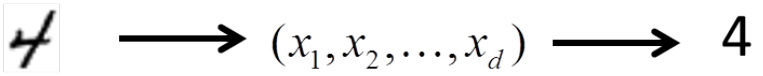
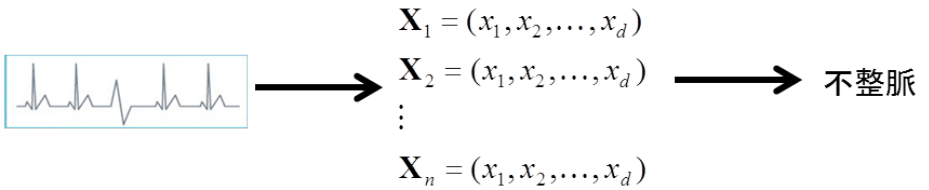
10．声をモデル化してみよう

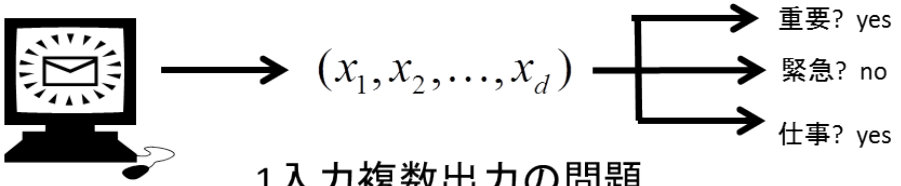
- 入出力数の違いによるパターン認識問題の分類



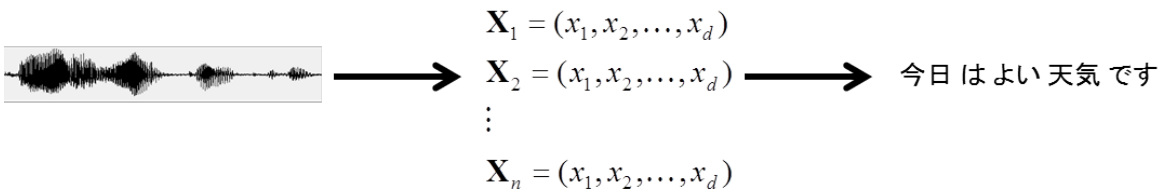
1入力1出力の問題



複数入力1出力の問題



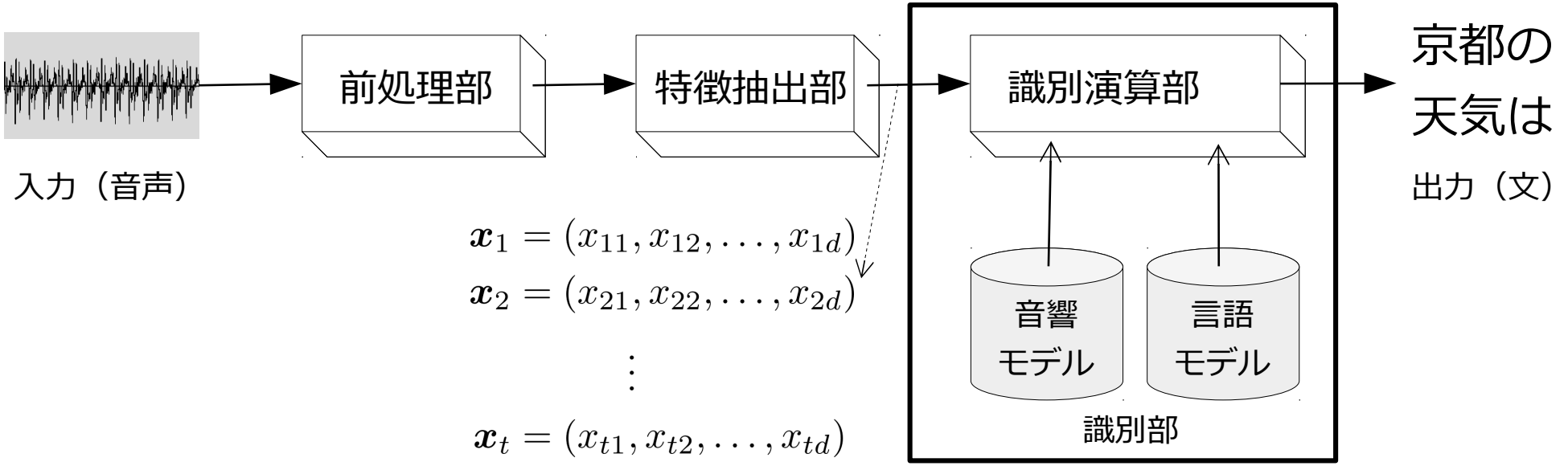
1入力複数出力の問題



音声認識

複数入力複数出力の問題

10.1 連続音声の認識


入力（音声）
前処理部
特徴抽出部
識別演算部
京都の
天気は
出力（文）
$\boldsymbol{x}_1 = (x_{11}, x_{12}, \ldots, x_{1d})$
$\boldsymbol{x}_2 = (x_{21}, x_{22}, \ldots, x_{2d})$
$\boldsymbol{x}_t = (x_{t1}, x_{t2}, \ldots, x_{td})$
音響
モデル
言語
モデル
識別部

10.1 連続音声の認識

- 統計的音声認識の定式化
  - 入力系列 $\boldsymbol{x}$ のもとで事後確率を最大にする単語列 $\hat{\boldsymbol{w}}$ を認識結果とする

$$\begin{aligned}\hat{\boldsymbol{w}} &= \arg\max_{\boldsymbol{w}} P(\boldsymbol{w}|\boldsymbol{x}) \\ &= \arg\max_{\boldsymbol{w}} \frac{p(\boldsymbol{x}|\boldsymbol{w})P(\boldsymbol{w})}{p(\boldsymbol{x})} \\ &= \arg\max_{\boldsymbol{w}} p(\boldsymbol{x}|\boldsymbol{w})P(\boldsymbol{w})\end{aligned}$$

ベイズの定理

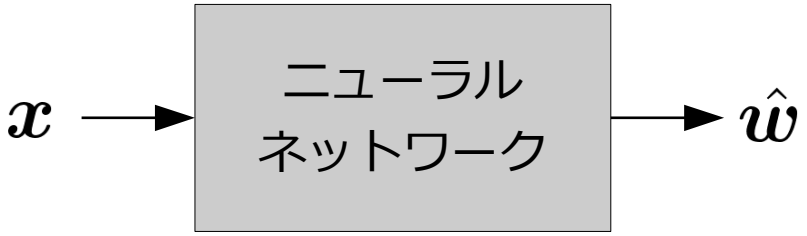
  - 音響モデル $p(\boldsymbol{x}|\boldsymbol{w})$
  - 言語モデル $P(\boldsymbol{w})$

10.1 連続音声の認識

- 一昔前の解法
  - 音響モデル $p(\boldsymbol{x}|\boldsymbol{w})$
    - 隠れマルコフモデル (HMM)
    - DNN-HMM 法
  - 言語モデル $P(\boldsymbol{w})$
    - 文法記述
    - n-gram + スムージング
    - RNN 言語モデル
  - 事後確率最大となる $\hat{\boldsymbol{w}}$
    - ヒューリスティック探索
    - WFST

