GMM-HMM in ASR

孟令辉

2020/02/24-2020/03/01

中国科学院
自动化研究所
INSTITUTE OF AUTOMATION
CHINESE ACADEMY OF SCIENCES

Overview

- General structure of conventional ASR
- Introduction of Gaussian mixture models
- Introduction of HMM
- HMM algorithm
- Workflow of GMM-HMM in ASR
- Context-dependent phone models

General structure of conventional ASR

General structure of conventional asr

$$\mathbf{W}^* = \arg\max_{\mathbf{W}} P(\mathbf{W} \,|\, \mathbf{X})$$

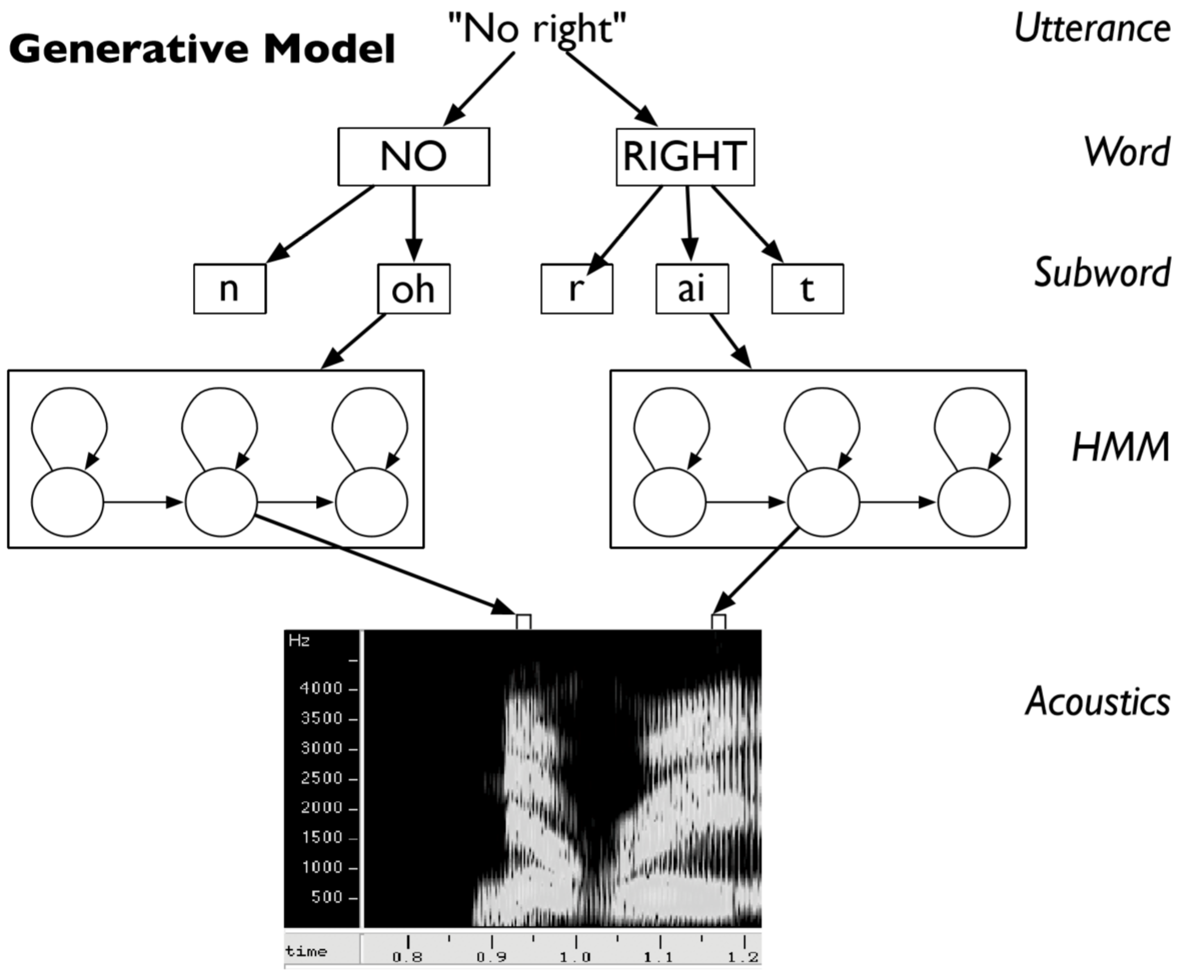
The Bayes theory:

$$P(\mathbf{W} \,|\, \mathbf{X}) = \frac{p(\mathbf{X} \,|\, \mathbf{W})\, P(\mathbf{W})}{p(\mathbf{X})}$$

$$\propto p(\mathbf{X} \,|\, \mathbf{W})\, P(\mathbf{W})$$

$$\mathbf{W}^* = \arg\max_{\mathbf{W}} \underbrace{p(\mathbf{X} \,|\, \mathbf{W})}_{\substack{\text{Acoustic} \\ \text{model}}} \; \underbrace{P(\mathbf{W})}_{\substack{\text{Language} \\ \text{model}}}$$

Hierarchical modelling of speech



Introduction of Gaussian mixture models

中国科学院
自动化研究所
INSTITUTE OF AUTOMATION
CHINESE ACADEMY OF SCIENCES

Calculation of marginal prob *P(X|W)*

P(X_*|/s/) is modeled by emission prob (Generally is Gaussian distribution to fit)

$P(X|/sayonara/) \approx P(X_1|/s/)\, P(X_2|/a/) \ldots\ldots P(X_8|/a/)$

Extreme Case: each state is independent and fixed

✗ *In general case where a phone lasts more than one frame and model parameters change over time, we need to employ HMM*

