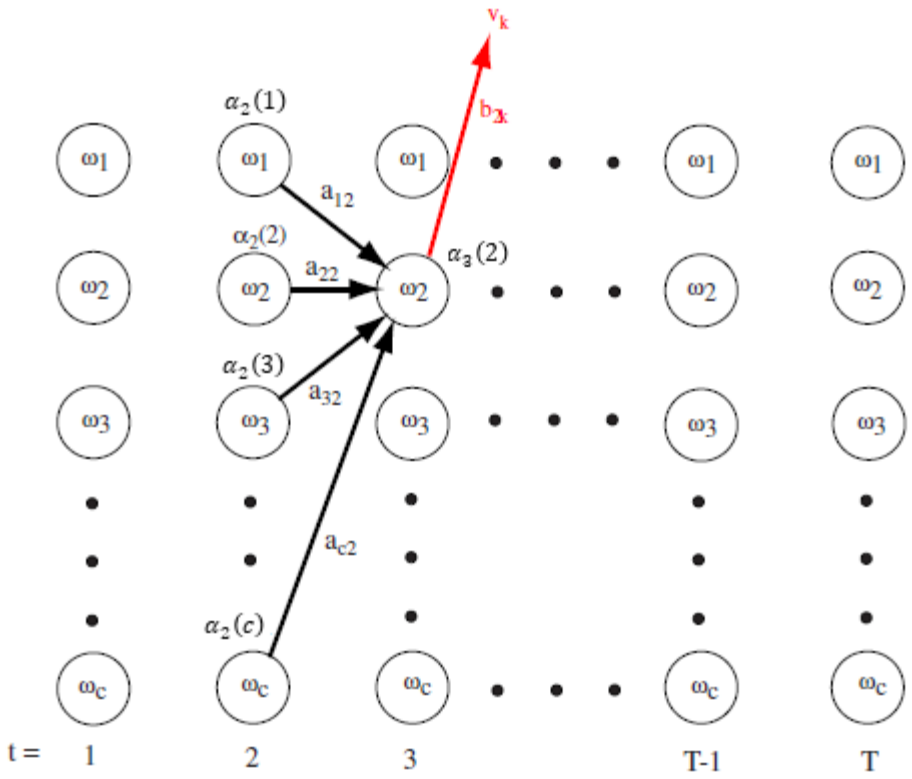
The evaluation problem of HMM

- In the weather example, how many possible state sequences for 5 days?
  - 32 possible state sequences, each state sequence will contribute to the overall probability of 5 sunny days.
  - For particular sequence HLHHL, the probability is 0.6 x 0.2 x 0.6 x 0.6 x 0.2 = 0.00864
  - What is the probability for sequence HLHHH ?
  - The probability is 0.6 x 0.2 x 0.6 x 0.6 x 0.6 = 0.02592
- It is tedious to compute for each possible sequence.
- Obviously, there is a lot repeated computation for sequence HLHHL and HLHHH

The Forward Algorithm

- Define $\alpha_t(j)$ to be the probability of that the HMM is in hidden state j at step t having generated the first t elements of V^T
- Initialization
  - $\alpha_1(j) = \pi_j b_j(x_1)$
- Induction
  - $\alpha_t(j) = \left[\sum_{i=1}^{N} \alpha_{t-1}(i)\, a_{ij}\right] b_j(x_t)$
- Termination
  - $P(V^T|\theta) = \sum_{i=1}^{N} \alpha_T(i)$

The Forward Algorithm



The decoding problem of HMM

- Given a sequence of observations and a model, what is the most likely sequence of states that produced the sequence.

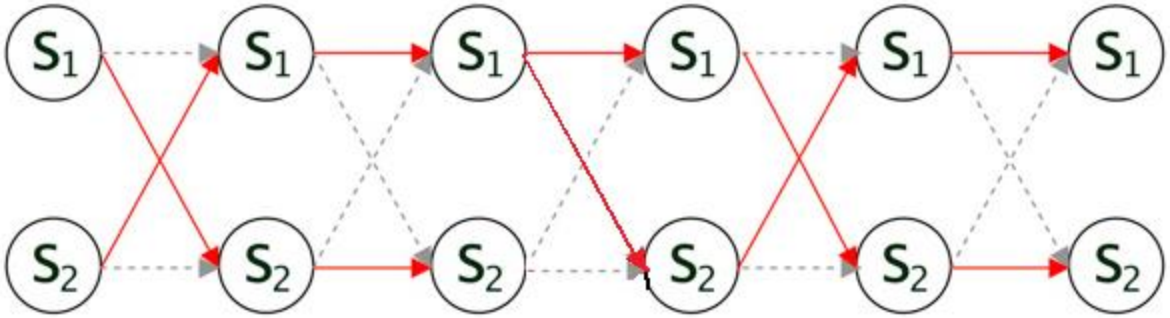
- The forward algorithm computes the probability that an HMM generates an observation sequence by summing up the probabilities of all possible paths, so it does not provide the best path (or state sequence).