10.1 連続音声の認識

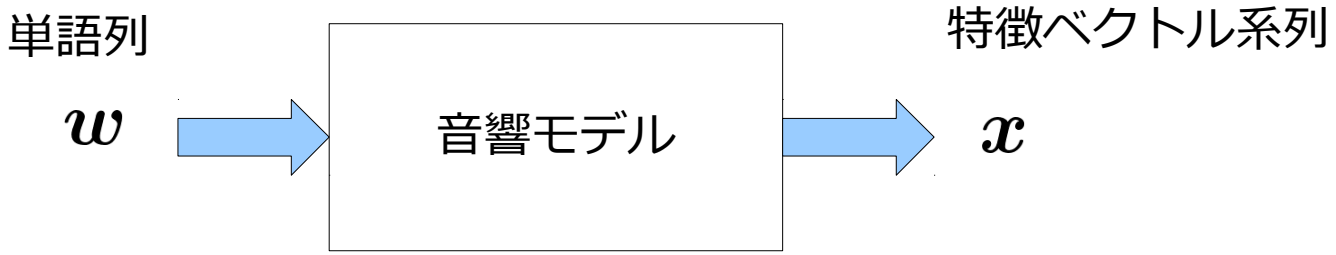
- 現在の主流の解法
  - end-to-end ニューラルネットワーク



    - 参考資料：形態素解析も辞書も言語モデルもいらない end-to-end 音声認識 https://www.slideshare.net/t_koshikawa/endtoend
- 1 文を越えた処理が必要になってくれば、ブラックボックスでは限界があるかも

10.2 音響モデルの作り方

- 音響モデル $p(\boldsymbol{x}|\boldsymbol{w})$ とは
  - *p*(特徴ベクトル系列 | 単語列) を計算するための確率モデル



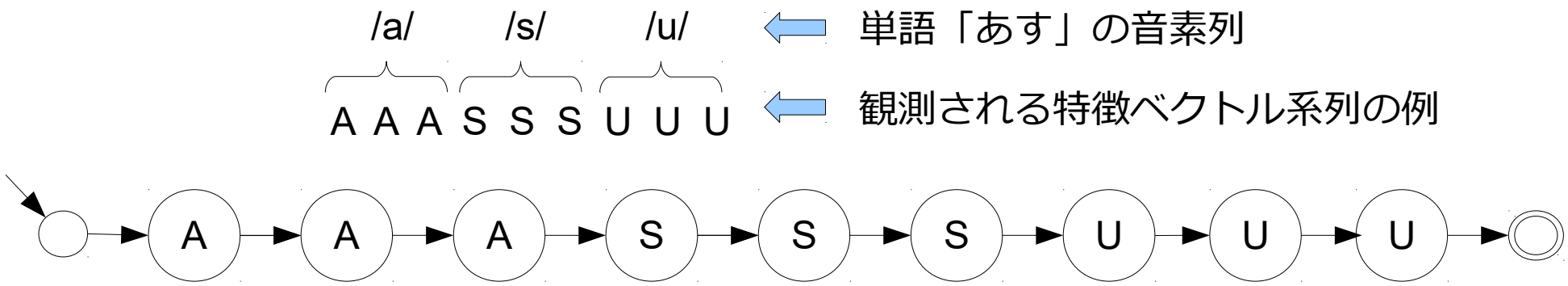
  - まず、単純化のために単語認識問題を扱う
    - 単語は音素の系列で表現されているとする