A single Gaussian distribution function for example: (2 params: μ and σ_s)

$$P(x|/s/) = \frac{1}{\sqrt{2\pi\sigma_s^2}} e^{-\frac{(x-\mu_s)^2}{2\sigma_s^2}}$$

$$\hat{\mu}_s = \frac{1}{N}\sum_{i=1}^{N} x_i, \qquad \hat{\sigma}_s^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{\mu}_s)^2$$

Parameters estimation of GMM

$$P(x) = \sum_{m=1}^{M} P(m)\, P(x|m) = \sum_{m=1}^{M} P(m)\, N(x; \mu_m, \sigma_m^2 I)$$

➢ Two conditions (MLE)

- When we know which component generated the data

$$N_m = \sum_{t=1}^{T} z_{mt}$$

z_{mt} = 1 if component m generated data point x_t (and 0 otherwise)

And estimate the mean, variance and mixing parameters as:

$$\hat{\boldsymbol{\mu}}_m = \frac{\sum_t z_{mt} \mathbf{x}_t}{N_m}$$

$$\hat{\sigma}_m^2 = \frac{\sum_t z_{mt} \|\mathbf{x}_t - \hat{\boldsymbol{\mu}}_m\|^2}{N_m}$$

$$\hat{P}(m) = \frac{1}{T} \sum_t z_{mt} = \frac{N_m}{T}$$

$$L = \ln P(x_1, x_2, \cdots x_T \mid \mu, \sigma^2)$$

$$= \ln \left[\prod_{t=1}^{T} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_t-\mu)^2}{2\sigma^2}\right) \right]$$

$$= \sum_{t=1}^{T} \left(-\frac{1}{2}\ln 2\pi - \frac{1}{2}\ln\sigma^2 - \frac{(x_t-\mu)^2}{2\sigma^2} \right)$$

$$= -\frac{1}{2\sigma^2} \sum_{t=1}^{T} (x_t-\mu)^2 - \frac{T}{2}\ln 2\pi - \frac{T}{2}\ln\sigma^2$$

$$\frac{\partial L}{\partial \mu} = -\frac{1}{2\sigma^2} \sum_{t=1}^{T} (x_t - \hat{\mu}) = 0, \quad \hat{\mu} = \frac{1}{T}\sum_{t=1}^{T} x_t$$

$$\hat{\sigma}^2 = \frac{1}{T} \sum_{t=1}^{T} (x_t - \hat{\mu})^2$$

Parameters estimation of GMM

$$P(x) = \sum_{m=1}^{M} P(m)\, P(x|m) = \sum_{m=1}^{M} P(m)\, N(x; \mu_m, \sigma_m^2 I)$$

- ➢ Two conditions (EM)
- When we don't know which component generated the data

Idea: use the posterior probability $P(m|\mathbf{x})$, which gives the probability that component m was responsible for generating data point $\mathbf{x}$.

$$P(m|\mathbf{x}) = \frac{p(\mathbf{x}|m)\,P(m)}{p(\mathbf{x})} = \frac{p(\mathbf{x}|m)\,P(m)}{\sum_{m'=1}^{M} p(\mathbf{x}|m')P(m')}$$

The $P(m|\mathbf{x})$s are called the *component occupation probabilities* (or sometimes called the *responsibilities*)

Since they are posterior probabilities:

$$\sum_{m=1}^{M} P(m|\mathbf{x}) = 1$$

MLE for Params estimation

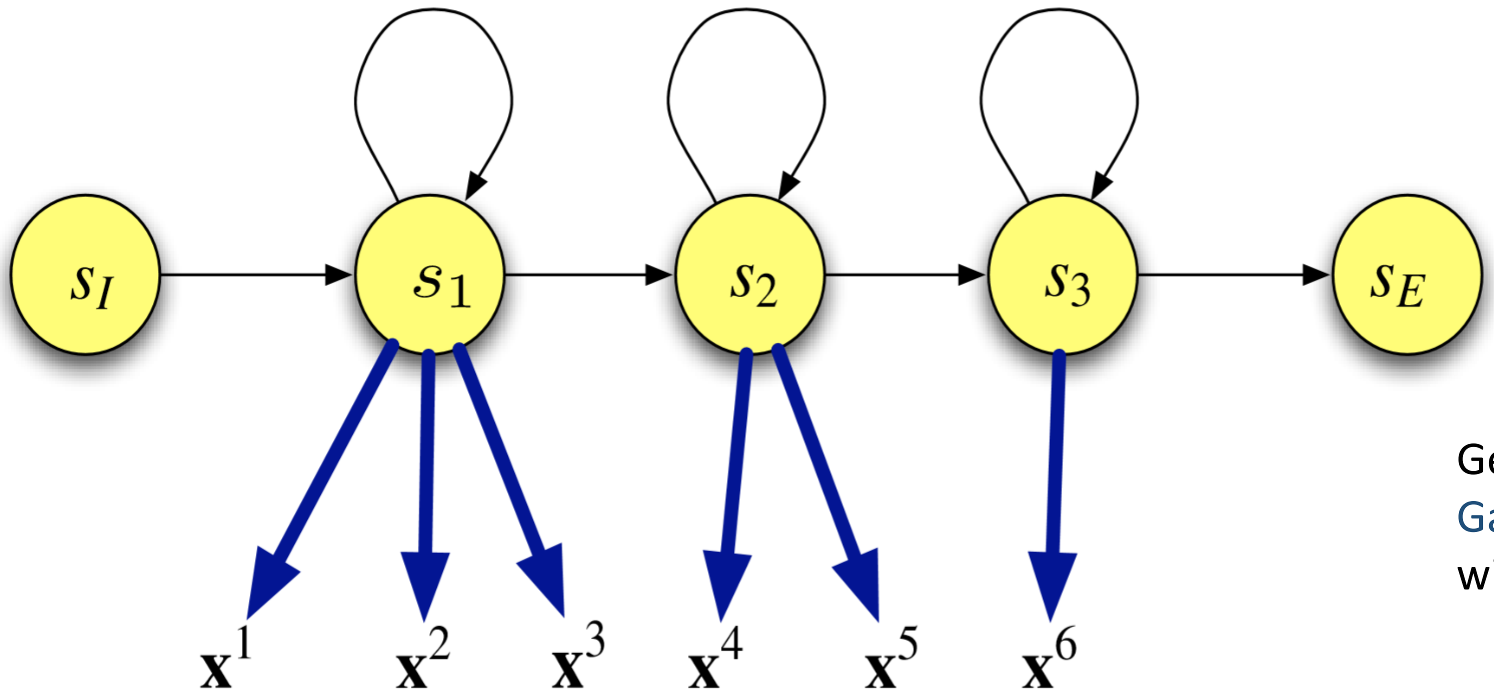
For a fixed training set X, the negative log of likelihood prob can be used as loss to optimize

