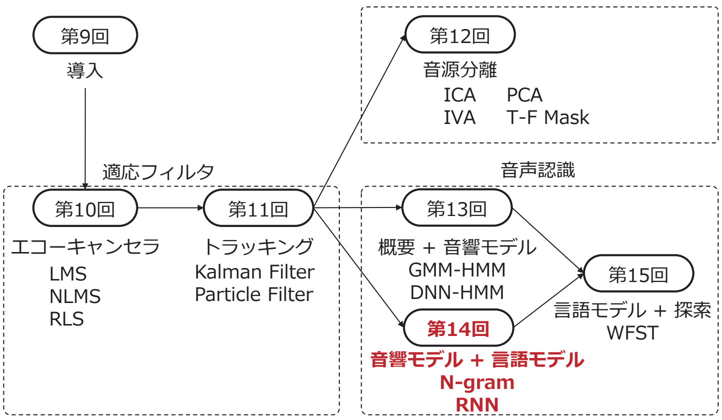
知的情報処理論 第14回

2023年7月18日（火）
武田

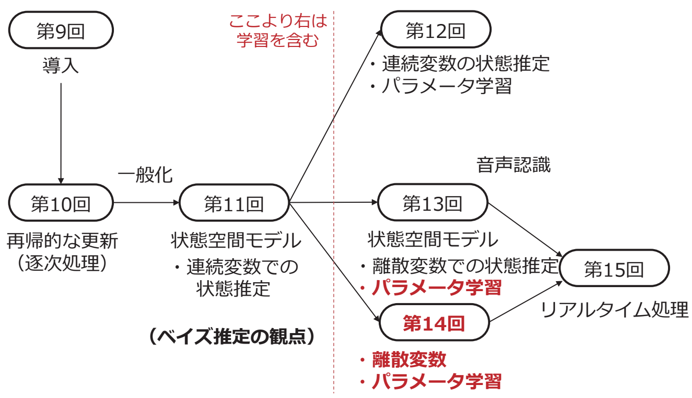
レポート課題

- 8/4 (金) 23:59 提出締め切り
- 詳細は CLE上の pdf を見ること

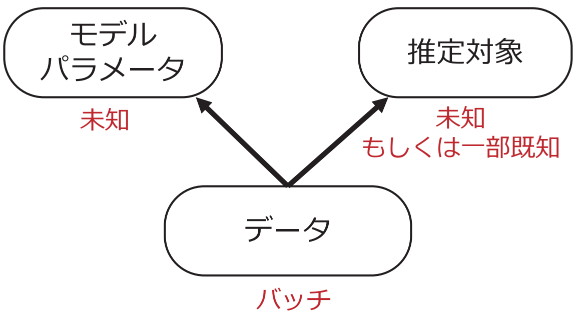
各回の内容（予定）



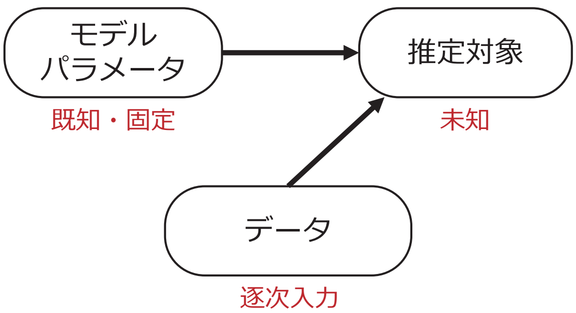
各回の内容（予定）



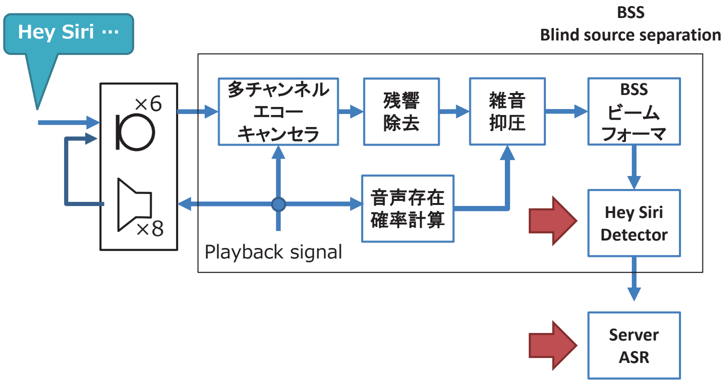
今回の話題：学習



今回の話題：認識



Apple Home Pod の構成 (SLT2018より)



音声認識

音声信号をテキストに変換

音声認識

音響モデル: (Infinite) GMM

言語モデル: 教師なし単語分割

頭の片隅に置いとく方が良い大まかな要素

コスト関数

これが何かわかってないとデバッグできない

- 「良さ」を測る関数
  - 教師あり学習: 二乗誤差, CrossEntropy, etc…
  - 教師なし学習: 尤度, 再構成誤差, etc…
  - 認識・推定: 期待値, 最大事後確率(MAP), etc…
- ロス/目的関数ともいう

モデル

- 入出力の関係や拘束条件を記述
- 方程式や確率モデルで表現

（コスト関数と一体化 or 切り離せない場合も）

ざっくりとした認識/学習/探索

- コスト関数を{最大化|最小化|良く}する値（集合）を求める手続き（**最適化等の分野**）
- モデル構造や補助変数・関数を利用した手続きも存在

「初期値依存性」 「局所的最適解（局所解）」 「大域的最適解」

本日の内容: 音声認識

0. 補足

第12回: 主成分分析(PCA)
独立成分分析（ICA）との関係

1. 音響モデル II (GMM, DNN)
   1. GMM モデルと学習
   2. DNN-HMM

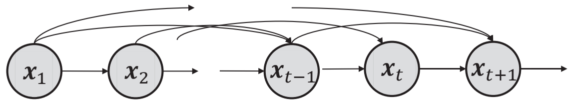
潜在変数が離散確率変数の場合（GMMなど）

2. 言語モデル I
   1. N-gram モデル
   2. 確率の平滑化

第13回: HMM

“状態”が隠れてない場合（離散的な自己回帰）

自己回帰モデル



$$p(\boldsymbol{x}_t | \boldsymbol{x}_{t-1}, \boldsymbol{x}_{t-2}, \dots, \boldsymbol{x}_{t-n})$$