10.2　音響モデルの作り方

- 設定 1
  - 各音素あたりの特徴ベクトル数が一定
  - 特徴ベクトルを離散値（記号）で近似したときに誤りがない

    → 単語ごとの有限状態オートマトンでモデル化

    – 受理すれば p>0，不受理ならば p=0

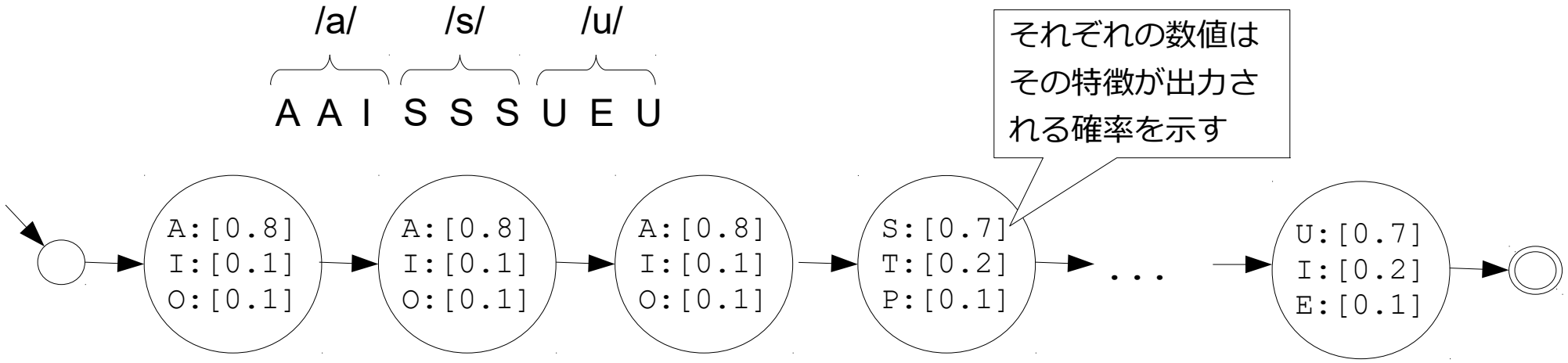


10.2　音響モデルの作り方

- 設定 2
  - 各音素あたりの特徴ベクトル数が一定
  - 特徴ベクトルの近似に誤りがあり得る

    → 単語ごとの確率オートマトンでモデル化
    - 各状態で、全てのシンボルに何らかの生成確率を与える

      p= 各状態における記号の生成確率の積

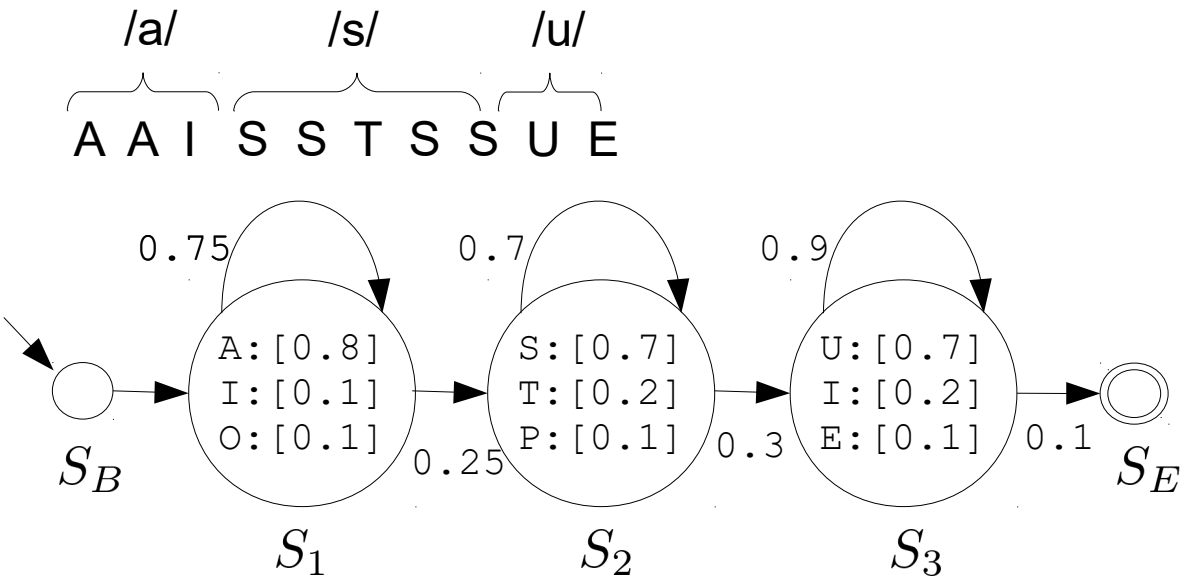


10.2　音響モデルの作り方

- 設定 3
  - 各音素あたりの特徴ベクトル数が不定
  - 特徴ベクトルの近似に誤りがあり得る

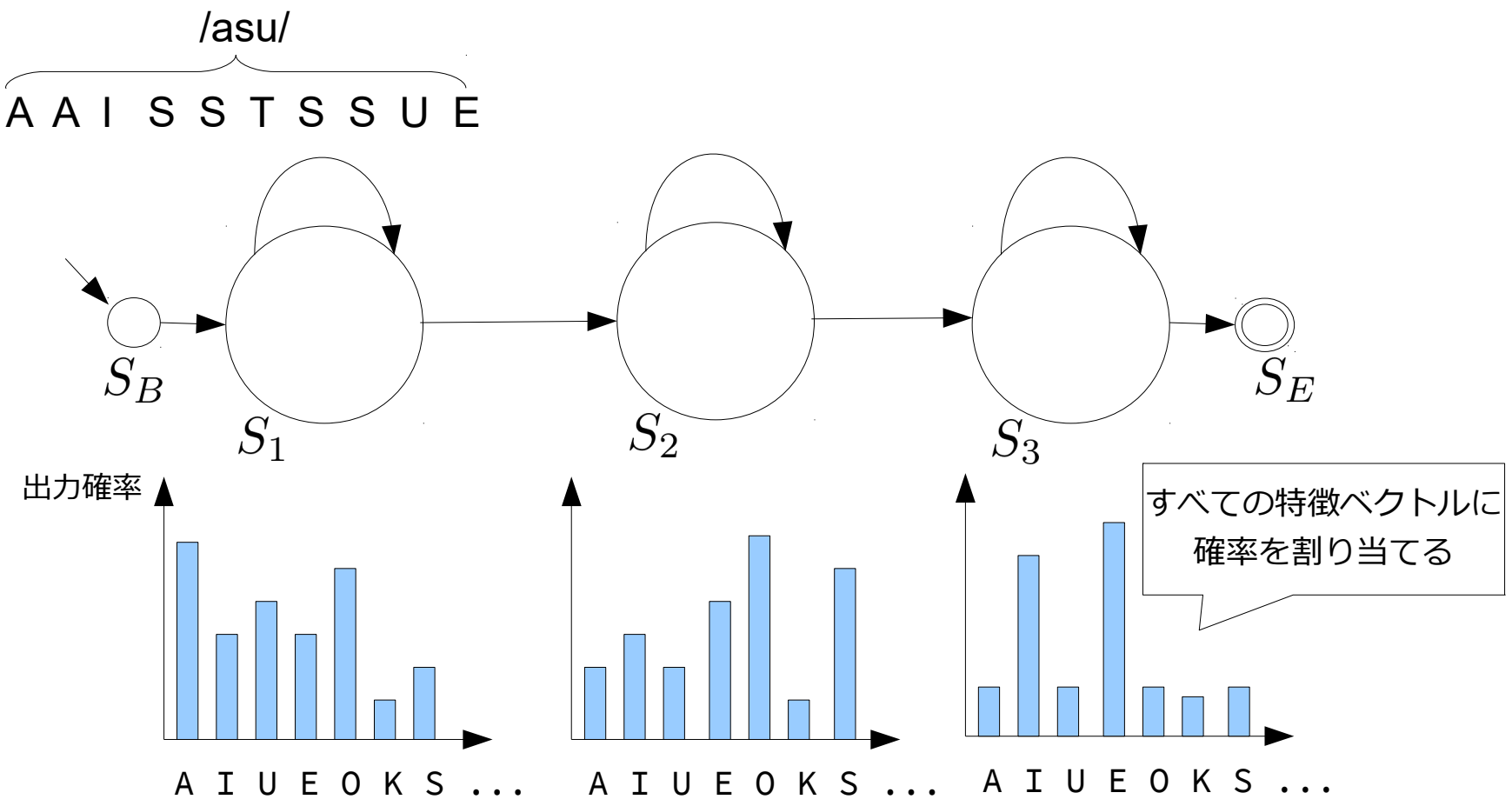
    → 非決定性確率オートマトン (=HMM) でモデル化
    - 各状態からの遷移が非決定的かつ確率的

    p= 「各状態における記号の生成確率と遷移確率の積」の可能な遷移に対する和



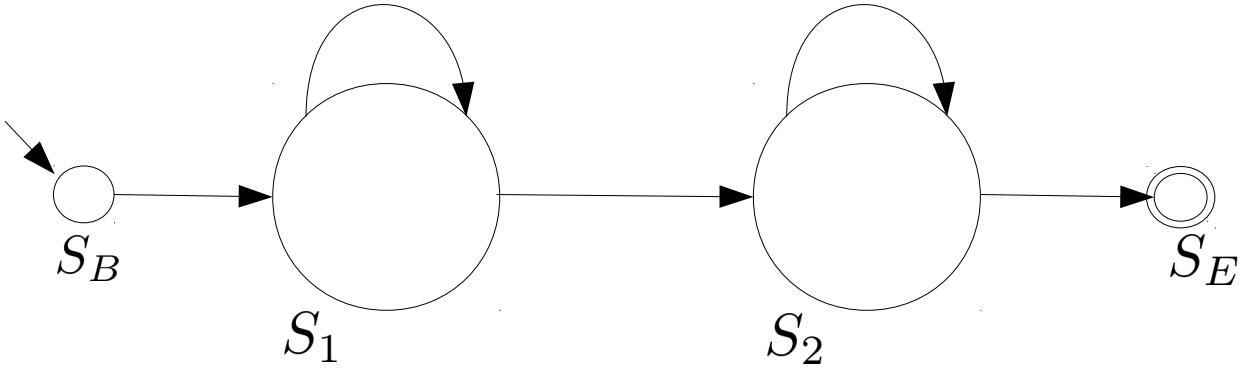
10.2　音響モデルの作り方

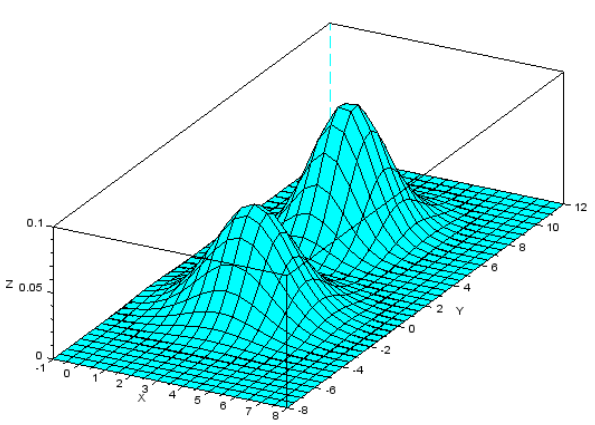
- 設定 4
  - 各状態ですべての特徴ベクトルに対して正の確率を割り当てる　→　状態遷移情報が隠れてしまう