Viterbi Algorithm

- Similar to the Forward algorithm
- Initialization
  - $V_1(j) = \pi_j b_j(x_1)$
  - $B_1(j) = 0$
- Induction
  - $V_t(j) = \max_i [V_{t-1}(i) a_{ij}]\, b_j(x_t)$
  - $B_t(j) = \operatorname{argmax}_i [V_{t-1}(i) a_{ij}]$
- Termination
  - The best score $= \max_i [V_T(i)]$
  - $S_T = \operatorname{argmax}_i [V_T(i)]$
- Backtracking
  - $S_t = B_{t+1}(S_{t+1})$
  - $S^* = \{S_1, S_2, S_3 \ldots S_T\}$ is the best path

Viterbi Algorithm

- $V_t(j)$ is the highest probability of having come from the best path of that the HMM is in hidden state j at step t having generated the first t elements of V^T



The learning problem of HMM

- Given a sequence of observations, find the set of model parameters that best fits the data.
- Let's simplify the problem a little bit: Given a sequence of observations and **the corresponding state sequence**, find the set of model parameters that best fits the data.
- "best fits" => Maximum likelihood estimation
- Model parameters in HMM => transition probabilities and emission probabilities

The learning problem of HMM

- In the weather example, given that an observation sequence is {Sunny, Rainy, Sunny, Rainy, Sunny} and the corresponding state sequence is HLHHL
- Collect the **sufficient statistic** and estimate the transition probabilities and emission probabilities separately
- transition probabilities:
  - H -> L = 2 / 3
  - H -> H = 1 / 3
  - L -> L = 0 / 1
  - L -> H = 1 / 1
- emission probabilities:
  - P(Sunny | H) = 2 / 3
  - P(Rainy | H) = 1 / 3
  - P(Sunny | L) = 1 / 2
  - P(Rainy | L) = 1 / 2

Sufficient statistic

- A sufficient statistic is a statistic that summarizes all of the information in a sample about a chosen parameter.
- For example, let's say you have the simple data set 1,2,3,4,5. You want to calculate the mean which should be (Sum of x) / (Number of data points)
- In this case,
  - Sum of x = 1 + 2 + 3 + 4 + 5 = 15
  - Number of data points = 5
  - are the sufficient statistic of the mean estimation
- After computing sum of x, the individual values of x are not needed
- Then a new sample of x, say, 6 is observed, after updating the sufficient statistic, 6 do not have to be memorized.

The learning problem of HMM

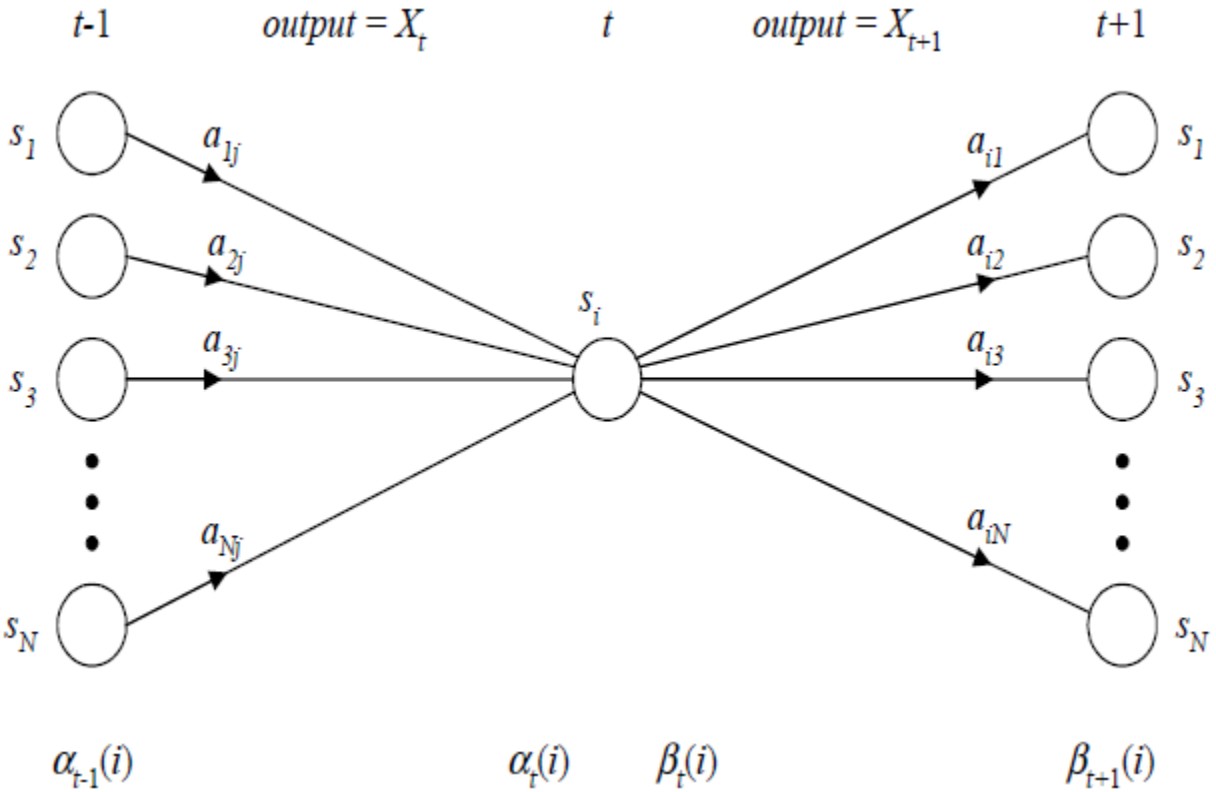
- Given a sequence of observations, find the set of model parameters that best fits the data.
- The problem is that we don't know **the corresponding state sequence.**
- This is by far the most difficult of the three problems, because there is no known analytical method that maximizes the joint probability of the training data in a closed form.
- The data is **incomplete** because of the hidden state sequence.
- We have to use the same principle as that of the **EM algorithm**.