音声処理での例: 短時間区間での 線形予測分析 (LPC)

$$x_t = a_1 x_{t-1} + \cdots + a_n x_{t-n} + n_t \qquad n_t \sim \mathcal{N}(0, \sigma^2)$$

$$p(x_t | x_{t-1}, \dots, x_{t-n}) = \mathcal{N}(\textstyle\sum_{i=1}^{n} a_i x_{t-i}, \sigma^2)$$

自然言語処理での例: N-gram 言語モデル

$$p(w_t | w_{t-1}, w_{t-2}, \dots, w_{t-n})$$

[当然こんなのも考えられる $x_t = f(x_{t-1}, \dots, x_{t-n}) + n_t \quad n_t \sim \mathcal{N}(0, \sigma^2)$]

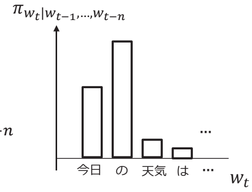
具体的な中身

言語モデルの場合

– カテゴリカル分布を仮定

$$p(w_t | w_{t-1}, \dots, w_{t-n}) = \pi_{w_t | w_{t-1}, \dots, w_{t-n}}$$

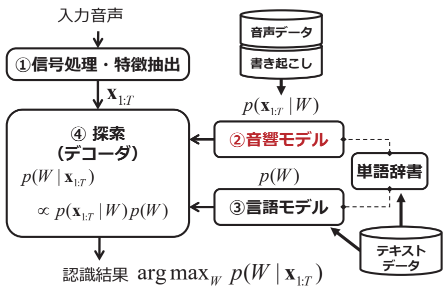
目の多い"いびつ"なサイコロみたいなもの $\sum_{w_t} \pi_{w_t | w_{t-1}, \dots, w_{t-n}} = 1$



– 識別モデル（NN実装など）を仮定

$$p(w_t | w_{t-1}, \dots, w_{t-n}) = \mathrm{softmax}(\boldsymbol{f}(w_{t-1}, \dots, w_{t-n}))$$

グラフィカルモデルとその中身: 別の設計



②音響モデル II (GMM, DNN)

あらすじ

HMM の学習: EM アルゴリズム

– 潜在変数の事後分布計算

– Q関数に基づくパラメータ更新

パラメータ

前回: HMM 状態遷移確率 $p(s_t | s_{t-1})$

今回: 尤度関数 $p(\mathbf{x}_t | s_t)$

- GMM
- その拡張

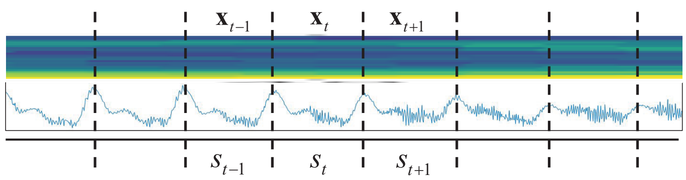
基礎的なモデル: HMM

隠れマルコフモデル(hidden Markov Model; HMM)

– 系列データの生成モデル（状態空間モデル）

– 状態（離散確率変数）s_t とその遷移で表現

状態は直接観測できない＝隠れ/潜在変数

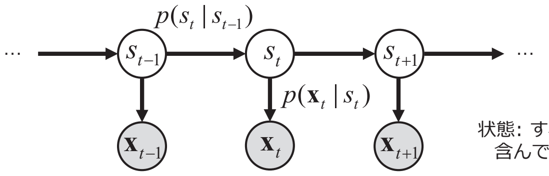


各 $\mathbf{x}_t$ に対し状態（クラス）s_t を割り当て

基礎的なモデル: HMM

隠れマルコフモデル(hidden Markov Model; HMM)
- 系列データの生成モデル（状態空間モデル）
- 状態（離散確率変数）s_t とその遷移で表現

状態は直接観測できない＝隠れ/潜在変数



状態: すべての情報を含んでいると仮定

状態変数列 $s_{1:T} = [s_1, \ldots s_T]$ 遷移確率 $p(s_t \mid s_{t-1})$ 尤度 $p(\mathbf{x}_t \mid s_t)$

基礎的なモデル: HMM

隠れマルコフモデル(hidden Markov Model; HMM)
- 系列データの生成モデル（状態空間モデル）
- 状態（離散確率変数）s_t とその遷移で表現

状態は直接観測できない＝隠れ/潜在変数

系列尤度

$$p(\mathbf{x}_{1:T}) = \sum_{s_{1:T}} p(\mathbf{x}_{1:T} \mid s_{1:T}) p(s_{1:T})$$

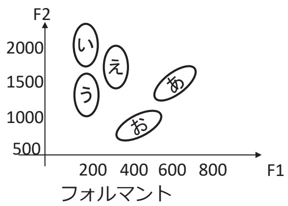
$$= \sum_{s_{1:T}} \prod_t p(\mathbf{x}_t \mid s_t) p(s_t \mid s_{t-1}) p(s_1)$$

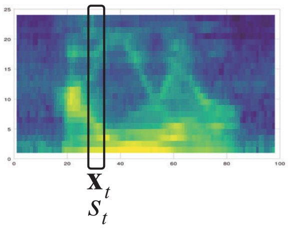
今回の対象

状態変数列 $s_{1:T} = [s_1, \ldots s_T]$ 遷移確率 $p(s_t \mid s_{t-1})$ 尤度 $p(\mathbf{x}_t \mid s_t)$

尤度関数のパラメータ学習

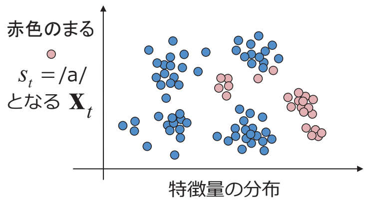
音素クラスにおける特徴量分布 $p(\mathbf{x}_t \mid s_t)$
- 音声特徴量 $\mathbf{x}_t$
- 音素クラス s_t （3状態音素HMMの隠れ状態）
  - mono-phone: /a/, /i/, /u/ など
  - tri-phone: 文脈依存の音素ラベル, /a-i+u/ など

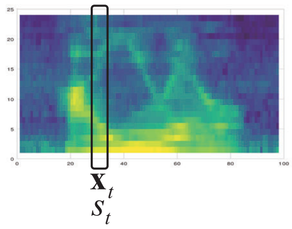




尤度関数のパラメータ学習

音素クラスにおける特徴量分布 $p(\mathbf{x}_t \mid s_t)$
- 音声特徴量 $\mathbf{x}_t$
- 音素クラス s_t （3状態音素HMMの隠れ状態）
  - mono-phone: /a/, /i/, /u/ など
  - tri-phone: 文脈依存の音素ラベル, /a-i+u/ など