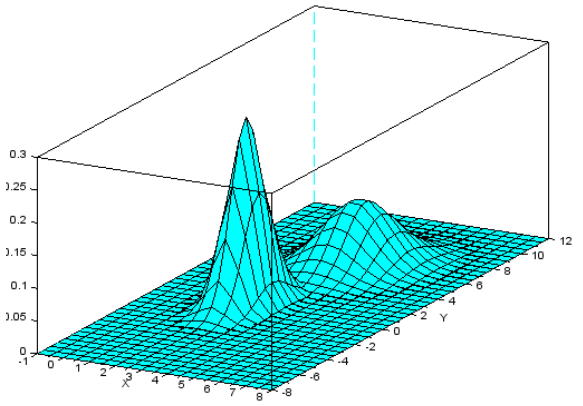


10.2　音響モデルの作り方

- 実際の HMM
  - 各状態での特徴ベクトルの生成確率を混合正規分布で表現





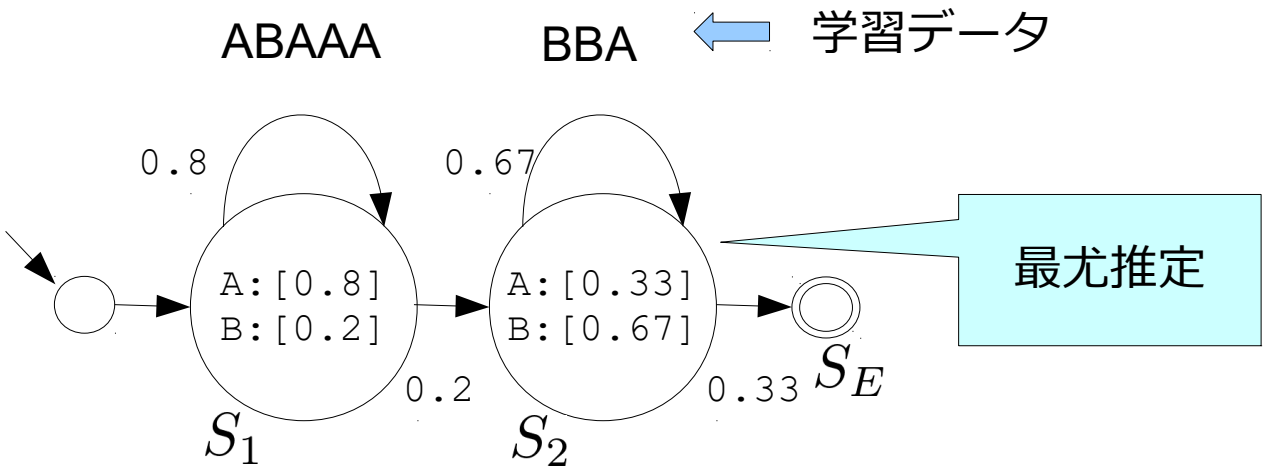


10.4 音響モデルの鍛え方

- HMM の学習
  - 離散記号：最尤推定
  - 連続値：パラメトリックな学習
    - 確率密度関数の平均と共分散行列を学習する
- 学習における問題点
  - 学習データに対して状態遷移系列がわからない

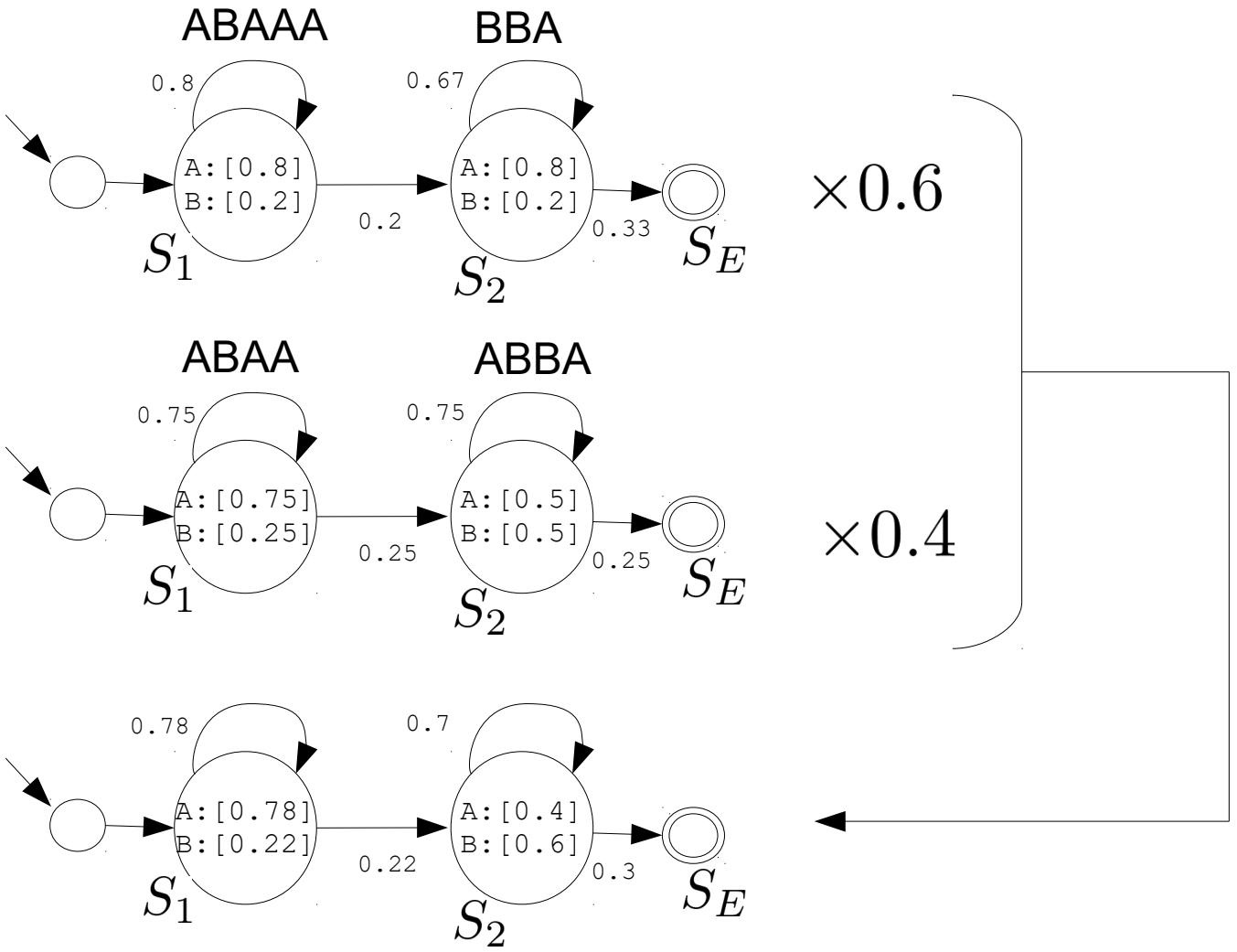
10.4 音響モデルの鍛え方

- 状態遷移系列が既知であれば
  - 状態遷移確率
    - 状態からの遷移を数え上げることによって学習可能
  - 信号出力確率
    - 状態ごとに平均・分散を計算することで学習可能