The Backward Algorithm

- Define $\beta_t(j)$ to be the probability of that the HMM is in hidden state ***j*** at step ***t*** having generated the last (***T- t)*** elements of V^T
- Initialization (for normalizing the probabilities)
  - $\beta_T(j) = 1/N$
- Induction
  - $\beta_t(j) = \sum_{i=1}^{N} a_{ij}\, b_j(x_{t+1})\, \beta_{t+1}(i)$

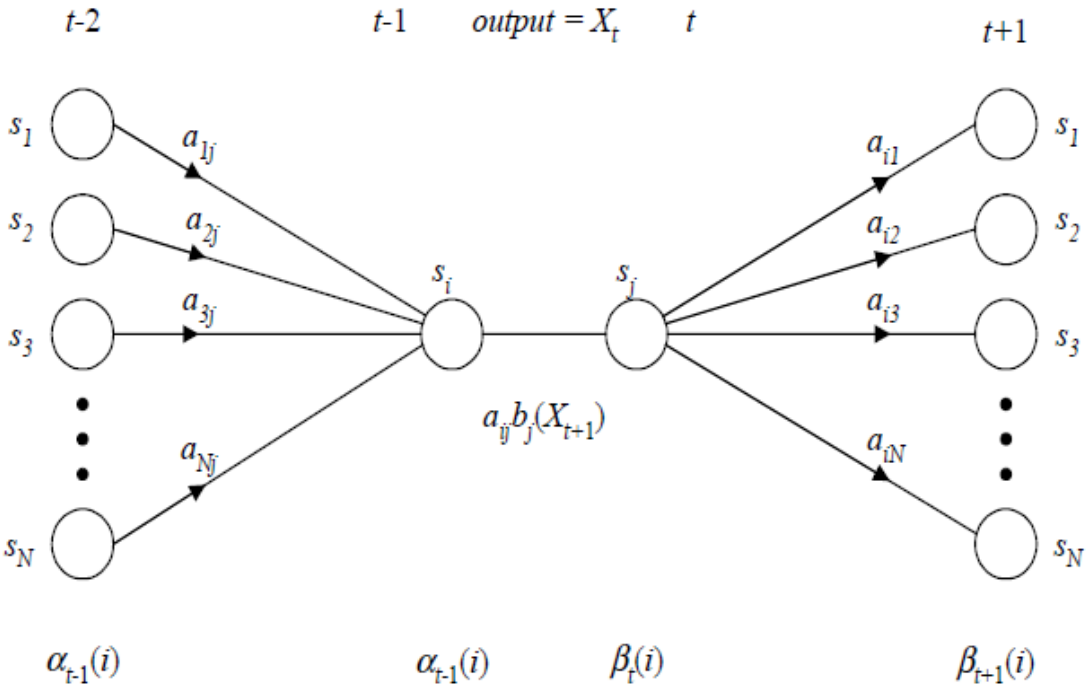
The Forward-backward algorithm

- It is the algorithm of applying the Forward algorithm and backward algorithm together.
- What is the meaning of $\alpha_t(j)\,\beta_t(j)$?
  - It is the probability of that the HMM is in hidden state *j* at step ***t*** having generated the whole sequence of V^T



The Forward-backward algorithm

- Now we define $\gamma_t(i,j)$ which is the probability of taking the transition from state *i* to state *j* at time *t*, given the observation sequence
- $$\gamma_t(i,j) = \frac{\alpha_{t-1}(i)a_{ij}b_j(x_t)\ \beta_t(j)}{P\left(V^T \middle| \theta\right)}$$



The Forward-backward algorithm

- Computing the $\gamma_t(i,j)$ is the E-step of the EM algorithm, now we should update the model parameters, which is the M-step of the EM algorithm.
- Update the model parameters:
  - $\hat{a}_{ij} = \frac{\sum_{t=1}^{T} \gamma_t(i,j)}{\sum_{t=1}^{T} \sum_{k=1}^{N} \gamma_t(i,k)}$
  - $\hat{b}_j(x_t) = \frac{\sum_{t=1}^{T} x_t \sum_{i=1}^{N} \gamma_t(i,j)}{\sum_{t=1}^{T} \sum_{i=1}^{N} \gamma_t(i,j)} = \frac{\sum_{t=1}^{T} x_t \alpha_t(j)\, \beta_t(j)}{\sum_{t=1}^{T} \alpha_t(j)\, \beta_t(j)}$