基本的なモデル GMM に基づく尤度関数

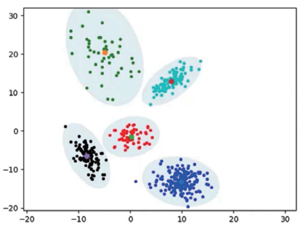
Gaussian mixture model
- 生成過程: ガウス分布の重み付き和 - 混合分布
  1. クラス z が確率的に決まる $z \sim p(z)$
  2. データ $\mathbf{x}$ がガウス分布 z から生成される $\mathbf{x} \sim p(\mathbf{x} \mid z)$
- 教師なしクラスタリングを行うことも可能

$$p(\mathbf{x}) = \sum_z p(\mathbf{x} \mid z) p(z)$$

$$= \sum_i \pi_i N(\mathbf{x} \mid \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$$

π_i クラス i の混合重み $p(z=i) = \pi_i$

$\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i$ クラス i の平均・共分散行列



GMM に基づく尤度関数

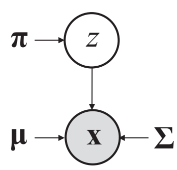
Gaussian mixture model
- 生成過程: ガウス分布の重み付き和 - 混合分布
  1. クラス z が確率的に決まる $z \sim p(z)$
  2. データ $\mathbf{x}$ がガウス分布 z から生成される $\mathbf{x} \sim p(\mathbf{x} \mid z)$
- 教師なしクラスタリングを行うことも可能

$$p(\mathbf{x}) = \sum_z p(\mathbf{x} \mid z) p(z)$$

$$= \sum_i \pi_i N(\mathbf{x} \mid \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$$

π_i クラス i の混合重み $p(z=i) = \pi_i$

$\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i$ クラス i の平均・共分散行列



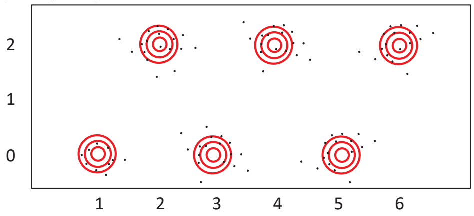
例: ダーツの的

プロセス: 複数の的を狙ってダーツを投げる

1. サイコロを振る $z \sim p(z)$
2. 出た目の的を狙って投げる $\mathbf{x} \sim p(\mathbf{x} \mid \boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z)$

Q: 第3者が形跡を見て，的の位置を推測する

- 「ダーツをしていた」「的は 6つ」ということは知っている



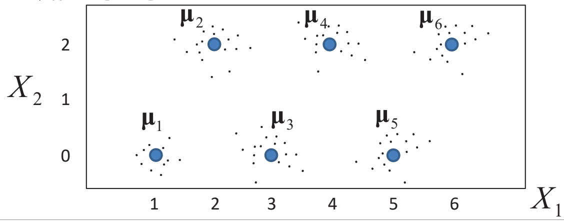
例: ダーツの的

プロセス: 複数の的を狙ってダーツを投げる

1. サイコロを振る $z \sim p(z)$
2. 出た目の的を狙って投げる $\mathbf{x} \sim p(\mathbf{x} \mid \boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z)$

Q: 第3者が形跡を見て，的の位置を推測する

- 「ダーツをしていた」「的は6つ」ということは知っている



GMM パラメータ学習

EM アルゴリズム

1. パラメータ初期値設定 $\boldsymbol{\theta}^{old}$
2. E-step: $p(\mathbf{z} \mid \mathbf{x}, \boldsymbol{\theta}^{old})$ を計算

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) = \sum_{\mathbf{z}} p(\mathbf{z} \mid \mathbf{x}, \boldsymbol{\theta}^{old}) \ln p(\mathbf{x}, \mathbf{z} \mid \boldsymbol{\theta})$$

3. M-step: $\boldsymbol{\theta}^{new}$ を計算

$$\boldsymbol{\theta}^{new} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{old})$$

4. 収束条件が満たされていなければ step 2へ

GMM パラメータ学習

EM アルゴリズム

データ: $\mathbf{x}_{1:N} = [\mathbf{x}_1, \mathbf{x}_2, .., \mathbf{x}_N]$ ダーツの座標

潜在変数: $z_{1:N} = [z_1, z_2, .., z_N]$ 「どの的を狙ったか」= 的のID（{1,2,3,4,5,6} のどれかを取る）

事後確率:

$$p(z_n \mid \mathbf{x}_n) = \frac{p(\mathbf{x}_n \mid z_n) p(z_n)}{\sum_{z_n} p(\mathbf{x} \mid z_n) p(z_n)}$$

$$\gamma(z_{n,i}) = p(z_n = i \mid \mathbf{x}_n, \boldsymbol{\Theta}_{old})$$

Q関数:

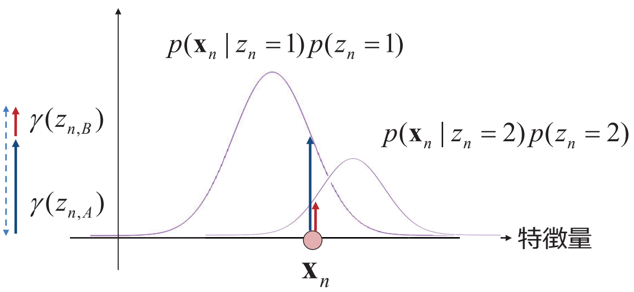
$$Q = E[\ln p(\mathbf{x}_{1:N}, z_{1:N})]$$

$$= \sum_{z_{1:N}} p(z_{1:N} \mid \mathbf{x}_{1:N}, \boldsymbol{\Theta}_{old}) [\underbrace{\boxed{\ln p(\mathbf{x}_{1:N} \mid z_{1:N})}}_{\text{尤度}} + \underbrace{\boxed{\ln p(z_{1:N})}}_{\text{事前確率}}]$$

事後確率（負担率）$\gamma(z_{n,i})$

データ点 $\mathbf{x}_n$ のそれぞれの分布（クラス）i との"割合"