10.4 音響モデルの鍛え方

- 状態遷移系列の確率がわかっていれば
  - 学習結果の重み付き加算

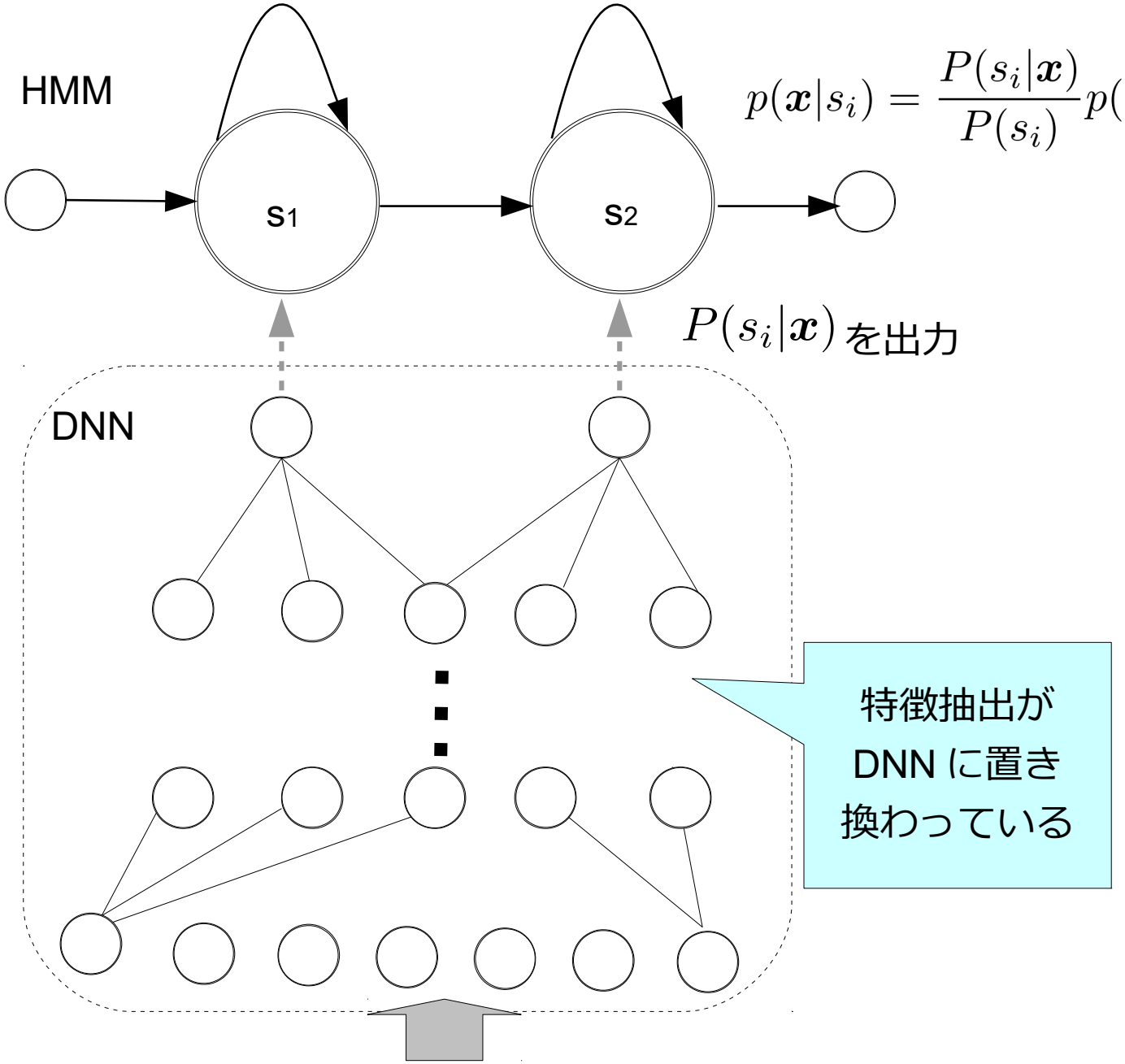


10.4 音響モデルの鍛え方

- Baum-Welch 法による HMM の学習
  - HMM のパラメータを適当な初期値に設定
  - E(Expectation) ステップ
    - 学習データ (入力) に対して、状態遷移を与えたときの確率を現在の HMM を用いて計算
    - それを全ての可能な状態遷移について求める (実際は d 動的計画法を用いて効率的に計算)
  - M(Maximization) ステップ
    - E ステップで得られたデータから HMM のパラメータを最尤推定
  - E,M ステップをパラメータの変化量が一定値以下になるまで繰り返し

ディープニューラルネットを用いた音声認識

DNN-HMM 法



12. 文法規則を書いてみよう

- 言語モデルとは
  - P(単語列) を計算するための確率モデル
- 2 つのアプローチ
  - 文法記述
    - 単語から文を構成する規則を文法として記述
    - 文法が受理する単語列 W に対して *P(W)>0*, そうでなければ *P(W)=0*
  - 統計的言語モデル
    - 大量のコーパスを元に確率を推定
    - $P(W) = P(w_1, \ldots, w_n)$ を何らかの近似で計算

12.2　タスクから文法を設計する

- 例題タスク
  - 新幹線の切符自動販売機の音声インタフェース
  - 機能
    - 乗車区間を指定できる
    - 席の種類を指定できる
    - 枚数を指定できる
  - 例文
    - 「東京から京都まで自由席 1 枚」
    - 「名古屋から品川までグリーン席 3 枚」

12.2 タスクから文法を設計する

- 文法 = 出現可能な単語列パターンの定義
  - 文のパターンを句の並びで定義
    - $ 文　→　$ 区間　$ 席種　$ 枚数
    - 例）東京から京都まで自由席 1 枚
  - 句のパターンを単語または単語集合の並びで定義
    - $ 区間　→　$ 駅名　から　$ 駅名　まで
  - 認識対象とする単語集合（ = 語彙）を定義
    - $ 駅名→東京 | 品川 | 新横浜 | …
    - $ 席種→グリーン席 | 指定席 | 自由席

13 章　統計的言語モデルを作ろう

- 統計的言語モデルとは
  - P(単語列) を言語統計から計算
  - 正しい文には高い確率を与えたい
  - 誤っている文には低い確率を与えたい

13.1 文の出現確率の求め方

- 単語列 $\boldsymbol{w}$ の生成確率

$$
\begin{aligned}
P(\boldsymbol{w}) &= P(w_1, \ldots, w_n) \\
&= P(w_1)P(w_2|w_1)P(w_3|w_1, w_2) \ldots P(w_n|w_1, \ldots, w_{n-1})
\end{aligned}
$$

- $P(w_i|w_1, \ldots, w_{i-1})$ の近似
  - 1- グラム ～ $P(w_i)$
  - 2- グラム ～ $P(w_i|w_{i-1})$
  - 3- グラム ～ $P(w_i|w_{i-2}, w_{i-1})$

13.2 N- グラム言語モデル

- N- グラム言語モデルとは
  - 単語の生起を (N-1) 重マルコフ過程で近似したモデル
  - ある時点での単語の生起確率は直前の N-1 単語にのみ依存すると仮定
  - 3- グラムによる単語列 $w_1, \ldots, w_n$ の生成確率

$$P(w_1, \ldots, w_n) = P(w_1)P(w_2|w_1)\prod_{k=3}^{n} P(w_k|w_{k-2}, w_{k-1})$$

13.2 N- グラム言語モデル

1.コーパスを準備する

- 大量の電子化された文章を集める

  例）新聞記事 DVD-ROM, Web, etc.

2.単語に区切る

- 英語の場合：空白で区切る
- 日本語の場合：形態素解析が必要

3.条件付き確率を求める

- スパースネスの問題を解決したうえで $P(w_i|w_{i-2}, w_{i-1})$ を求める

13.2 N- グラム言語モデル

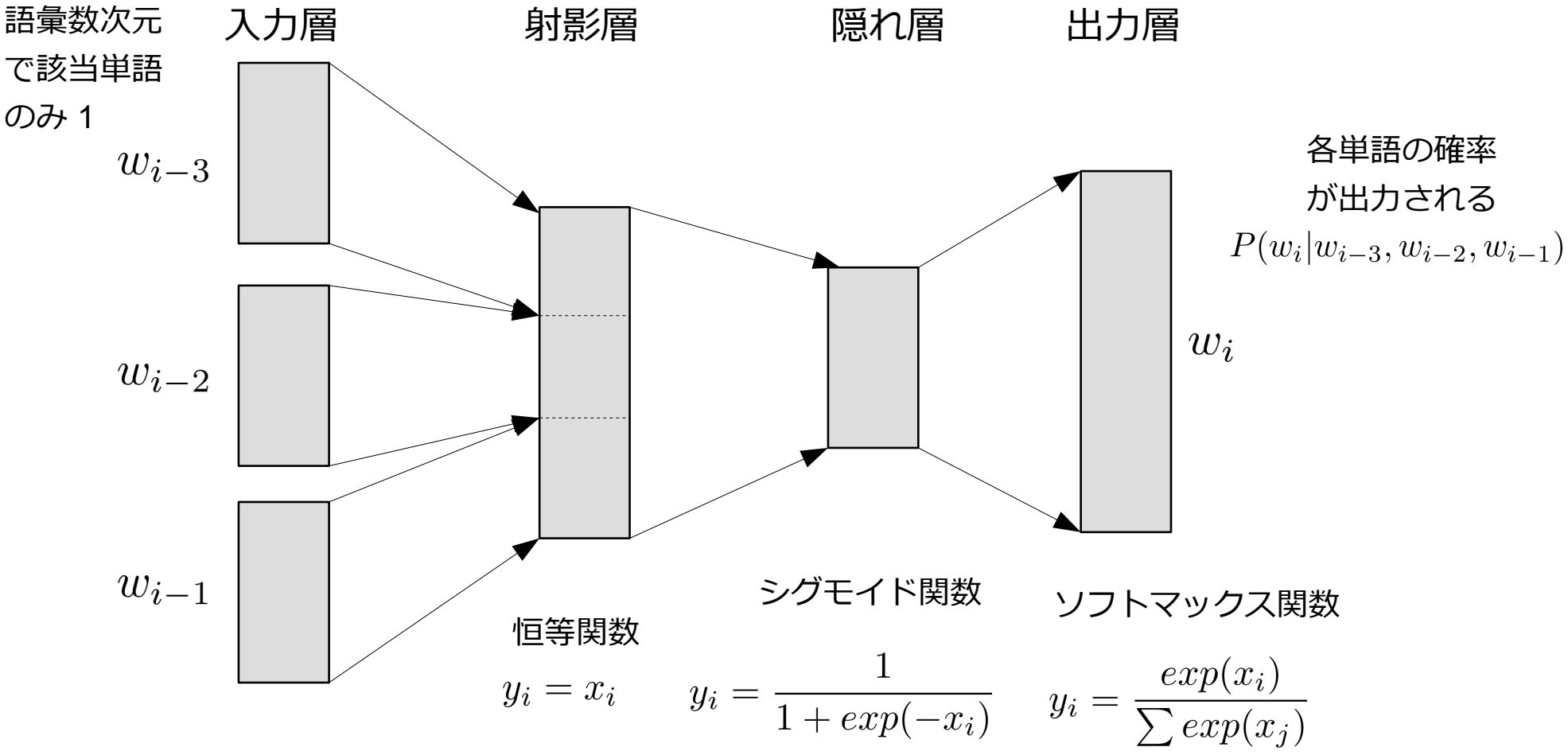
- 3- グラム確率の推定
  - 最尤推定を用いる
  - C(w): 単語列 w の出現回数

