Vol.2013-SLP-97 No.9
2013/7/26

CSJを用いた日本語講演音声認識へのDNN-HMMの適用と話者適応の検討

三村 正人[1]　河原 達也[1]

概要：近年、深い階層構造を持つニューラルネットワーク（DNN）と HMM を組み合わせたハイブリッド型モデル (DNN-HMM) の有効性が種々の音声認識タスクで報告されている。本研究では、DNN-HMM を『日本語話し言葉コーパス』（CSJ）を用いて構築し、種々の日本語講演音声タスクで評価を行う。また、類似話者を用いたネットワークの再学習による適応手法を提案する。DNN-HMM は CSJ の評価セットに対して、従来の GMM-HMM より 2.5 %高い認識精度を示した。また、階層を増やす毎に認識精度が向上した。CSJ と音響条件や話題の異なるシンポジウムの講演音声に対しても、すべての話者で認識精度が向上し、平均で 5.4 %の向上があった。話者適応実験では、類似話者を用いた手法により 2 %精度が向上した。さらに、認識対象音声の初期認識結果と類似話者の両者を用いた再学習により 3.4 %精度が向上した。

1. はじめに

ニューラルネットワークによるパターン認識は長い歴史を持つが、近年、深い構造を持つニューラルネットワーク (DNN: Deep Neural Network) に対する有効な事前学習法 [1] が確立されたことと、計算資源および学習データ量の増加を背景として、あらためて注目を集めている。音声認識においても、HMM との組み合わせにより、多くのタスクで従来の GMM-HMM よりもはるかに高い認識精度を示すことが報告されている [2][3][4][5]。

DNN を HMM と組み合わせる方法としては、二つのアプローチが考えられる。一つは、ネットワークにより計算した状態事後確率を（無相関化、次元削減ののち）通常の GMM-HMM の入力特徴量とするタンデム型アプローチ [6][7][8] である。もう一つは、HMM の状態確率を DNN により直接計算するハイブリッド型アプローチ (DNN-HMM) である [3][4][5]。

本研究では、大規模な『日本語話し言葉コーパス』（CSJ）を用いて DNN-HMM を学習し、日本語講演音声において評価を行う。評価データには、CSJ の評価セットおよび CSJ とは音響条件や話題の異なるシンポジウムの講演音声を用いる。

未知の話者や環境のデータに対してはモデルパラメータの適応が有効であるが、DNN-HMM では、MLLR や MAP などの適応手法が利用できないという問題がある。本研究では、類似話者のデータを用いたネットワークの再学習に基づく話者適応の手法を提案し、シンポジウムの講演音声により評価を行う。

[1] 京都大学 学術情報メディアセンター
Academic Center for Computing and Media Studies, Kyoto University

2. DNN-HMM

DNN-HMM は、従来の GMM-HMM の状態確率計算を DNN の出力する状態事後確率に基づいて行うハイブリッド型モデルである。したがって、DNN の出力層のノードは、HMM 状態と対応させる必要がある。音素コンテクスト依存モデル（トライフォン）を用いる場合は、DNN の出力層の各ノードを個々の共有状態と一致させる。状態の共有構造や状態遷移確率は、学習済みの GMM-HMM のパラメータをそのまま用いる。

DNN-HMM では、非線形な活性化関数を持つ隠れ層の乗算的な効果（”product of experts”）によりはるかに複雑な識別面を学習することが可能である。また、複数フレームの特徴量をまとめて入力にできる利点もある。

2.1 DNN

DNN-HMM に組み込むためのニューラルネットワークは、HMM の他のコンポーネントと独立に学習するのが一般的である。DNN の学習は、適切な初期値を得るための教師なし事前学習 (pre-training) と、教師あり学習 (fine-tuning) の二つのステップにより行う。DNN の学習の流れを図 1 に示す。

まず、RBM(Restricted Boltzmann Machine) を一層ずつ独立に学習する。次に、これらの RBM を積み重ねて DBM(Deep belief network) を作成する。さらに、乱数によって初期化したソフトマックス層を追加することで、DNN の初期ネットワークを構成する。最後に、フレーム毎の正解ラベル（状態 ID）を用いて、誤差逆伝播法（バックプロパゲーション:BP）によ