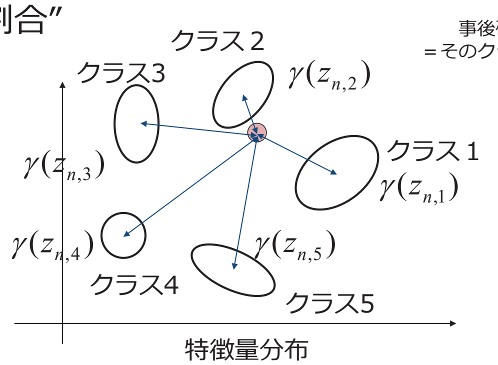
事後確率が大
＝そのクラスに近い



事後確率（負担率）$\gamma(z_{n,i})$

データ点 $\mathbf{x}_n$ のそれぞれの分布（クラス）i との"割合"

事後確率が大
＝そのクラスに近い



GMM パラメータ学習

混合重みパラメータのコスト関数

$$J = \lambda\left(\sum_k \pi_k - 1\right) + \sum_{z_{1:N}} p(z_{1:N} \mid \mathbf{x}_{1:N}, \mathbf{\Theta}_{old}) \ln p(z_{1:N})$$

$$= \underbrace{\lambda\left(\sum_k \pi_k - 1\right)}_{\text{拘束条件}} + \underbrace{\sum_n \sum_k \gamma(z_{n,k}) \ln \pi_k}_{\text{データ間の独立性より (I.I.D)}}$$

$\dfrac{\partial J}{\partial \pi_i} = 0$ より $\pi_i \lambda + \sum_n \gamma(z_{n,i}) = 0$

$$\lambda = -\sum_n \sum_i \gamma(z_{n,i}) = -N$$

$$\pi_i = \frac{1}{N} \sum_n \gamma(z_{n,i})$$ “割合”の平均化

GMM パラメータ学習

ガウス分布パラメータのコスト関数

$$N(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \mathbf{\Sigma}_k) = \frac{1}{(2\pi)^{D/2} |\mathbf{\Sigma}_k|^{1/2}} \exp\left\{ -\frac{1}{2} (\mathbf{x}_n - \boldsymbol{\mu}_k)^T \mathbf{\Sigma}_k^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_k) \right\}$$

$$J = \sum_{z_{1:N}} p(z_{1:N} \mid \mathbf{x}_{1:N}, \mathbf{\Theta}_{old}) \ln p(\mathbf{x}_{1:N} \mid z_{1:N})$$

$$= \sum_n \sum_k \gamma(z_{n,k}) \ln N(\mathbf{x}_n \mid \boldsymbol{\mu}_k, \mathbf{\Sigma}_k)$$ データ間の独立性

$$= \sum_n \sum_k \gamma(z_{n,k}) \left[-\frac{D}{2} \ln 2\pi - \frac{1}{2} \ln |\mathbf{\Sigma}_k| - \frac{1}{2} (\mathbf{x}_n - \boldsymbol{\mu}_k)^T \mathbf{\Sigma}_k^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_k) \right]$$

GMM パラメータ学習

平均値の更新

$$\frac{\partial J}{\partial \boldsymbol{\mu}_i} = \frac{\partial J}{\partial \boldsymbol{\mu}_i} \sum_n \sum_k \gamma(z_{n,k}) \left[-\frac{1}{2} (\mathbf{x}_n - \boldsymbol{\mu}_k)^T \mathbf{\Sigma}_k^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_k) \right]$$

$$= \sum_n \gamma(z_{n,i}) \mathbf{\Sigma}_i^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_i) = 0$$

$$\sum_n \gamma(z_{n,i}) \boldsymbol{\mu}_i = \sum_n \gamma(z_{n,i}) \mathbf{x}_n$$

$$\boldsymbol{\mu}_i = \frac{\sum_n \gamma(z_{n,i}) \mathbf{x}_n}{\sum_n \gamma(z_{n,i})}$$ “割合”を用いた重み付き平均

共分散行列も同様に closed-form で出せる

Mini Quiz #1

- GMM のダーツの例において，ダーツを投げた本人が，後でダーツの形跡だけを見て，的の座標とバラつきを推定する場合，どのようにパラメータを推定すればよいか？
  - 投げた本人は「各ダーツでどれを狙って投げたかは覚えている」とする
    - 第3者目線では，「どの的を狙ったか」は直接観測できない＝潜在変数.
    - 投げた本人は覚えている = 潜在変数ではない
  - これは何学習になるだろうか

他: 事前分布の導入 $p(\boldsymbol{\pi})$

最尤推定 → ベイズ的な方法 $p(z) \Rightarrow \dfrac{p(z \mid \boldsymbol{\pi})}{p(\boldsymbol{\pi})}$

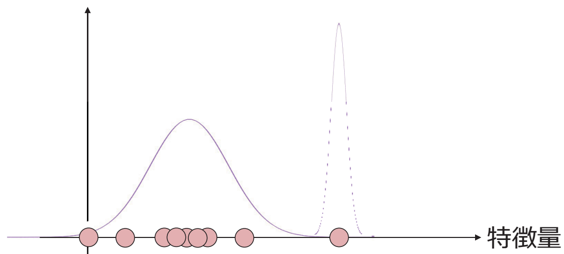
パラメータ → 確率変数

共役事前分布の利用
- 混合重み: ディリクレ分布
- ガウス 平均・分散: ガウス-ウィシャート分布

推定: 変分ベイズ法・モンテカルロ等の利用

利点: スムージング的役割
- 最尤推定で現れる「特異な解」を回避
- 混合数に対する過学習が起きない – （不要なクラスタの）混合係数の期待値は 0 へ

他: 事前分布の導入



右側のガウス分布: サンプル点が1つ

最尤推定: 分散 異常値になりがち ☹

ベイズ推定: 分散の値をスムージング
- 事前分布のパラメータとデータからの推定値

37

他: Infinite GMM

GMM にディリクレ過程を導入

- ディリクレ分布の無限次元拡張
  - 潜在的に無限のクラスが存在
  - サイコロの「面数」そのものも確率的に変動
- 分布に対する分布

事前分布の構成法

- Stick-breaking process (SBR)
- Chinese restaurant process (CRP)
  - 既存クラス or 新規クラスの判定
  - クラスタとの関係を理解しやすい