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

  - $P(w_i|w_{i-2}, w_{i-1}) = f(w_i|w_{i-2}, w_{i-1})$ とするとスパースネスの問題が生じる
    - 妥当な単語列であっても偶然コーパスに出現しなければ 3- グラムの確率が 0 になる
    - 補間法、スムージングなどで対処

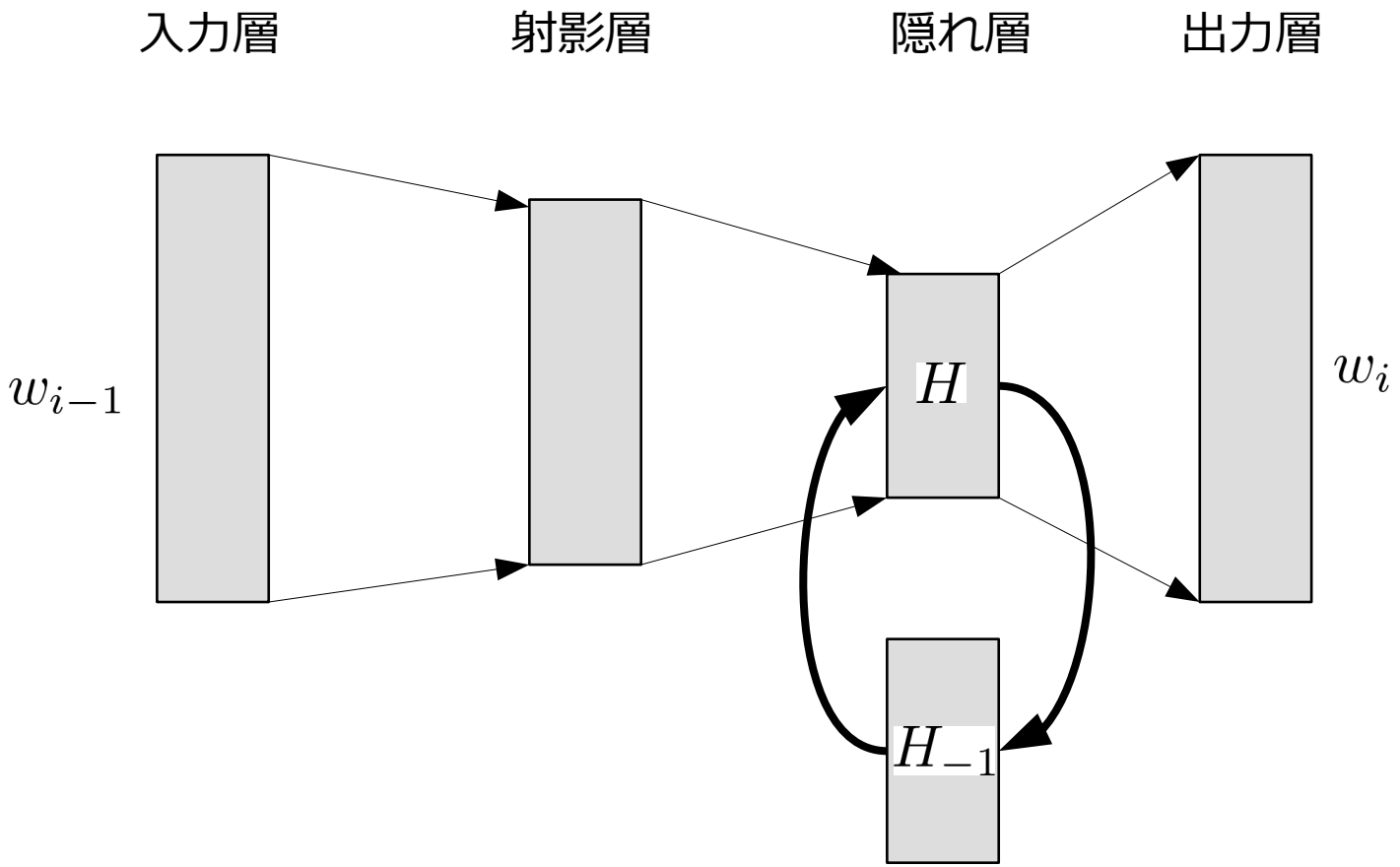
13.5 ニューラルネットワークを用いた言語モデル

- フィードフォワード型
  - 過去 N 単語から次単語の確率分布を求める



13.5 ニューラルネットワークを用いた言語モデル

- リカレント型
  - フィードバックで仮想的にすべての履歴を表現



14. 連続音声認識に挑戦しよう

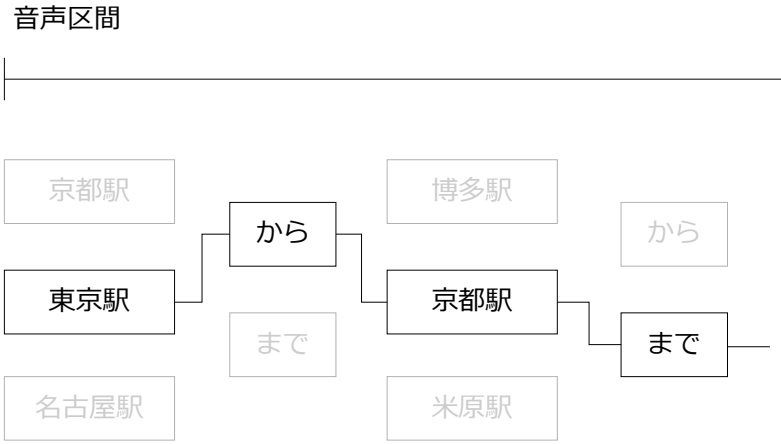
- 音声認識の原理

$$\hat{\boldsymbol{w}} = \arg\max_{\boldsymbol{w}} P(\boldsymbol{w}|\boldsymbol{x}) = \arg\max_{\boldsymbol{w}} P(\boldsymbol{x}|\boldsymbol{w})P(\boldsymbol{w})$$

  - 入力 $\boldsymbol{x}$ のもとで事後確率 $P(\boldsymbol{w}|\boldsymbol{x})$ を最大にする単語列 $\hat{\boldsymbol{w}}$ を認識結果とする
  - $P(\boldsymbol{x}|\boldsymbol{w})$: 音響モデル ...HMM を用いて計算
  - $P(\boldsymbol{w})$: 言語モデル ...N-gram を用いて計算
- 問題点
  - 大語彙（数千語以上）の場合、全ての可能な $\boldsymbol{w}$ をリストアップすることは不可能