$$\mathcal{L} = \prod_{t=1}^{T} p(\boldsymbol{x}_t) = \prod_{t=1}^{T} \sum_{m=1}^{M} p(\boldsymbol{x}_t \mid m)\, P(m)$$

Another method EM will be discussed later

Introduction of HMM

Acoustic Model: Continuous Density HMM



Generally bj(x) is fit by Gaussian mixture models with M components

Probabilistic finite state automaton

$$b_j(\mathbf{x}) = p(\mathbf{x} \mid S = j) = \sum_{m=1}^{M} c_{jm} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{jm}, \boldsymbol{\Sigma}_{jm})$$

Parameters $\boldsymbol{\lambda}$:

- Transition probabilities: $a_{kj} = P(S = j \mid S = k)$
- Output probability density function: $b_j(\mathbf{x}) = p(\mathbf{x} \mid S = j)$

HMM Assumptions

- Markov process: The present state's prob only depends on the previous state
- Short time stationary process: each acoustic segment can be viewed as a Short time stationary process which allowed to fit with GMM
- Observation independence: The output observation depends only on the state that produced the observation

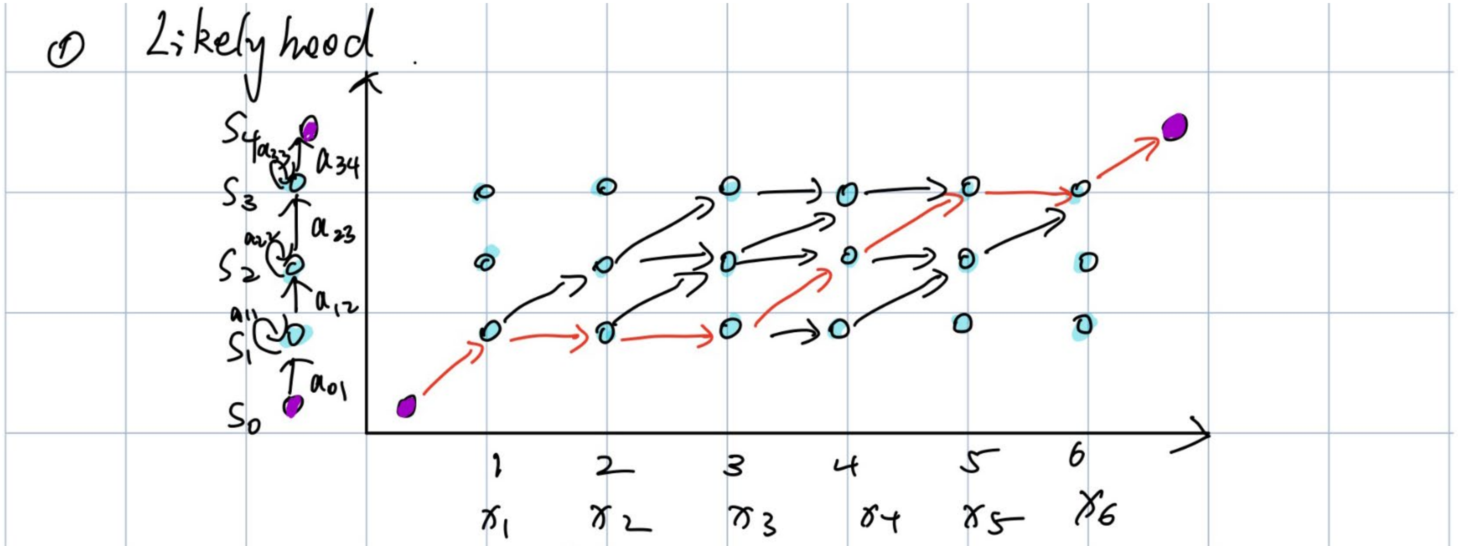
HMM algorithm

HMM algorithm

Three problems

- Likelihood: The likelihood of X with a fixed HMM
- Decoding: Given an observation sequence and an HMM, determine the most probable hidden state sequence
- Training: Given observation sequence and an HMM, learn the best params $\lambda=\{\{\alpha_{jk}\}, \{b_j\}\}$