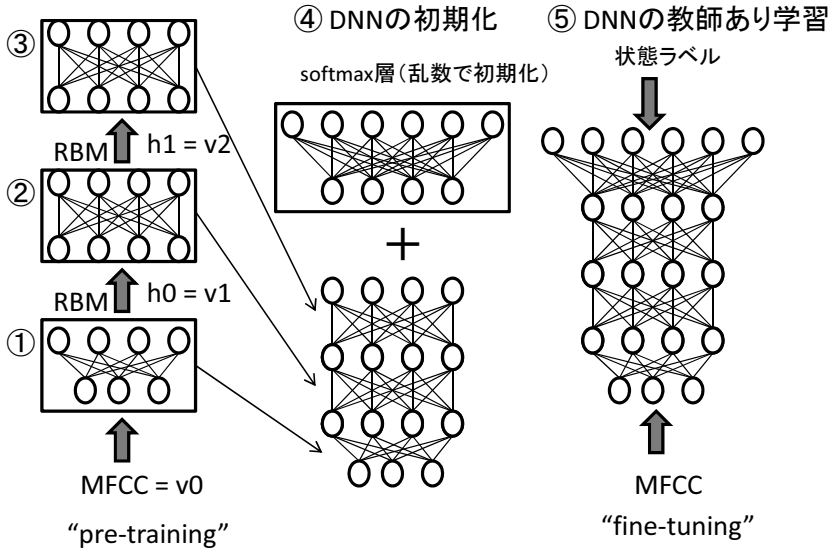

図 1 DNN の学習の概要

る教師あり学習を行う。

2.2 RBM による事前学習

DNN を乱数による初期値から誤差逆伝播法によって学習しようとすると、1 より小さな値の乗算の繰り返しにより特に浅い層のパラメータがほとんど更新されなくなり、適切な解が得られないことが多い。そこで、教師ラベルを用いた識別的な学習を行う前に、教師なし生成学習によりネットワークの適切な初期値を設定する [1]。

パターン認識において最も一般的には、事前学習に RBM を用いる。RBM は双方向型モデルであり、次式により可視ノード $\mathbf{v}$ と隠れノード $\mathbf{h}$ の同時確率を与える。

$$E(\mathbf{v},\mathbf{h}) = -\mathbf{b}^T\mathbf{v} - \mathbf{ch} - \mathbf{v}^T\mathbf{Wh} \tag{1}$$

$$P(\mathbf{v},\mathbf{h}) = \frac{\mathrm{e}^{-E(\mathbf{v},\mathbf{h})}}{\sum_{\mathbf{v},\mathbf{h}} \mathrm{e}^{-E(\mathbf{v},\mathbf{h})}} \tag{2}$$

$\mathbf{v}$ が与えられたときの $\mathbf{h}$ の活性化確率および $\mathbf{h}$ が与えられたときの $\mathbf{v}$ の活性化確率は次式で与えられる。

$$P(\mathbf{v}=\mathbf{1}|\mathbf{h}) = \sigma(\mathbf{b}+\mathbf{h}^T\mathbf{W}^T) \tag{3}$$

$$P(\mathbf{h}=\mathbf{1}|\mathbf{v}) = \sigma(\mathbf{c}+\mathbf{v}^T\mathbf{W}) \tag{4}$$

ここで、$\sigma()$ はシグモイド関数であり、多層パーセプトロンの隠れ層の活性化関数と一致する。

学習データの尤度 $P(\mathbf{v})$ をモデルパラメータ θ で偏微分すると、次式が得られる。

$$\begin{aligned} -\frac{\partial \log P(\mathbf{v})}{\partial\theta} &= -\frac{\partial \log \sum_{\mathbf{h}} \mathrm{e}^{-E(\mathbf{v},\mathbf{h})}}{\partial\theta} + \frac{\partial \log \sum_{\mathbf{v},\mathbf{h}} \mathrm{e}^{-E(\mathbf{v},\mathbf{h})}}{\partial\theta} \\ &= \sum_{\mathbf{h}} P(\mathbf{h}|\mathbf{v})\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial\theta} - \sum_{\mathbf{v},\mathbf{h}} P(\mathbf{h},\mathbf{v})\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial\theta} \\ &= \langle\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial\theta}\rangle_{data} - \langle\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial\theta}\rangle_{model} \end{aligned} \tag{5}$$

モデルパラメータをこの微分の向きに更新することにより、学習データの尤度が向上する。すなわち、RBM の学習は、$\langle\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial\theta}\rangle_{data}$ と $\langle\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial\theta}\rangle_{model}$ の二つの項の計算に帰着できる。