14.1 基本的な探索手法

- 探索の導入
  - 膨大な候補から解になりそうな部分のみに絞る

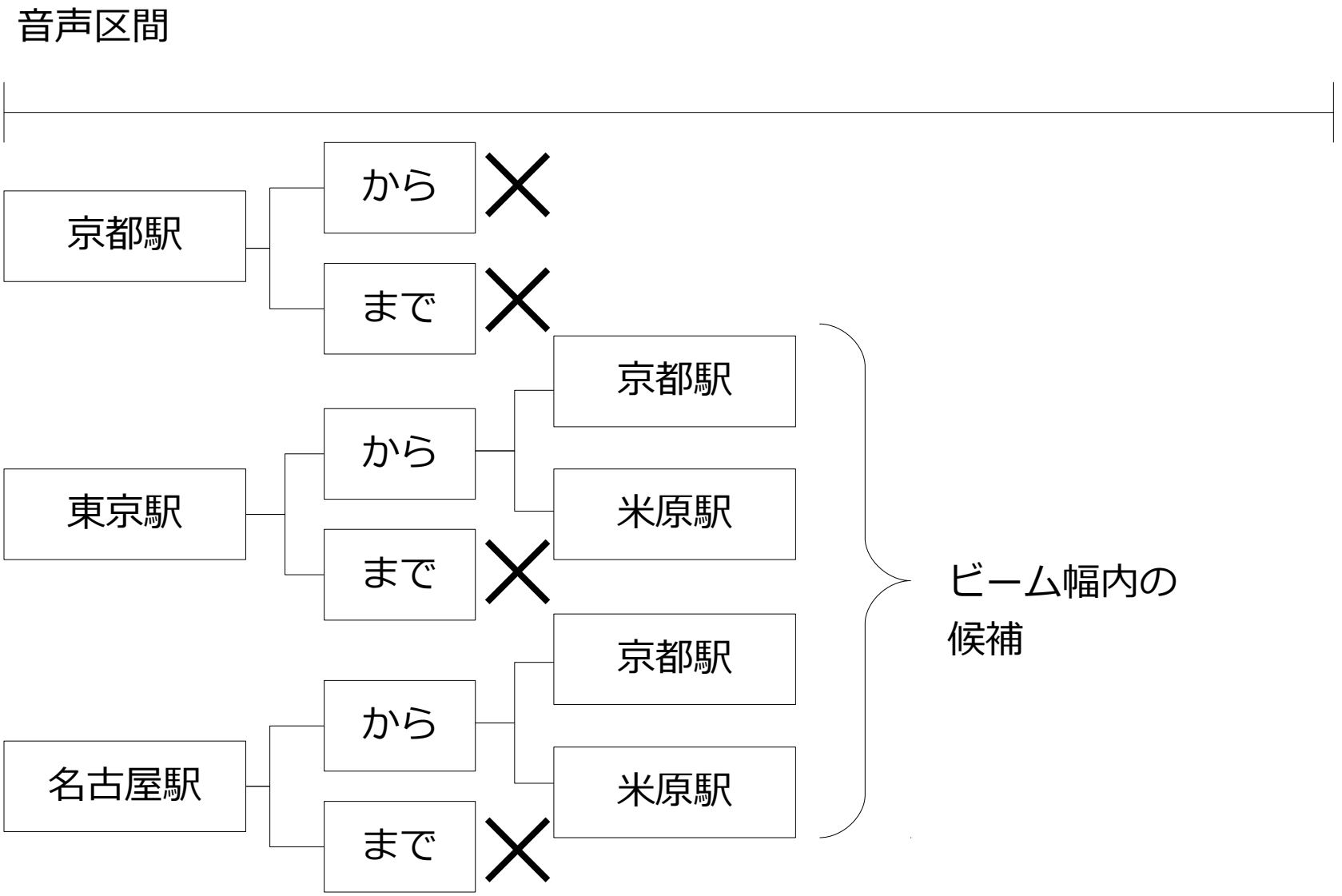


- 縦型探索
- 横型探索
- ビームサーチ

- 解の絞り方
  - 評価値の高い候補を優先する
    ... ヒューリスティックサーチ
  - 探索空間を静的に展開し、最適化 ...WFST

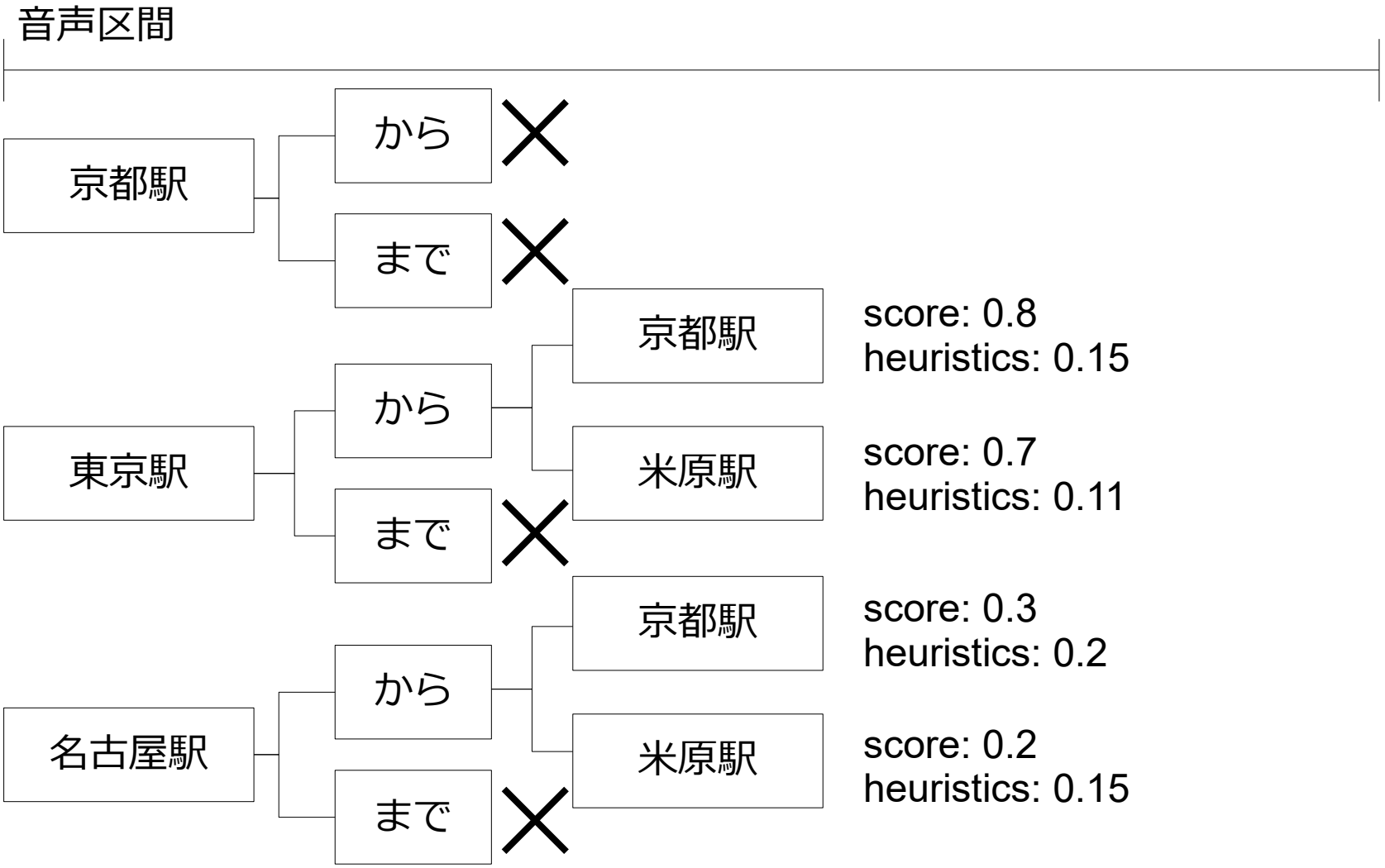
14.1 基本的な探索手法

- ビームサーチ：探索幅（ビーム）の導入



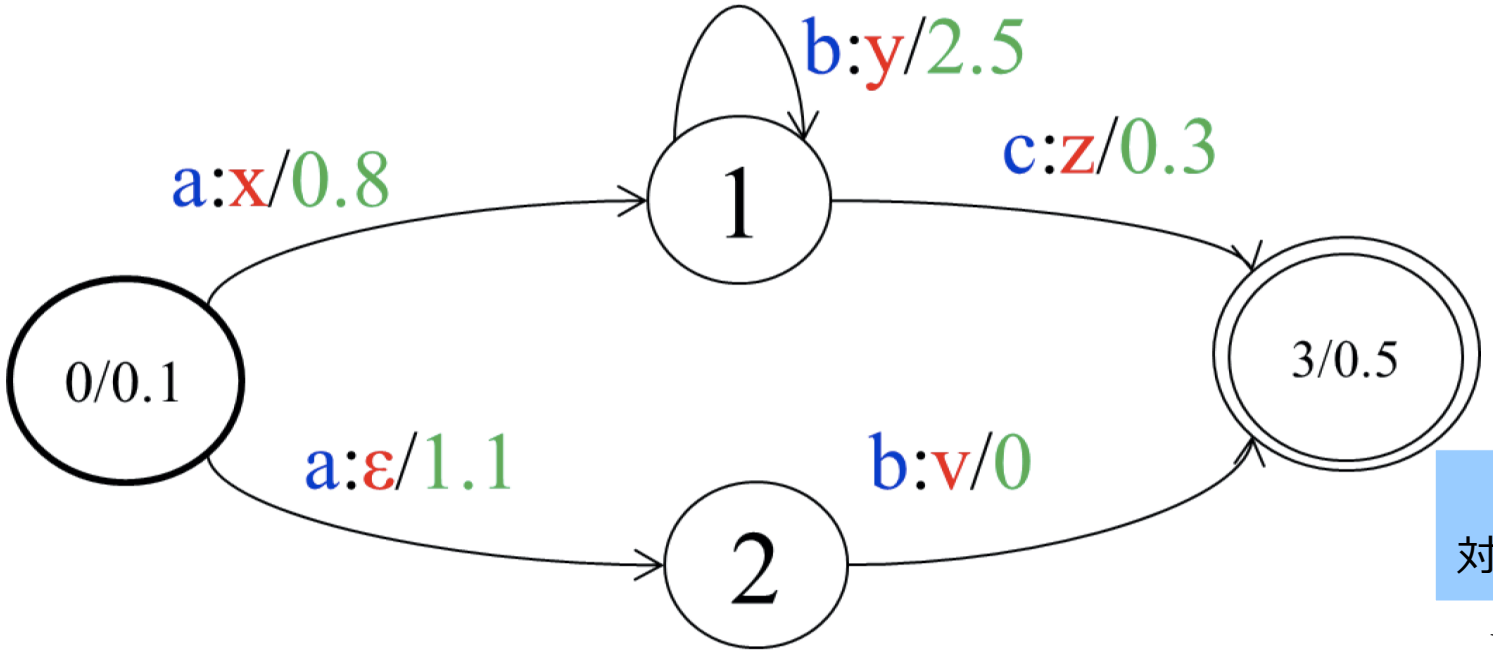
14.2 ヒューリスティック探索

- ヒューリスティック探索とは
  - 各候補の**今後の**スコアを予測し、高い順に探索



14.3 WFST による探索手法

- WFST とは
  - Weighted Finite State Transducer （重み付き有限状態トランスデューサ）