Likelihood



红线

$$P(X, path_c \mid \lambda) = P(X \mid path_c, \lambda)\, P(path_c \mid \lambda)$$

$$= P(X \mid s_0 s_1 \cdots s_4, \lambda)\, P(s_0 s_1 \cdots s_4 \mid \lambda)$$

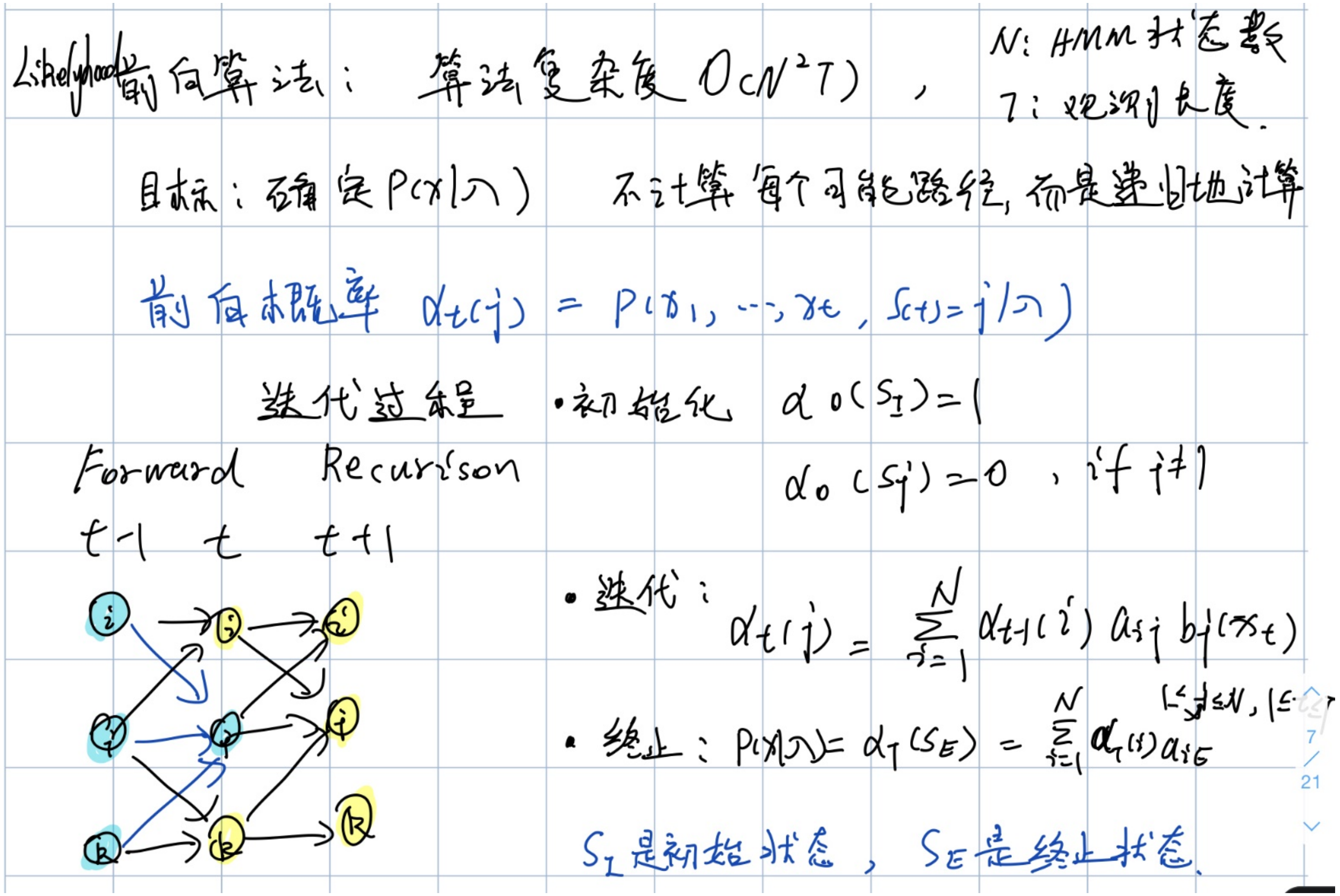
$$= b_1(x_1)\, b_1(x_2)\, b_1(x_3)\, b_2(x_4)\, b_3(x_5)\, b_3(x_6)\, a_{01} a_{11} a_{11} a_{12} a_{23} a$$

$$P(X \mid \lambda) = \sum_{path_c} P(X, path_c \mid \lambda) \simeq \max_{path_c} P(X, path_c \mid \lambda)$$

Forward/backward algorithm Viterbi algorithm

Likelihood: Forward algorithm

Compute the probability recursively



Viterbi algorithm is different in recursive step, which need to save the previous state to backtrace.

Decoding: Viterbi algorithm

② Decoding 解码问题

用维特比 Viterbi 算法，过程如下

- 初始化. $V_0(i)=1$

  $V_0(j)=0$ ，若 $j\neq i$

  $bt_0(j)=0$

- 迭代 $V_t(j)=\max_{i=1}^{N} V_{t-1}(i)\,a_{ij}\,b_j(x_t)$

  $bt_t(j)=\arg\max_{i=1}^{N} V_{t-1}(i)\,a_{ij}\,b_j(x_t)$

- 终止： $p^*=V_T(s_E)=\max_{i=1}^{N} V_T(i)\,a_{iE}$

  $s_T^*=bt_T(q_E)=\arg\max_{i=1}^{N} V_T(i)\,a_{iE}$

Training: Baum-Welch algorithm

Goal: Efficiently estimate the parameters of an HMM λ from an observation sequence

Approaches: Viterbi algorithm as approximation and BW algorithm(EM) for all paths

- 假设发射概率为单高斯 $b_j(x) = P(x|j) = N(x; \mu_j, \Sigma_j)$
- 参数λ： 转移概率 a_{ij}： $\sum_j a_{ij} = 1$
  Gaussian 参数 for 状态 j： μ_j, Σ_j

Viterbi 训练：想通过 Viterbi 算法找到最可能路径来获得状态和时间的对齐，即每个观测都可知道对应哪个状态

a_{ij} 的估计可以用 $\hat{a}_{ij} = \frac{C(i \to j)}{\sum_k C(i \to k)}$

若 j 对应观测序列组合是 Z_j 则

$$\hat{\mu}_j = \frac{\sum_{x \in Z_j} x}{|Z_j|} \quad \hat{\Sigma}_j = \frac{\sum_{x \in Z_j} (x - \hat{\mu}_j)(x - \hat{\mu}_j)^T}{|Z_j|}$$

EM 算法： 用 Viterbi 训练近似所有路径.