Baum-Welch algorithm

- The Forward-backward algorithm is also known as *Baum-Welch* algorithm.
- To be more precise, Baum-Welch is an expectation-maximization algorithm that iteratively uses the forward-backward algorithm
- After the parameters are updated, $\gamma_t(i,j)$ are recomputed
- $\gamma_t(i,j)$ is the soft count from the consideration of all possible paths
- EM algorithm is not perfect, if helps to guess the incomplete part of the data
- If we have more hints on the hidden state sequences, the training will be more effective

HMM reading list

- *Spoken Language Processing: A Guide to Theory, Algorithm and System Development, ch 8.2*

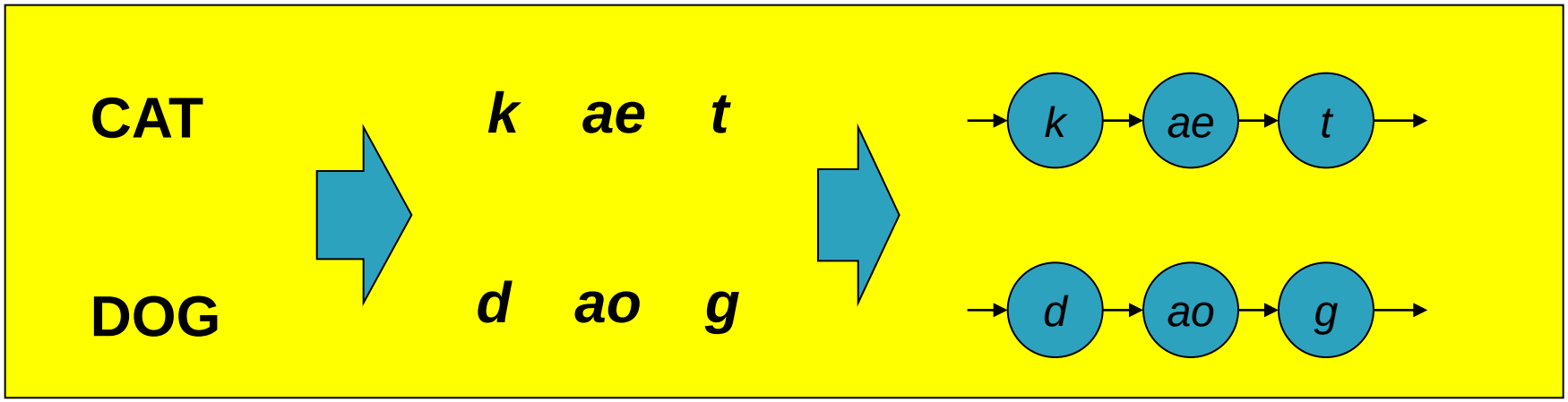
Applying the HMM in ASR

- The decoding problem
  - Given the model and the MFCC sequences of a testing utterance, knowing the best path means knowing the acoustic unit sequence. We can guess the word sequence concurrently.
- The learning problem
  - Given the MFCC sequences and the corresponding transcription of a training utterance, we want to train the model.
  - Although the transcription of the utterance is known, we don't know the alignment of the MFCC sequences on each HMM state

Recognition

Feature vector sequence
($X = x_1, x_2, \ldots, x_T$)