38

HMM 上での尤度関数

GMM-HMM の場合

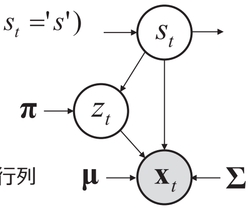
- HMMの状態でさらに条件付けが行われる
- 入れ子構造 → HMM のEM適用時に注意
  - $\{s_t, z_t\}_{t=1}^{T}$ が潜在変数: 同時/事後分布の定義からやる
  - 状態の事後確率は時刻に依存: forward-backward

$$p(\mathbf{x}_t \mid s_t = 's') = \sum_{z_t} p(\mathbf{x}_t \mid z_t, s_t = 's') p(z_t \mid s_t = 's')$$

$$= \sum_i \pi_{i|s} N(\mathbf{x}_t \mid \boldsymbol{\mu}_{i|s}, \boldsymbol{\Sigma}_{i|s})$$

$\pi_{i|s}$ 状態 s クラス i 混合重み

$\boldsymbol{\mu}_{i|s}, \boldsymbol{\Sigma}_{i|s}$ 状態 s クラス i の平均・共分散行列



39

音声データに対する 実際の GMM-HMM 学習

HMM 状態共有化: tied-state モデル

- tri-phone: 状態数は音素数の3乗に比例
  - ☹ 尤度関数の学習にはデータ不足の状態も存在
- 分類規則を用いた状態のクラスタリング
  - パラメータ数削減など**実用上重要**
  - DNN-HMM でも共有化後の状態IDを学習に利用

識別学習: 対立候補も用いて学習

例: 事後確率最大化

$$p(W \mid \mathbf{x}_{1:T}) = \frac{p(W) \sum_{s_{1:T}} p(\mathbf{x}_{1:T} \mid s_{1:T}, W) p(s_{1:T} \mid W)}{\sum_W p(W) \sum_{s_{1:T}} p(\mathbf{x}_{1:T} \mid s_{1:T}, W) p(s_{1:T} \mid W)}$$

40

深層学習: DNN-HMM

HMM とのハイブリッドモデル

- 音響尤度を DNN 事後確率で置き換え

$$p(\mathbf{x}_t \mid s_t) = \frac{p_{DNN}(s_t \mid \mathbf{x}_t) \boxed{p(\mathbf{x}_t)}}{p(s_t)}$$

定数
（データ点は固定）

- DNN 事後確率: 最終層 softmax 関数の出力

$$\mathrm{softmax}(s \mid \mathbf{z}) = \frac{\exp(z_s)}{\sum_i \exp(z_i)}$$

- 特徴量抽出過程を含めて最適化可能

41

深層学習: DNN-HMM

DNN: 基本は教師あり学習

☹ フレーム単位の状態ラベルが必要

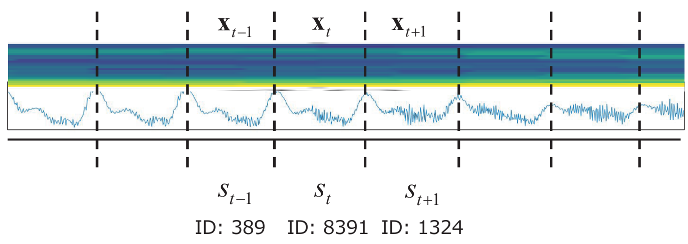
→ 別の手順で仮ラベルを用意して学習

学習手順

1. GMM-HMM を学習
2. （共有）状態の（ビタビ）アライメント
3. cross-entropy コストで DNN を学習
4. 必要があれば直接 sequence training
   - 事後確率を元に Back-propagation

42

深層学習: DNN-HMM

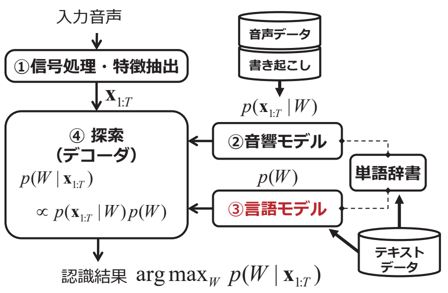


各 $\mathbf{x}_t$ に対し状態（クラス）s_t を識別するように学習

学習データ量: 100～1000時間分の音声データ

- 1秒間に100フレーム分のデータ点
- 100時間分で3000万以上のデータ点

43



③ 言語モデル

44

話の流れ

0. 補足
1. 言語モデルの役割

2. 記述文法モデル

3. 単語 N-gram モデル
- 最尤推定
- 0頻度問題
- 確率補完: 平滑化
- 深層モデル

確率モデル

次回

45

補足: テキスト/文字の計算機内での表現

色々な表現方法: 0と1の列を人が決める
- ASCIIコード: 2進表記と10進表記(整数)
  - ‘A’ : 01000001 - 65
  - ‘B’ : 01000010 - 66 …

例: “This” の数値表現 [84, 104, 105, 115]

- Uni-code
  - ‘あ’ : 111000111000000110000010

  …

テキスト/文字も（離散的な）数値列として表現

ただし，“値の大小（順序構造）” 自体には意味ない

46

補足: テキストデータの処理

表現の単位: 用途に応じてかわりうる
- 文字単位: 先ほどの文字コードなど
- 単語単位: 文字列を単語分割
  - 「明日」「の」「天気」「は」…
  - 文字と同様，単語ごとに数値(ID)を事前に割り当て

あ	0	明日	78
嗚呼	1	の	104
愛知	2	天気	1259
⋮		⋮	

通常，対応表を
あらかじめ作成

- サブワード単位: 文字と単語の間

以降，おおよそ単語を想定

47

補足: 単語の特徴量表現

例: 識別モデルで何かを予測

モデル: $f(\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{w}_3) = \sigma(\boldsymbol{a}_1^{\mathrm{T}}\boldsymbol{w}_1 + \boldsymbol{a}_2^{\mathrm{T}}\boldsymbol{w}_2 + \boldsymbol{a}_3^{\mathrm{T}}\boldsymbol{w}_3)$ sigmoid

入力: 単語列 ‘今日’, ‘の’, ‘天気’

[78,104,1259]

ダメな例 $f(w_1 = 78, w_2 = 104, w_3 = 1259)$

$= \sigma(a_1 78 + a_2 104 + a_3 1259)$

☹ IDを数値としてそのまま利用

理由: IDの値の比較やその大きさに意味がない

普通の数値演算を直接適用してはいけない

48

補足: 単語の特徴量表現

一般的な特徴量表現: one-hot vector
- 次元数=語彙サイズのベクトルを利用
- 特定の次元のみ1: 番号に対応した単語を表現

‘今日’: 78 ⇨ $\boldsymbol{w} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$ (78+1)次元目だけ1