用 $\gamma_t(j)$ 表示 t 时刻观测属于状态 j 的概率

接下来用 EM 算法迭代计算.

每轮迭代有两步：

E-step 估计 $\gamma_t(j)$ 概率（期望）

M-step 根据 $\gamma_t(j)$ 来重新估计 HMM 参数

（Maximisation）

Soft-alignment

Training: Baum-Welch algorithm

E-step: with X and λ

$$\alpha_t(j)\beta_t(j) = p(\mathbf{x}_1, \ldots, \mathbf{x}_t, S(t)=j \mid \boldsymbol{\lambda})\, p(\mathbf{x}_{t+1}, \ldots, \mathbf{x}_T \mid S(t)=j, \boldsymbol{\lambda})$$

$$= p(\mathbf{x}_1, \ldots, \mathbf{x}_t, \mathbf{x}_{t+1}, \ldots, \mathbf{x}_T, S(t)=j \mid \boldsymbol{\lambda})$$

$$= p(\mathbf{X}, S(t)=j \mid \boldsymbol{\lambda})$$

$$P(S(t)=j \mid \mathbf{X}, \boldsymbol{\lambda}) = \frac{p(\mathbf{X}, S(t)=j \mid \boldsymbol{\lambda})}{p(\mathbf{X} \mid \boldsymbol{\lambda})}$$

后向算法

为了计算 $\gamma_t(j)$，定义后向概率 $\beta_t(j)$

$\beta_t(j) = p(x_{t+1}, \ldots, x_T \mid S(t)=j, \lambda)$

给定HMM中时刻t是状态j的概率

迭代计算
- 初始：$\beta_T(i) = a_{iE}$
- 迭代：$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(x_{t+1}) \beta_{t+1}(j)$
- 终止：$P(x \mid \lambda) = \beta_0(I) = \sum_{j=1}^{N} a_{Ij} b_j(x_1) \beta_1(j) = \alpha_T(s_E)$

State Occupation Probability $\gamma_t(j)$

是给定观测序列在 t 时刻属于状态 j 的概率

整个序列

$\gamma_t(j) = P(S(t)=j \mid X, \lambda)$

$= \frac{1}{\alpha_T(s_E)} \alpha_t(j) \beta_t(j)$; $P(X \mid \lambda) = \alpha_T(s_E)$

Training: Baum-Welch algorithm

M-step: Re-estimation of HMM

- Similarly to the state occupation probability, we can estimate $\xi_t(i, j)$, the probability of being in i at time t and j at $t+1$, given the observations:

$$\begin{aligned}\xi_t(i, j) &= P(S(t)=i, S(t+1)=j \mid \mathbf{X}, \boldsymbol{\lambda}) \\ &= \frac{p(S(t)=i, S(t+1)=j, \mathbf{X} \mid \boldsymbol{\lambda})}{p(\mathbf{X} \mid \boldsymbol{\lambda})} \\ &= \frac{\alpha_t(i) a_{ij} b_j(\mathbf{x}_{t+1}) \beta_{t+1}(j)}{\alpha_T(s_E)}\end{aligned}$$

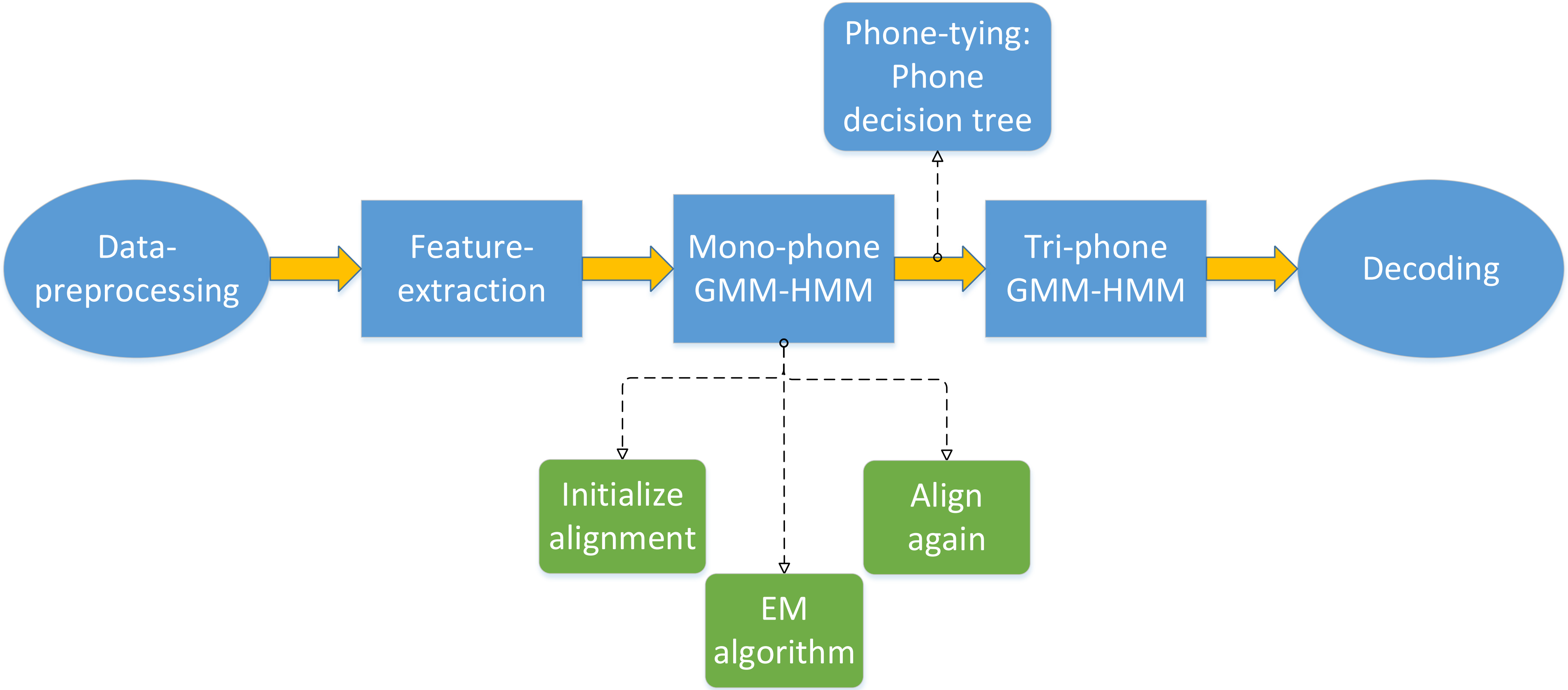
- We can use this to re-estimate the transition probabilities