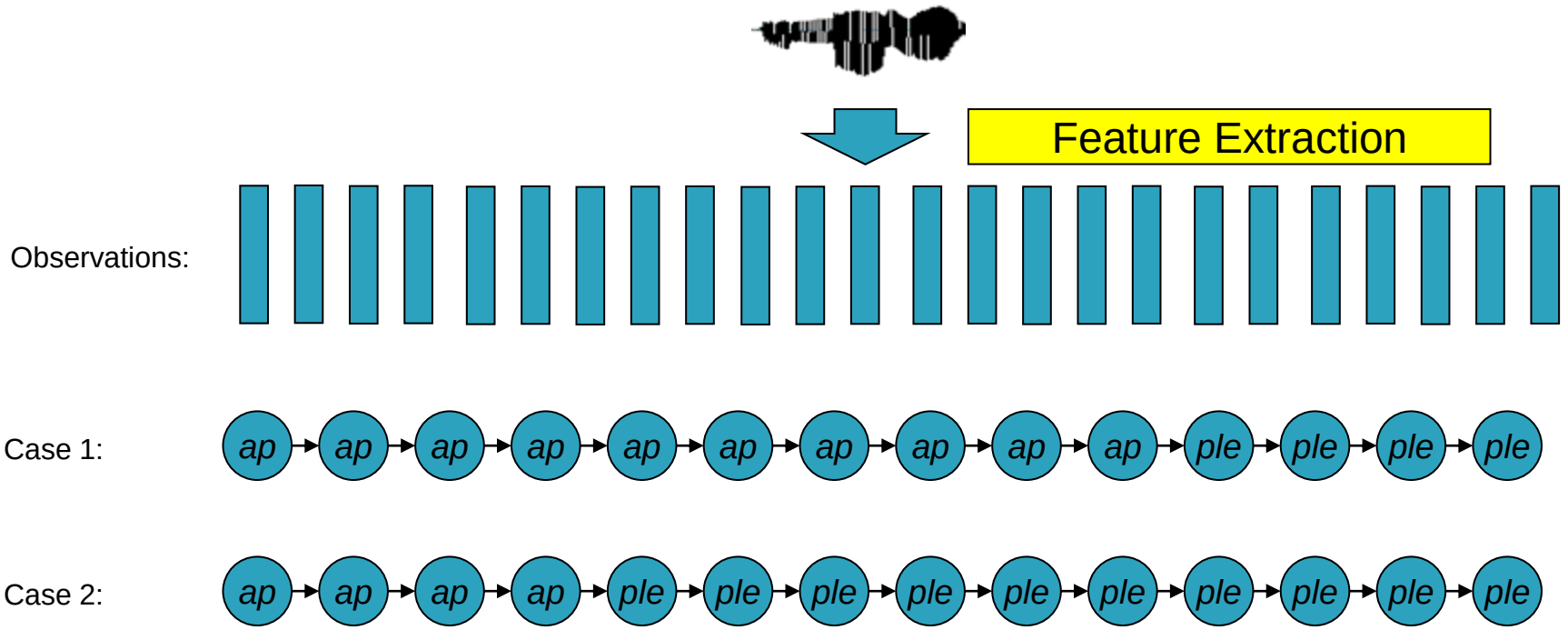
Recognizer



Dog

State alignment in ASR

- Given a 1-sec audio, resulting in 100 MFCCs, with a transcription of word “apple”



Out-of-vocabulary

- If we have no training data of a particular class, can we recognize it from other classes?
- In English, there are a total of a million words, about 170,000 are in current use, and 20,000-30,000 words used by each individual person.
- In Chinese, there are about 50,000 Chinese characters, about 6000 of them are commonly used.
- It is a fact that not every one of them will appear in the training set, especially the rare word. Then how to recognize them in the test set?

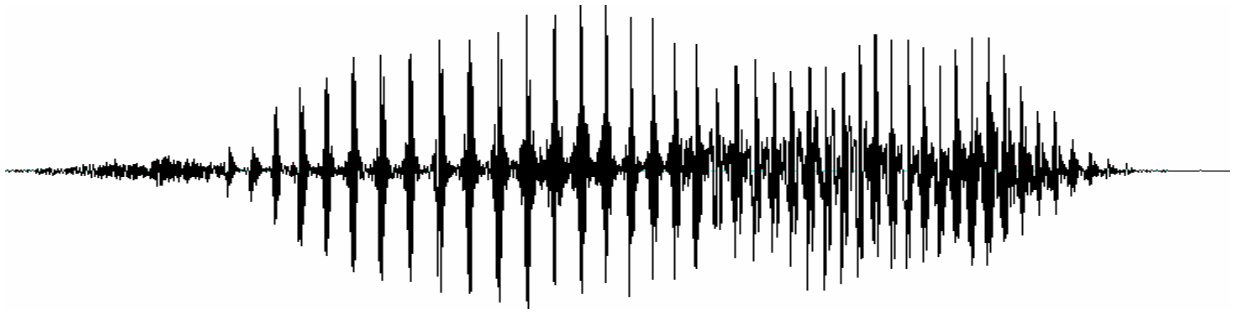
Phone-based modeling

- Phonemes are the minimal speech units in a language that can serve to distinguish one word from another.
  - There are about 40-60 phonemes in English
- Phones are the acoustic realization of phonemes.
- Every words can be transcribed into their phonetic transcription to label their pronunciation.
- Phone-based modeling is the most popular way in doing acoustic modeling.
- Examples: [aa], [ay], [t], [s] ….etc

Dictionary	
consist	*k ah n s ih s t*
invert	*ih n v er t*
convert	*k ah n v er t*