第一項 $\langle\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial\theta}\rangle_{data}$ は、$\mathbf{v}$ の条件付きの $\mathbf{h}$ の期待値であり、次式で計算できる。

$$\langle\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial w_{i,j}}\rangle_{data} = \langle v_i h_j\rangle_{data} = \mathbf{v}_{0i}\cdot\mathbf{h}_{0j} \tag{6}$$

$$\langle\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial b_i}\rangle_{data} = \langle v_i\rangle_{data} = \mathbf{v}_{0i} \tag{7}$$

$$\langle\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial c_j}\rangle_{data} = \langle h_j\rangle_{data} = \mathbf{h}_{0j} \tag{8}$$

ここで $\mathbf{v}_0$ は入力ベクトルそのものであり、$\mathbf{h}_0$ は $\mathbf{v}_0$ の条件付きの $\mathbf{h}$ のサンプルである。ただし、実際の計算では、サンプリングの結果（2 値）ではなく、活性化確率（式 (4)）そのものを用いる。

第二項は、モデルについての期待値であり、ギブスサンプリングを用いて計算できる。すなわち、図 2 のように $\mathbf{h}$ と $\mathbf{v}$ を (式 (3)、(4) を用いて) 交互にサンプリングすることにより期待値を得る。まず、$\mathbf{v}_0$ の条件付きで $\mathbf{h}_0$ をサンプリングする。次にこの $\mathbf{h}_0$ を用いて $\mathbf{v}_1$ をサンプリングし、さらに $\mathbf{v}_1$ を用いて $\mathbf{h}_1$ をサンプリングする。実用上この回数が一回でも十分有効な近似が得られる [9]。したがって、結局、第二項は以下で計算できる。

$$\langle\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial w_{i,j}}\rangle_{model} \simeq \langle v_i h_j\rangle_1 = \mathbf{v}_{1i}\cdot\mathbf{h}_{1j} \tag{9}$$

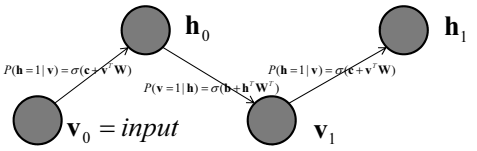

図 2 RBM のギブスサンプリング

$$\langle\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial b_i}\rangle_{model} \simeq \langle v_i\rangle_1 = \mathbf{v}_{1i} \tag{10}$$

$$\langle\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial c_j}\rangle_{model} \simeq \langle h_j\rangle_1 = \mathbf{h}_{1j} \tag{11}$$

パラメータの学習は、確率的勾配降下法 (SGD) を用いて行う。微分は式 (5) に従ってサンプル毎に計算する。ただし、計算量の削減と、微分値のスムージングのために、ミニバッチと呼ばれる小さなまとまりで微分値を平均する。パラメータの更新もこのミニバッチ毎に行う。さらに、次式により、前回の更新値とのスムージングを行う。

$$\Delta\theta(t+1) = m\Delta\theta(t) + \alpha\frac{\partial \log P(\mathbf{v})}{\partial\theta} \tag{12}$$

前回の更新値には 1 より小さい係数 m（モメンタム）をかけて用いる。α はラーニングレートである。

全学習データを複数回数（エポック）用いて一層の学習が終わると、この層の隠れノードの値を、次層の可視ノードの値として用いる。その際、サンプリングにより 2 値のいずれかに決定するのでなく、活性化確率をそのまま用いると、サンプリングノイズの軽減の意味で有効である [10]。

なお、音声認識では入力は実数値であるため、第一層には Gaussian-Bernoulli RBM を用いる。GRBM では、エネルギー関数が

$$E(\mathbf{v},\mathbf{h}) = \frac{1}{2}(\mathbf{v}-\mathbf{b})^2 - \mathbf{ch} - \mathbf{v}^T\mathbf{Wh} \tag{13}$$

で与えられる。したがって、$\mathbf{h}$ の条件付きの $\mathbf{v}$ の確率は、

$$P(\mathbf{v}|\mathbf{h}) = N(\mathbf{v};\mathbf{b}+\mathbf{h}^T\mathbf{W}^T,\mathbf{I}) \tag{14}$$

となり、多次元正規分布となる。$\mathbf{v}$ の条件付きの $\mathbf{h}$ の確率は式 (4) と同じであり、学習アルゴリズムは (6)〜(11) 式をそのまま利用できる。

2.3 DNN の教師あり学習

DNN の学習は、RBM により得られた初期ネットワークに対して、誤差逆伝播法 (BP)[11] を適用することで行う。損失関数には、以下の相対エントロピーを用いる。

$$L = \sum_i y_i \log x_i^{output} \tag{15}$$

$\mathbf{y}$ は教師信号で、出力層ノードと同数の次元を持ち、正解状態に対応する次元のみが 1 で、残りが 0 のベクトルである。

表 1 GPGPU による高速化

	CPU	GPU
RBM 第 1 層 (２０エポック)	130 h	1.3 h
RBM 第 2 層 (１０エポック)	290 h	2.3 h
RBM 第 3 層 (１０エポック)	330 h	2.5 h
DNN ３隠れ層 (２０エポック)	880 h	8.3 h
計	67 days	0.6 days