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

Workflow of GMM-HMM in ASR

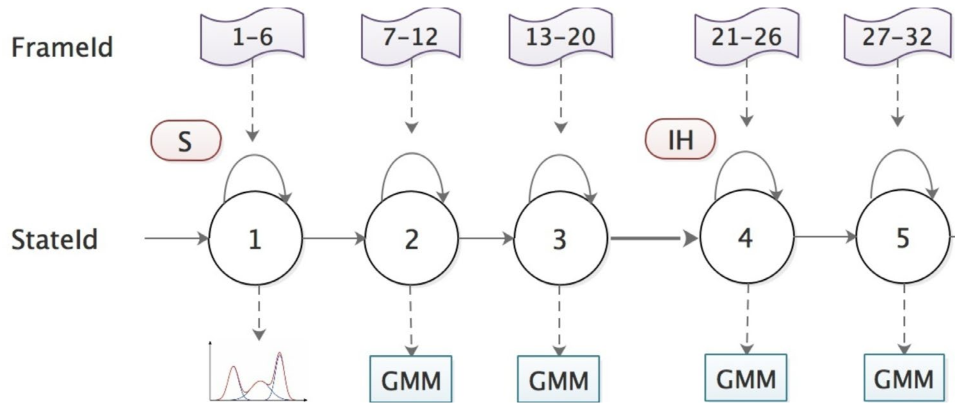
中国科学院
自动化研究所
INSTITUTE OF AUTOMATION
CHINESE ACADEMY OF SCIENCES

Workflow of GMM-HMM in ASR



Multi-step workflow of ASR

Mono-phone GMM-HMM



- ➢Initialize alignment: k-means for GMM
- ➢EM algorithm: compute α_{ij} and b_j
- ➢Align again: Baum-Welch and Viterbi

Soft-alignment *Hard-alignment*

✓ Repeat 2 and 3 until convergence

Context-dependent phone models

中国科学院自动化研究所
INSTITUTE OF AUTOMATION
CHINESE ACADEMY OF SCIENCES

Context-dependent phone models

The need to model phonetic context

- **Context**: The acoustic phonetic context of a speech unit has an effect on its acoustic realization

- **Coarticulation:** The place of articulation for one speech sound depends on a neighboring speech sound

Consider /n/ in ten and tenth

- alveolar in ten
- dental in tenth

Context-dependent phone models

- **Triphone:** Represent a phone x with left context l and right context r as l-x+r
- **Word-internal triphone:** Only take account of context within words
- **Cross-word triphone:** Don’t ask: “sil sil-d+oh d-oh+n oh-n+t n-t+a …”