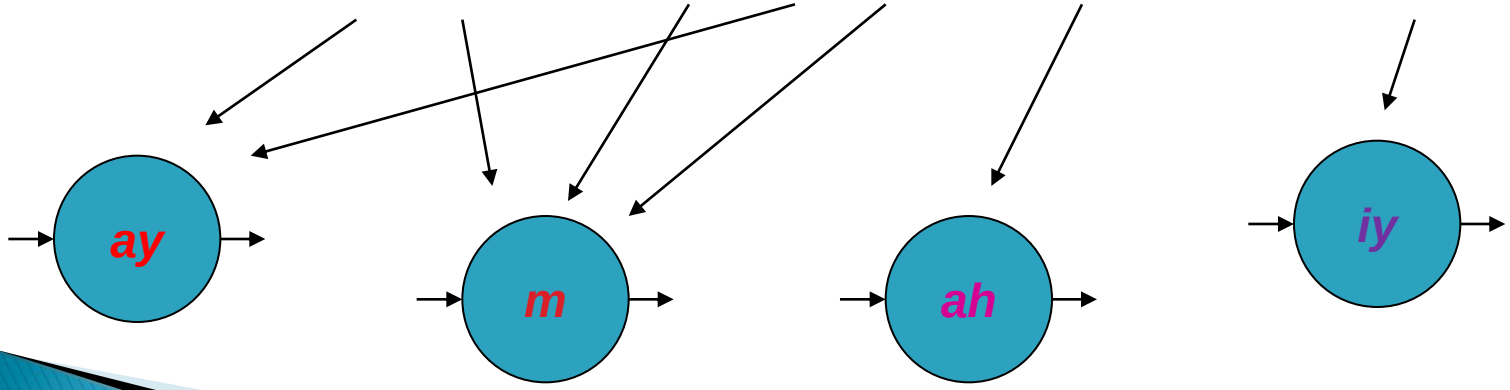
Dictionary	
高	*g ao1*
張	*zh ang1*
岡	*g ang1*
招	*zh ao1*

Training of Acoustic Models



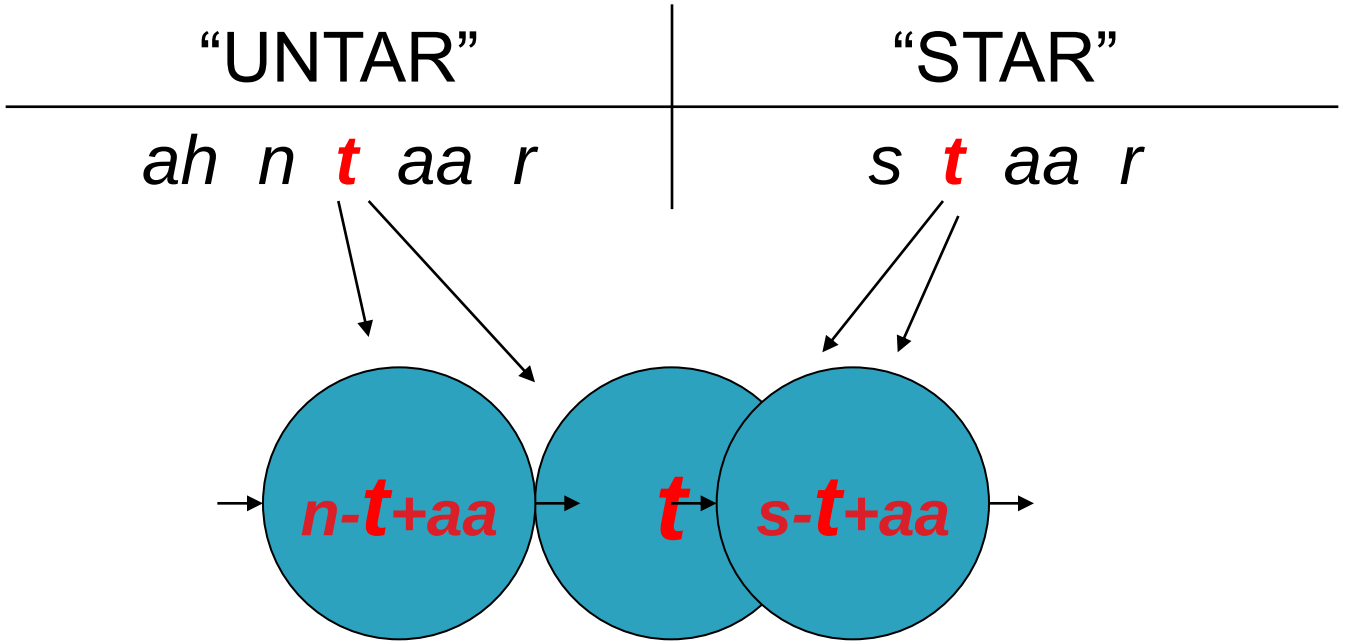
I'm Tom, I'm hungry.