  - 記号列を入力し、別の記号列と重みを出力



入力 ac に
対する出力は？

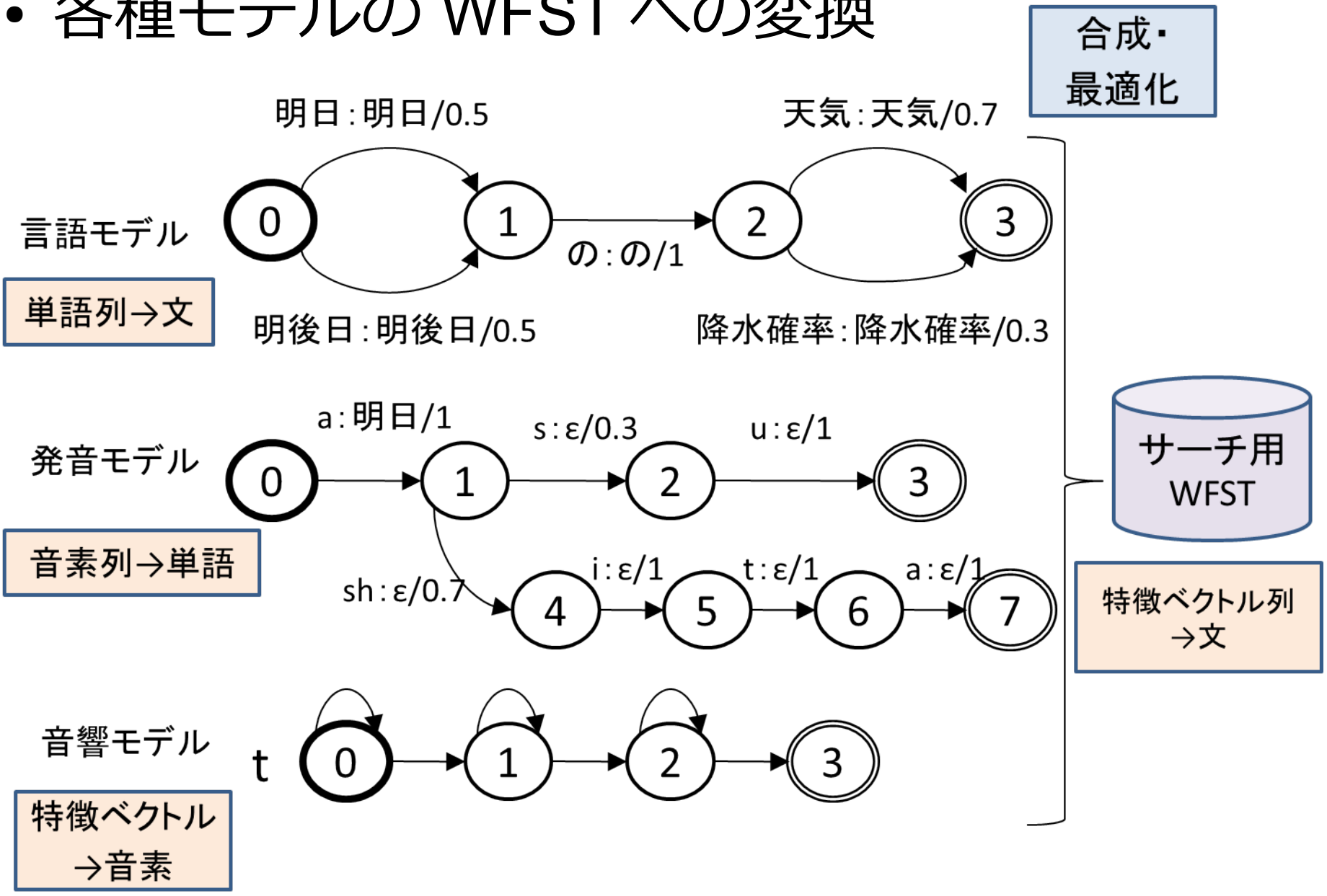
xz+0.0012

14.3 WFST による探索手法

- WFST によるデコードのアイディア
  - 音声認識に用いる確率モデル（ HMM 、単語辞書、言語モデルなど）は WFST で表現可能
  - 記号列 A を記号列 B に変換する WFST1 と、記号列 B を記号列 C に変換する WFST2 を合成すると、記号列 A を記号列 C に変換する WFST になる
    - ただし、状態数は組み合わせ的に増える
  - WFST には、 FSA と同様、決定化・最小化のアルゴリズムが存在する

14.3 WFST による探索手法

- 各種モデルの WFST への変換



参考文献

講義用スライドも公開中
https://masahiroaraki.github.io/GuideToASR/