BP による学習は、まず次の前向き処理により各ノードの activation を計算する ($l = 1...L$)。

$$\mathbf{a}^l = \mathbf{W}^l\mathbf{x}^{l-1} + \mathbf{c}^l \tag{16}$$

$$\mathbf{x}^l = \sigma(\mathbf{a}^l) \tag{17}$$

ただし、出力層のみ以下のソフトマックス関数を用いる。

$$\mathbf{a}^{output} = \mathbf{W}^{output}\mathbf{x}^L + \mathbf{c}^{output} \tag{18}$$

$$\mathbf{x}_i^{output} = \frac{\exp(a_i^{output})}{\sum_j \exp(a_j^{output})} \tag{19}$$

次に、後ろ向き処理によりモデルパラメータの微分を再帰的に計算していく ($l = L...1$、ただし $L+1 = output$)。

$$\delta_j^{output} = \frac{\partial L}{\partial a_j^{output}} = y_j - x_j^{output} \tag{20}$$

$$\frac{\partial L}{\partial w_{ji}^{output}} = x_i^L\delta_j^{output}, \frac{\partial L}{\partial c_i^{output}} = \delta_i^{output} \tag{21}$$

$$\delta_j^l = \frac{\partial L}{\partial a_j^l} = \sum_k w_{kj}^{l+1}\delta_k^{l+1} \tag{22}$$

$$\frac{\partial L}{\partial w_{ji}^l} = x_i^{l-1}\delta_j^l, \frac{\partial L}{\partial c_i^l} = \delta_i^l \tag{23}$$

DNN の BP も、ミニバッチによる確率的勾配降下法を用い、過去の更新値とのスムージングを行う。

2.4 GPU による高速化

RBM の教師なし事前学習および DNN の教師あり学習は、各層のノードの値をミニバッチ分まとめた行列の行列演算に帰着できる。しかし、通常の CPU では、(GMM-HMM のような並列化がしにくいこともあり) 膨大な計算時間を必要とする。そこで、この行列計算を GPGPU で行うことにより、計算時間を大幅に削減できる。本研究では、トロント大学で作成された Python 用のライブラリである CUDAMat[12] を用いる。

CSJ の学会講演の一部（女性のみ、40 時間）を用いた予備実験の CPU および GPGPU の計算時間を表 1 に示す。CPU は Intel Xeon X5670(2.93GHz)、GPGPU は NVIDIA Tesla K20 を用いた。GPGPU により、総学習時間を 1/100 以下に抑えることができる。

2.5 DNN-HMM を用いたデコーディング

DNN の出力は状態の事後確率であるため、従来の音声認識

デコーディングアルゴリズムに組み込むためには、以下のベイズ則を用いてスケーリングを行う。

$$p(\mathbf{x}_t|q_t) \propto \frac{p(q_t|\mathbf{x}_t)}{p(q_t)} \tag{24}$$

ここで、$p(q_t)$ は各状態の事前確率であり、学習データ中の状態の出現回数を総フレーム数で割った値を用いる。

DNN-HMM と GMM-HMM の違いは状態の確率計算のみなので、既存の GMM-HMM 用デコーダを用いて DNN-HMM の認識を行うことができる。この際、MFCC のような生の特徴量の代わりに、事前に式 (24) で計算した状態確率のベクトルを入力とする。一方、デコーダ側は、GMM の確率計算を行う代わりに、入力ベクトル中の確率値をそのまま返すように改変しておく。

3. DNN-HMM の話者適応

未知の話者・環境のデータに対して高精度な認識を行うためには、モデルパラメータの適応が必須である。しかし、DNN-HMM では、GMM-HMM で確立された MLLR や MAP などの適応手法を利用することができない。また、VTLN や CMLLR などの特徴量領域の適応手法は、ネットワークが深い階層を持つ場合、あまり有効でないことが報告されている [13]。

一方、適応データを用いてモデルの再学習を行うことで、簡易に適応の効果が得られる [14]。ただし、BP による教師あり学習はラベルの品質の影響を大きく受けるため、適応話者について正解つきデータが利用できるか、すでに十分高い精度の認識結果が存在することが前提となる。