ay m t aa m ay m hh ah ng g r iy



Context Dependence

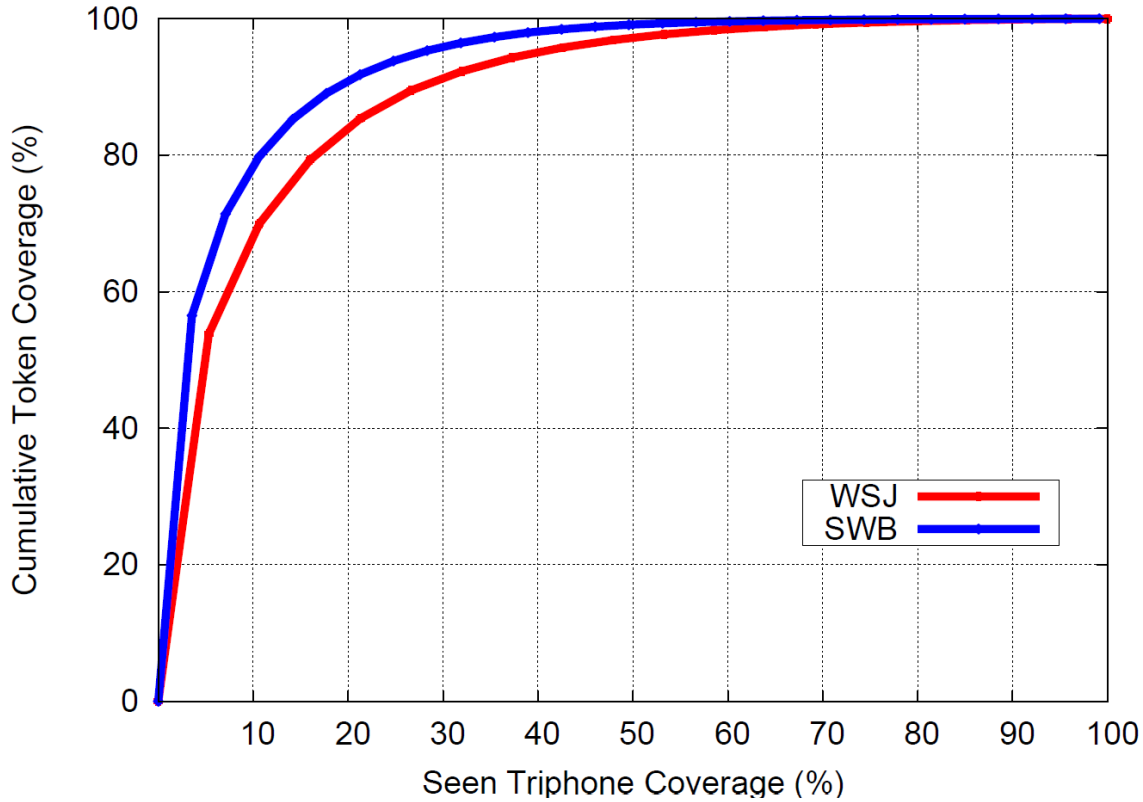
- Observation: In human speech, the acoustic behavior of a phone is influenced by its neighboring phones.



Triphones as Modeling Units

- In 1980s, triphones become the most popular modeling units in ASR.
- If there are N monophones, there will be N^3 triphones.
- Drawback of using triphones:
  - Estimation of infrequent triphone models is not robust.

Data Sparsity in Triphone Modeling



- Wall Street Journal (WSJ): 80% of samples consist of the most frequent 20% of triphones
- Switchboard (SWB): 90% of samples consist of the most frequent 20% of triphones