- 実装レベル: “配列”の要素にアクセスする際の添え字の役割
  - 他: “確率”を格納した配列 (softmax 出力など)

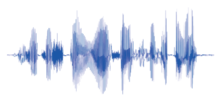
49

言語モデルの役割

50

音声認識における
言語モデル $p(W)$ の目的

言語的に尤もらしい文を選ぶ

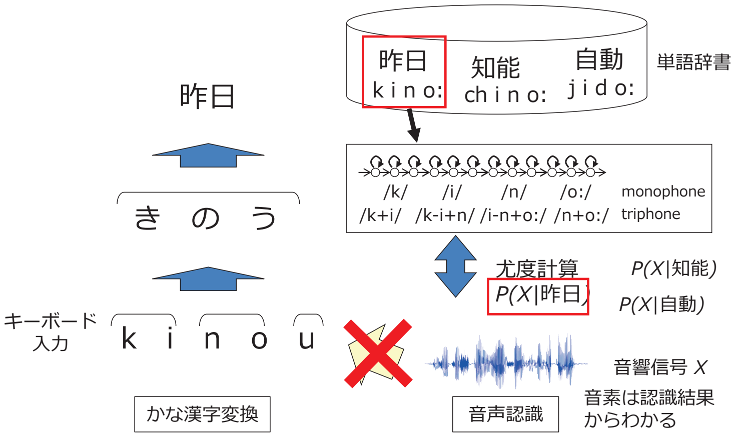


明石 / かかし / たかし / あ、菓子 に行きました

1. 音響的にはかなり近い
2. 最も（言語モデル）確率が高い漢字が選ばれる

51

かな漢字変換との処理の違い



52

（音声認識で用いられる）
言語モデルの大まかな種類

1. **記述文法モデル**
   - 受理すべき文を事前設定 – ハードなスコア
   - 人手 or テキストデータに基づき構築
     - 「文法」のようなルールによる表現も
2. **統計的言語モデル**
   - 確率モデルを仮定 – ソフトなスコア
     - 受理可能な文パターンは多い
   - テキストデータからパラメータを事前学習

53

1. 記述文法モデル

「出現する文」のみに限定したモデル

- 候補文の集合 W_G：候補間で重みは同じ

$$p(W) = \begin{cases} \text{const} & W \in W_G \\ 0 & \text{otherwise} \end{cases}$$

- 使いどころ
  - トピック，言い方などが限定される場合
  - 特定の型に当てはめたい場合
    （・・・）駅から（・・・）駅まで
    Hey, Siri（・・・）の予定を教えて

54

2. 統計的言語モデル

単語の言語モデル

- 単語列に対して出現確率を与えるモデル

$$W = w_1, w_2, \ldots, w_L$$
$$p(w_1, w_2, \ldots, w_L)$$

例： $\mathrm{P}(今日,は,良,い,天気,です,ね) = 1.158 \times 10^{-115}$
$\mathrm{P}(は,今日,良,です,天気,い,ね) = 5.275 \times 10^{-337}$

- 単語単位：事前に文から単語への分かち書き（と読みの付与）→ 形態素解析
- **単語の集合（語彙）は Given**

2. 統計的言語モデル

言語モデルの目的

– 単語列の確率が高い＝出現しやすい言語表現

– 良い言語モデル = 認識したい言語表現に高い出現確率を与える

統計的言語モデル

– データから統計的手法によって確率を推定

– 確率モデルを設定し，データからパラメータを学習

N-gram モデル

<u>統計的言語モデルとして最も広く利用</u>

仮定

– 直前の (N-1) 単語のみに出現確率が依存

$$p(w_1, w_2, \ldots, w_L) = p(w_2, \ldots, w_L \mid w_1) p(w_1)$$

条件付き確率

$$= p(w_3, \ldots, w_L \mid w_1, w_2) p(w_2 \mid w_1) p(w_1)$$

$$= p(w_4, \ldots, w_L \mid w_1, w_2, w_3) p(w_3 \mid w_2, w_1) p(w_2 \mid w_1) p(w_1)$$

$$\approx p(w_1) \cdots p(w_N \mid w_{N-1}, \ldots, w_1) \prod_n p(w_n \mid \underline{w_{n-1}, \ldots, w_{n-N+1}})$$

履歴自体が隠れてない「状態」
自己回帰モデル

N-gram モデル

コンテキスト長 N

– どの程度の過去の単語まで依存するか

N = 1： ユニグラム（unigram）

N = 2： バイグラム（bigram）

N = 3： トライグラム（trigram）

– 実用上，N=3 程度は必要

$$p(w_1, w_2, \ldots, w_L) =$$

$$\approx p(w_1) \cdots p(w_N \mid w_{N-1}, \ldots, w_1) \prod_n p(w_n \mid w_{n-1}, \ldots, w_{n-N+1})$$

トライグラムの例

「大学/に/行/く」という単語列の出現確率は？

P(<s>, 大学，に，行, く, </s>)
= P(大学|<s>)
× P(に|<s>, 大学)
× P(行|大学, に)
× P(く|に, 行)
× P(</s>|行, く)

※ <s>, </s>はそれぞれ文頭，文末を表す

N-gram の確率モデル

出現確率パラメータ π

$$p(w_n \mid w_{n-1}, \ldots, w_{n-N+1}) = \pi_{w_n \mid w_{n-1}, \ldots, w_{n-N+1}}$$

$$\sum_w \pi_{w \mid w_{n-1}, \ldots, w_{n-N+1}} = 1$$

例

$$p(\text{今日},\text{の},\text{天気}) = p(\text{今日} \mid <s>) p(\text{の} \mid <s>, \text{今日})$$
$$p(\text{天気} \mid \text{今日},\text{の}) p(</s> \mid \text{天気},\text{の})$$

$$= \pi_{\text{今日}|<s>} \pi_{\text{の}|<s>,\text{今日}} \pi_{\text{天気}|\text{今日},\text{の}} \pi_{</s>|\text{天気},\text{の}}$$

N-gram 確率の最尤推定

最尤推定: トライグラムの場合

– 全単語列データ $\mathbf{w}$

– コスト（対数尤度） $\ln p(\mathbf{w}) = \sum_n \ln \pi_{w_n \mid w_{n-1}, w_{n-2}}$

$$J = \sum_{a,b} \lambda_{a,b} \sum_w (1 - \pi_{w|a,b}) + \sum_n \ln \pi_{w_n \mid w_{n-1}, w_{n-2}}$$