Problem: Unseen data, Too more params, Data sparse

Solution: Smoothing, Parameter sharing

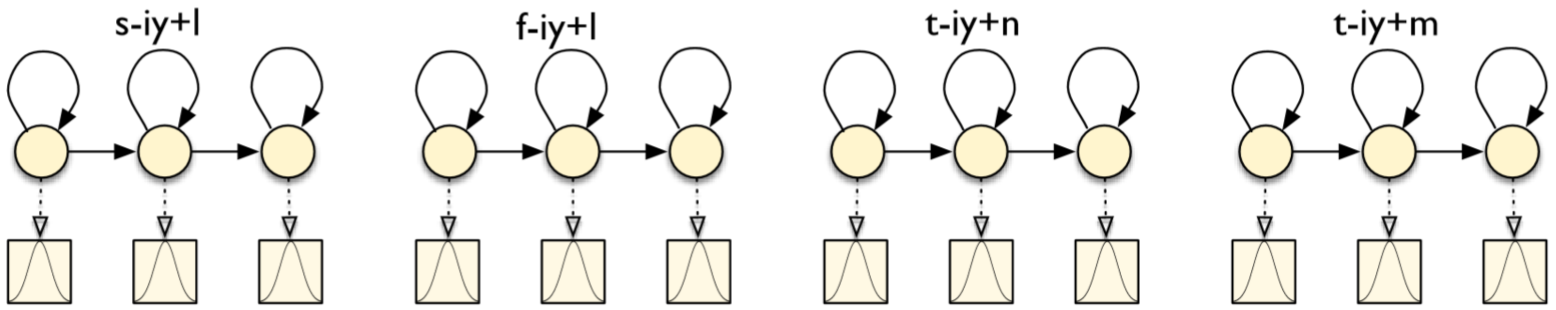
CD-phone models: Parameter sharing

Core idea: Explicitly share models or parameters between different contexts

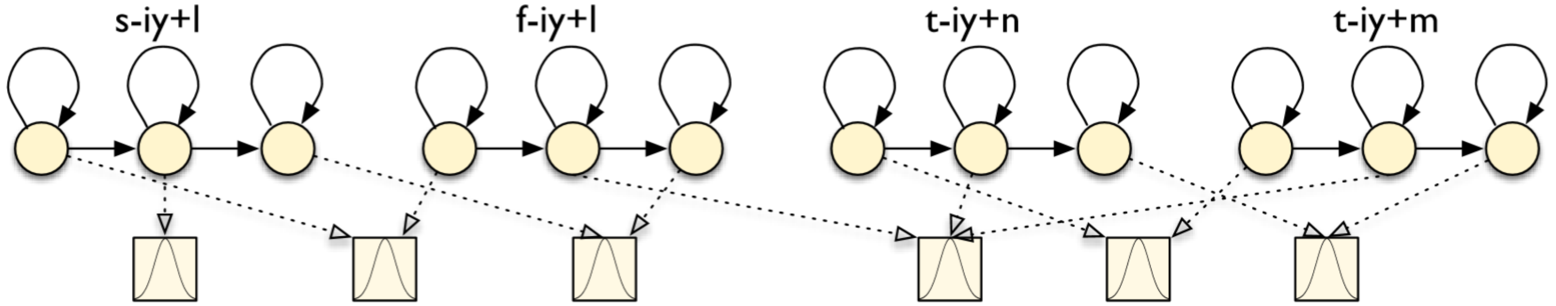
- Enable training data to be shared between the models
- Enable models to share parameters

1. Sharing Gaussians: all distributions share the same set of Gaussians but have different mixture weights (tied mixtures)
2. Sharing states: allow different models to share the same states (state clustering)
3. Sharing models: merge those context-dependent models that are the most similar (generalised triphones)

CD-phone models: State clustering



Simple triphones (no sharing)



State-clustered triphones (state sharing)

ata are shared

. (tree-based

stered state is

Phonetic Decision Trees: Top-down clustering

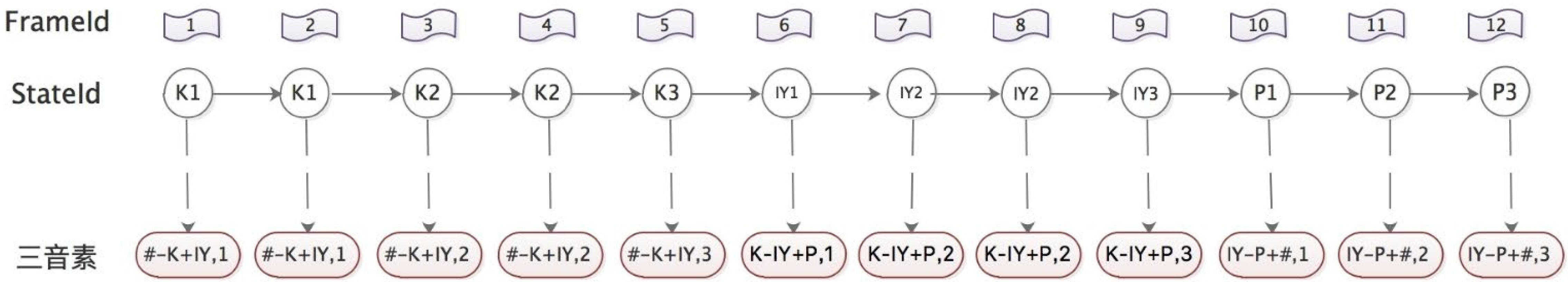
Input: a) mono-phone system and definite observation sequence for each state of mono-phone by Viterbi algorithm; b) a question set

Goal: extend mono-phone to triphone for state-tying , build a decision tree for each state of each triphone

Output: Triphone GMM-HMM system with decision trees

Phonetic Decision Trees: Building Steps

Step 1: Mono-phone alignment to Triphone alignment (3 states)



Phonetic Decision Trees: Building Steps

Step 2: Build binary decision tree based on the question set

- 分成两类：

$$L(S_l) + L(S_r) = -\frac{1}{2}mN(1+\log(2\pi)) - \frac{1}{2}[m_l \sum_{k=1}^{N} \log(\sigma_{lk}^2) + m_r \sum_{k=1}^{N} \log(\sigma_{rk}^2)]$$

- 似然增益(Likelihood gain)

$$D = L(S_{l1}) + L(S_{r1}) - L(S)$$

- 最优问题 q^*

$$q^* = \operatorname*{argmin}_q \left[m_l \sum_{k=1}^{N} \log(\sigma_{lk}^2) + m_r \sum_{k=1}^{N} \log(\sigma_{rk}^2) \right]$$

$$\sigma_{lk}^2 = \frac{1}{m_l} \sum_{x \in S_l} x_k^2 - \frac{1}{m_l^2} (\sum_{x \in S_l} x_k)^2$$

$$\sigma_{rk}^2 = \frac{1}{m_r} \sum_{x \in S_r} x_k^2 - \frac{1}{m_r^2} (\sum_{x \in S_r} x_k)^2.$$