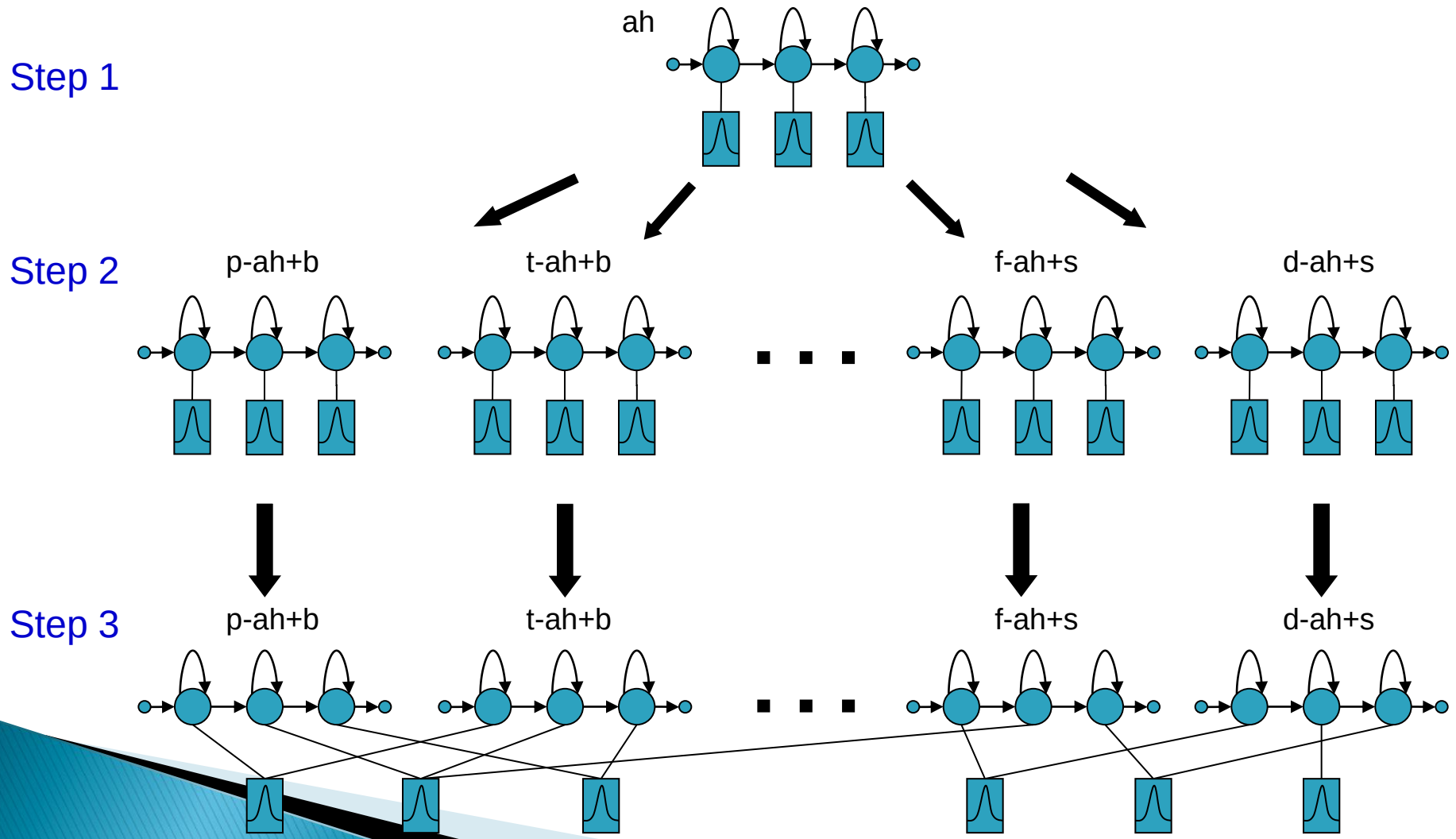
Solution: Parameter Tying

- Basic idea: The parameters of infrequent triphones and frequent triphones are tied together.

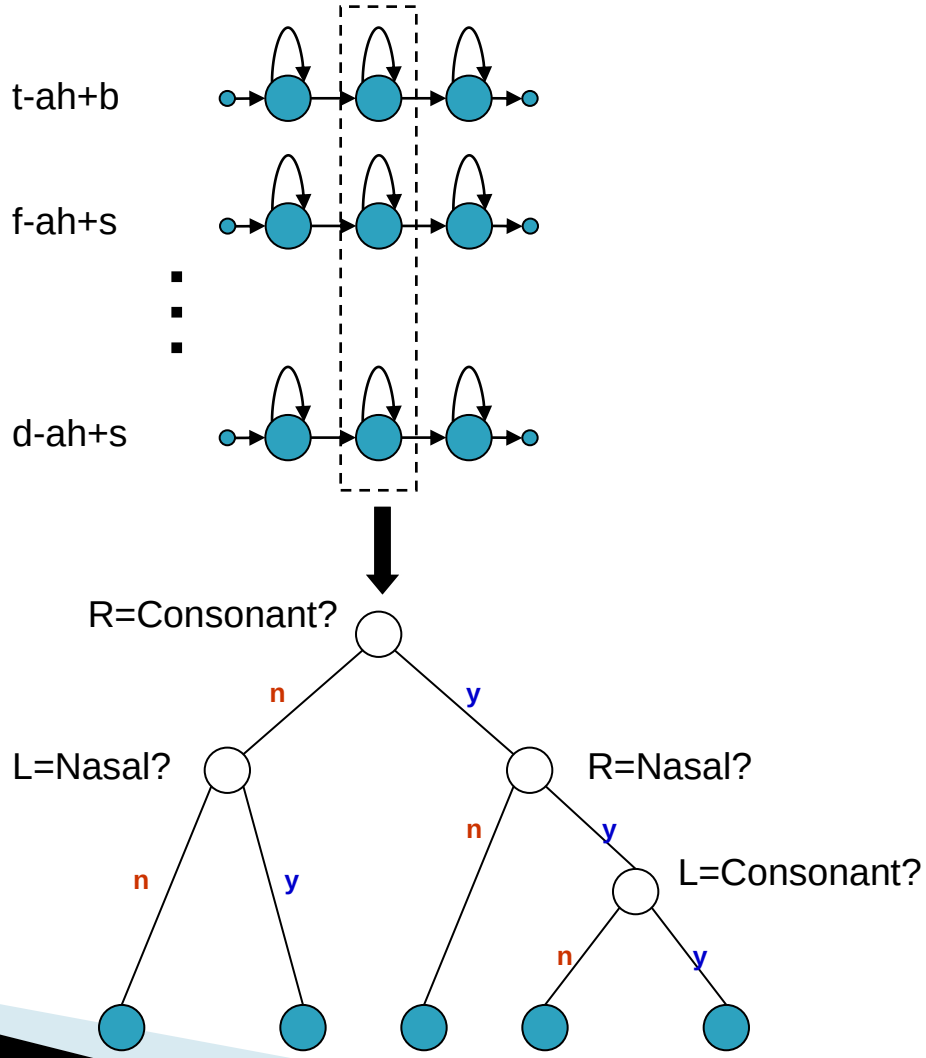
- Advantages:
  - Good recognition performance
  - Reducing model size and enhancing speed

- Things to consider:
  - Level of tying: model, state or distribution level
  - Which one to tie with?

State Tying



Phonetic Decision Tree-based Clustering



Reading list

- Pattern Classification, Chapter 3.10