– パラメータ推定

$N_{c|a,b}$(a,b,c) の出現回数

$$\frac{\partial J}{\partial \pi_{c|a,b}} = -\lambda_{a,b} + \frac{N_{c|a,b}}{\pi_{c|a,b}} = 0 \quad \Rightarrow \quad \pi_{c|a,b} = \frac{N_{c|a,b}}{\lambda_{a,b}}$$

$$\Rightarrow \quad 1 = \frac{1}{\lambda_{a,b}} \sum_w N_{w|a,b} \quad \Rightarrow \quad \pi_{c|a,b} = \frac{N_{c|a,b}}{\sum_w N_{w|a,b}}$$

N-gram 確率の最尤推定

まとめ: Nグラム確率

– 出現数のカウントで計算可能

$$p(w_n \mid w_{n-1}, w_{n-2}) = \frac{N(w_n, w_{n-1}, w_{n-2})}{N(w_{n-1}, w_{n-2})}$$

– $N(w_n, w_{n-1}, w_{n-2})$ ：単語の三つ組み w_{n-2}, w_{n-1}, w_n の出現回数

– $N(w_{n-1}, w_{n-2})$：単語2つ組の w_{n-2}, w_{n-1} の出現回数

N-gram確率算出の例

$$P(\text{晴れ}|\text{天気 は}) = \frac{N(\text{天気 は 晴れ})}{N(\text{天気 は})}$$

3-gramの出現回数

2-gramの出現回数

テキストデータ

2-gram	出現回数
…	
今日 の	200
天気 は	120
の 天気	140
…	

3-gram	出現回数
…	
今日 の 天気	25
天気 は 晴れ	10
の 天気 は	15
…	

語彙制限とカットオフ

単純な N-gram の構築

– 学習データから素直に作成

→ 語彙 及び N-gramカウントが爆発的に増大

例：60k語の3-gram → (60k)^3 = 216,000,000,000,000 (216Tエントリ)

☹ 音声認識時に必要なメモリ量の増大

☹ 不要な単語の出現

☹ 数回しか出てない組み・信用できない確率

語彙制限とカットオフ

語彙制限

– 扱う語彙を一定サイズ（60k等）に制限

– 除外単語:「未知語」(UNK; Unknown) 扱い

カットオフ

– 出現頻度の少ない N-gramカウントを除外

- 3-gram エントリのほとんどの生起回数は 0
- 数百万語の英語コーパス: 半数以上の 3-gramが1回しか出現しない
- 80%の3-gramが5回以下しか出現しない

– データスパースネス (data sparseness)

確率の平滑化

最尤推定の問題点

– 学習データにたまたま出現しなかったN-gramの出現確率が0になる

– **ゼロ頻度問題**

☹ その N-gram は認識候補にあがらない

対応: 確率の平滑化 (smoothing)

– 確率値を頻度の低いエントリにも分け与える

– 確率が0になるのを防ぐ

確率の平滑化法

a) 加算スムージング（最も単純）

b) バックオフ平滑化 (back-off smoothing)

– ベイズモデル

d) 線形補間法 (linear interpolation)

– EM で係数を求める

加算スムージング

方法: ゲタをはかせる

– 各 N-gram が δ回多く現れたとみなす

$$P(w_n | w_{n-1}, w_{n-2}) = \frac{N(w_n, w_{n-1}, w_{n-2}) + \delta}{N(w_{n-1}, w_{n-2}) + \delta V}$$

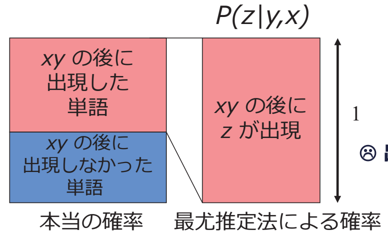
V: 単語列の異なり総数, 0 < δ ≦ 1

– ディリクレ事前分布による MAP 推定も似たようなもの

バックオフ平滑化（1）

方法: 階層的に平滑化

– 存在しない N-gram を表すのに (N-1以下)-gram を使って推定

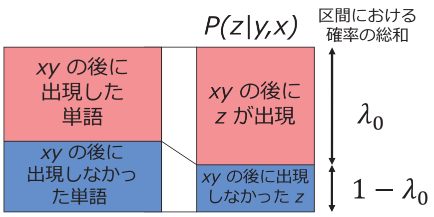


☹ **出現しなかった3-gramには確率は与えられない**

バックオフ平滑化（2）

ディスカウンティング: 割引

– 既知のN-gramの出現確率の一部を未知のN-gramに割り当て

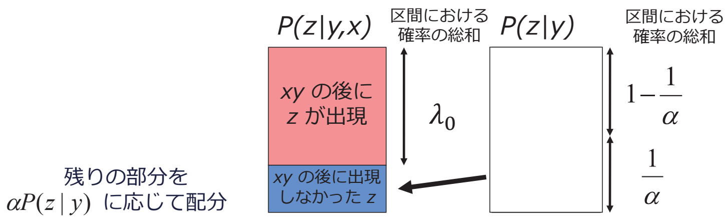


平滑化によって、出現しなかった3-gramに確率を配分

バックオフ平滑化（3）

再配分:

– 未知のN-gramの確率を（N-1)-gramの確率で比例配分

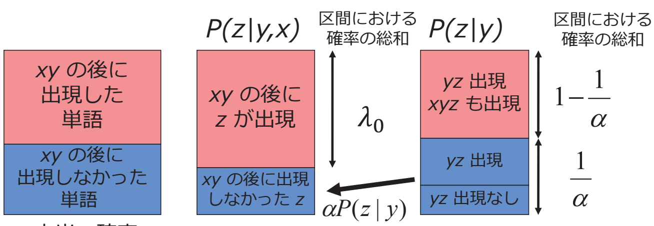


未知の3-gram確率を2-gram確率に応じて配分

バックオフ平滑化（3）

確率の割引＋再配分:

– 未知のN-gramの確率を（N-1)-gramの確率で比例配分



未知の3-gram確率を2-gram確率に応じて配分

バックオフ平滑化（4）

具体的な計算

– 最尤推定に基づく確率を計算

$$f(w_i | w_{i-1}, w_{i-2}) = \frac{N(w_i, w_{i-1}, w_{i-2})}{N(w_{i-1}, w_{i-2})}$$