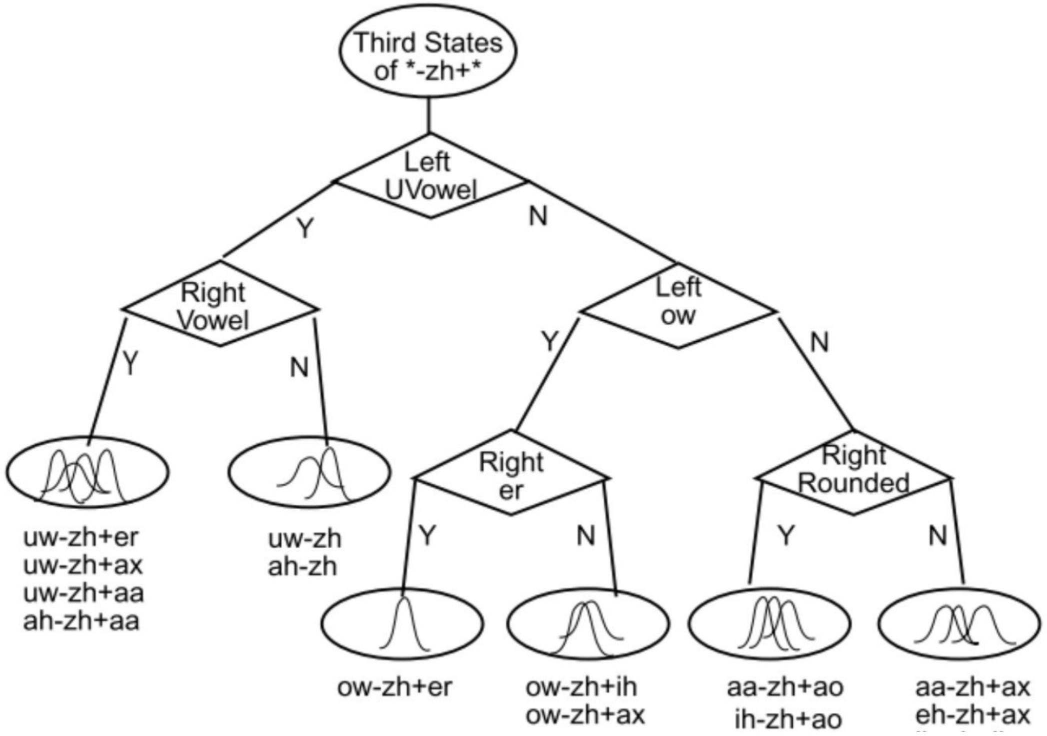
- The root node is one state of triphone
- Select the optimal question that divide corresponding observation samples
- Endue different probs of other unseen triphone

Phonetic Decision Trees: Building Steps

Step 3: Repeat step 2 until convergence

Stop criterion:

1. The number of leaves nodes exceed a fixed number;
2. The likelihood gain is below than the threshold



Inference process

- Given observation sequence X with the trained triphone GMM-HMM acoustic models, we can get marginal prob of each triphone in phonetic decision tree’s leaves.

- Then we can decoding with LM to achieve the conventional structure

Inference process

WFST: HCLG for decoding

	transducer	input sequence	output sequence
G	word-level grammar	words	words
L	pronunciation lexicon	phones	words
C	context-dependency	CD phones	phones
H	HMM	HMM states	CD phones

LM score *AM score*

Training order: G->L->C->H

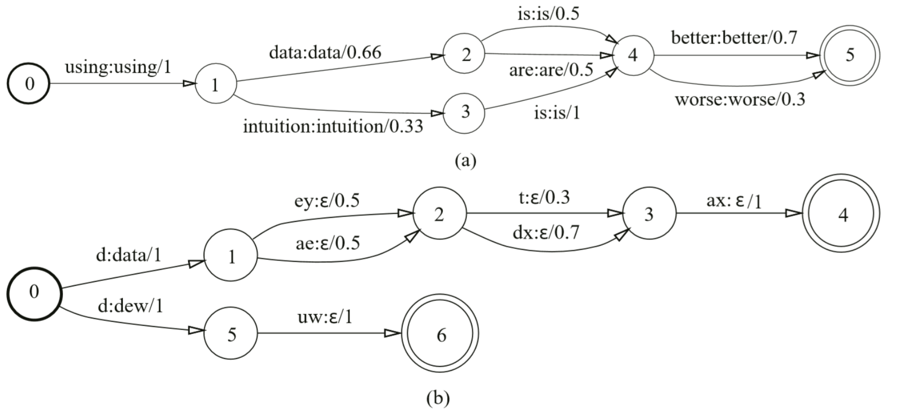
HCLG = asl(min(rds(det(H' o min(det(C o min(det(Lo G))))))))

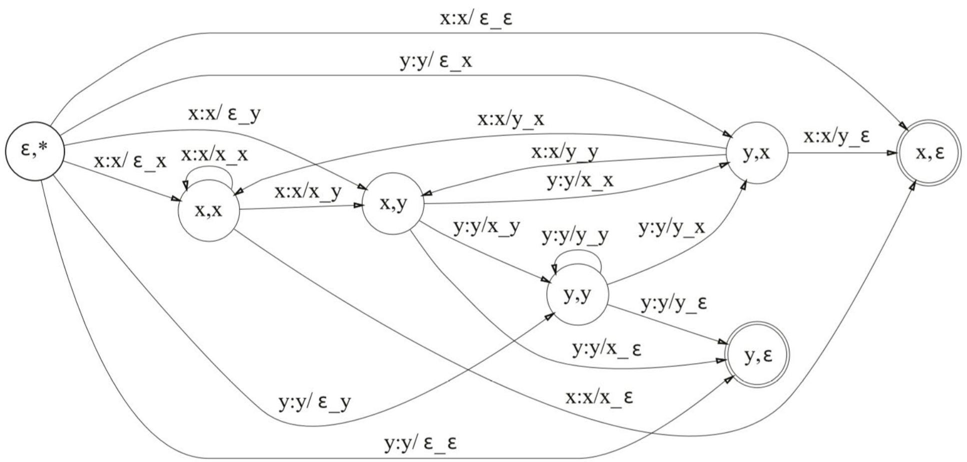
Inference process

WFST: HCLG for decoding

G is generated from statistic, L is the lexicon in generating G, C generated from context-dependent phone decision tree based on L

- ➢ Lattice is used to save N-best in decoding
- ➢ Viterbi or beam search to get results





Thanks

中国科学院
自动化研究所
INSTITUTE OF AUTOMATION
CHINESE ACADEMY OF SCIENCES