補正後の確率を P と表記するため f で表記

– ディスカウント係数を導入（確率の割引）

$\lambda(w_i, w_{i-1}, w_{i-2})$

- **決め方は次回: とりあえず定数扱いで given**
- N-gram 毎に定まる

バックオフ平滑化（５）

補正後の N-gram 確率

$$P(w_i \mid w_{i-1}, w_{i-2}) =$$

$$\begin{cases} \overbrace{\lambda(w_i, w_{i-1}, w_{i-2}) f(w_i \mid w_{i-1}, w_{i-2})}^{\lambda_0(w_{i-1}, w_{i-2})} & \mathit{if}\ N(w_i, w_{i-1}, w_{i-2}) > 0 \\ [1-\lambda_0(w_{i-1}, w_{i-2})]\alpha P(w_i \mid w_{i-1}) & \mathit{otherwise} \end{cases}$$

ただし，2ページ前の割合の関係から

$$\lambda_0(w_{i-1}, w_{i-2}) = \sum_{w:N(w_i, w_{i-1}, w_{i-2})>0} \lambda(w, w_{i-1}, w_{i-2}) f(w \mid w_{i-1}, w_{i-2})$$

$$\frac{1}{\alpha} = 1 - \sum_{N(w_i, w_{i-1}, w_{i-2})>0} P(w_i \mid w_{i-1})$$

$w_{i-2}w_{i-1}w_i$ が存在した 2-gram $w_{i-1}w_i$ の総和

α の値は $P(w_i \mid w_{i-1}, w_{i-2})$ の合計が1から導く（帳尻合わせ）

Mini Quiz #2

- N-gram 確率は，パラメータ推定に用いるテキストによって偏りが生じる．
  - 例えば，スポーツに関する記事のテキスト，天気に関するテキストなど，明らかに単語の出現確率は異なると考えられる．
- 音声認識で用いられる場合，N-gram 言語モデルにおける確率の偏りはどういう状況では良くて，どういう状況だと良くないのだろうか．

バックオフにおける ディスカウント法

ディスカウント係数 $\lambda(w_i, w_{i-1}, w_{i-2})$ の決定法には様々なものがある： ディスカウント法

1. 線形法
2. グッド・チューリング (Good-Turing) 法
3. ウィッテン・ベル (Witten-Bell) 法
4. 絶対法
5. Kneser-Ney 法
6. 階層 Pitman-Yor （ベイズモデル）

バックオフにおける ディスカウント法

ディスカウント係数 $\lambda(w_i, w_{i-1}, w_{i-2})$ の決定法には様々なものがある： ディスカウント法

1. 線形法
2. グッド・チューリング (Good-Turing) 法
3. ウィッテン・ベル (Witten-Bell) 法
4. 絶対法
5. Kneser-Ney 法
6. 階層 Pitman-Yor （ベイズモデル）

階層 Pitman-Yor 言語モデル [Teh'06]

ベイズ的にスムージング

- ディスカウント付きの確率的な生成モデル
  - Pitman-Yor 過程を階層化
  - 潜在変数: 基底分布から生成されたか，などを表現
- 高性能な Kneser-Ney smoothing と一致

$$p(w \mid h) = \sum_{k=1}^{t_h} \frac{n_{w,k,h} - d_h}{n_{*,*,h} + \theta_h} + \frac{\theta_h + d_h t_h}{n_{*,*,h} + \theta_h} p(w \mid h')$$

N-gram 確率　　(N-1)-gram 確率

h : コンテキスト　$h = [w_{n-1}, w_{n-2}]$　　θ_h, d_h : パラメータ

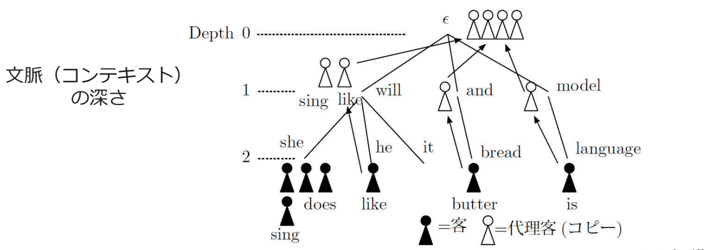
h' : オーダを１つ落としたコンテキスト　$h = [w_{n-1}]$　　$n_{w,k,h}$: 頻度

3-gram → 2-gram に相当

階層 Chinese Restaurant Process によるイメージ図

ある確率で代理客（カウント）をより上位の店（短いコンテキスト）に送り込む

= たまに **N-gram と (N-1)-gram 以下の頻度を同時にカウント**



http://chasen.org/~daiti-m/paper/nl178vpylm.pdf から拝借

Nested Pitman-Yor 言語モデル [mochihashi '09]

単語 1-gram を文字 N-gram で補完

- 単語 1-gram 確率の基底分布に 文字 N-gram 確率を利用
- 「未知の"単語"」に対する確率も割り当て可
  - 文字が1度でも出現していれば

応用: 教師なしでの文字列からの単語分割

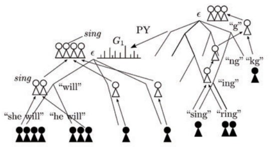


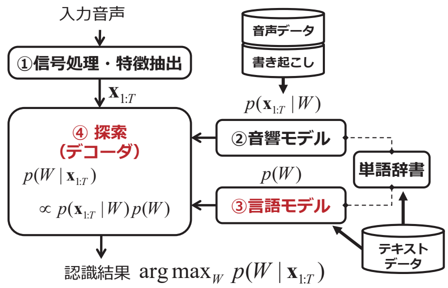
Figure 2: Chinese restaurant representation of our Nested Pitman-Yor Language Model (NPYLM).

Inference: ブロック化 Gibbs サンプリング
（forward filtering → backward sampling
cf. HMM の forward-backward algorithm）

http://chasen.org/~daiti-m/paper/acl2009segment.pdf
より拝借

次回（最後）

- ニューラル言語モデル
- デコーディング



参考文献

- C.M.ビショップ「パターン認識と機械学習 上下」
- 金森敬文 他「機械学習のための連続最適化」
- 河原達也 「音声認識システム」
- 安藤彰男 「リアルタイム音声認識」

参考文献

Y. W. The: "A Hierarchical Bayesian Language Model Based On Pitman-Yor Processes," In proc. of ACL, pp.985-992, 2006.

Mochihashi *et al.*: "Bayesian Unsupervised Word Segmentation with Nested Pitman-Yor Language Modeling", In proc .of ACL, pp.100-108, 2009.