これに対して、正則化や分布間距離の制約、ドロップアウト学習などを組み合わせることで、モデルパラメータの更新幅を制限する手法が提案されている [15][16]。

本研究では、学習データベース中の類似話者のデータでネットワークの追加学習を行う手法を提案する。学習データについては正解ラベルが存在するため、ラベル精度の問題はない。類似話者の選択には、いくつかの基準が考えられるが、本研究では適応・評価話者と学習データ中の話者の MLLR 変換行列の類似度により行う。

この類似度は、MLLR のアフィン変換を行列 $\mathbf{A}_s$ およびバイアス $\mathbf{b}_s$ としたとき、

$$\mu_s = \mathbf{A}_s\mu + \mathbf{b}_s \tag{25}$$

と変換済みの平均ベクトルの距離で評価する。μ は適応元 GMM-HMM の平均ベクトルを並べた高次元のスーパーベクトルである。適応後の平均ベクトル μ_s のベクトル間の距離が小さい複数の話者を類似話者として選択する。ただし、簡易にアフィン変換 ($\mathbf{A}_s$ と $\mathbf{b}_s$) の要素の差の自乗和を用いても類似度は評価できる。こうして選択した複数の類似話者の学習データを用いて、ネットワークの教師つき追加学習を行う。追加学習では、ラーニングレートは 0.001 と比較的小さな値を用いてパラメータの更新量を制限する。

表 2 学習パラメータ

	パラメータ	値
RBM	初期値	分散 0.01 の正規乱数
	ラーニングレート	0.004
	モメンタム	0.9
	ミニバッチサイズ	256
	エポック数	10
DNN	出力層の初期値	分散 0.0001 の正規乱数
	ラーニングレート	0.006(エポック毎の半減)
	モメンタム	0.9
	ミニバッチサイズ	256
	エポック数	20

表 3 CSJ 評価セットの認識精度

	単語認識精度
GMM-HMM ML	76.88
GMM-HMM MPE	79.97
DMM-HMM 隠れ層 1	80.01
DMM-HMM 隠れ層 2	81.03
DMM-HMM 隠れ層 3	81.37
DMM-HMM 隠れ層 4	81.92
DMM-HMM 隠れ層 5	82.50

4. 評価実験

CSJ の学会講演（967 講演、257 時間）を用いて、コンテクスト依存（トライフォン）DNN-HMM の学習を行った。

比較用の GMM-HMM を同じデータで学習した。GMM-HMM の特徴量は、MFCC12 次元、ΔMFCC、ΔΔMFCC、Δパワー、ΔΔ パワーの計 38 次元であり、CMN、CVN、VTLN により正規化した。学習は MPE 基準により行った。状態数は 3015 であり、状態毎のガウス分布数は 16 である。

DNN の入力は、上記の 38 次元特徴量を前後 5 フレーム、あわせて 11 フレーム分まとめた計 418 次元とした。出力層で識別する状態は、GMM-HMM の 3015 状態を用いる。DNN の教師あり学習に用いる状態ラベルは、GMM-HMM による強制アライメントにより作成した。なお、SGD を効果的に行うために、学習に用いるサンプルはフレーム単位でシャッフルしておく。RBM・DNN の学習用プログラムは Python+Numpy で実装した。また、行列演算部分については 2.4 節で述べた CUDAMat クラスを用いて高速化した。

DNN の学習における各種パラメータを表 2 に示す。

デコーダには、Julius-4.2 を一部改変したものを用いた。ビーム幅などのサーチパラメータは、GMM-HMM と DNN-HMM で同じ値を用いた。ただし、言語重みおよび挿入ペナルティは各モデルで最適化した。GMM-HMM の言語重みは 11、挿入ペナルティは-5.0、DNN-HMM は 12 および-1.0 となった。DNN-HMM では挿入ペナルティが小さくなる傾向があった。

表 4 京都大学 OCW CiRA シンポジウム講演の認識精度

ID	2009_prof_N	2009_prof_T	2009_prof_Y	2010_prof_O	2010_prof_T	2010_prof_Y	計
時間 (分)	19.9	18.8	22.2	19.3	26.4	33.2	139.8
GMM	59.33	73.93	50.53	77.49	78.78	81.91	71.85
DNN	71.69	80.88	55.28	81.22	83.19	83.90	77.12

表 5 京都大学 OCW 震災シンポジウム講演の認識精度

ID	4-4	4-7	7-3	8-2	8-4	8-5	10-7	13-1	16-2	17-8	計
時間 (分)	13.2	8.5	9.8	70.5	37.3	34.1	28.5	38.9	45.6	6.7	293.2
GMM	61.19	57.13	52.22	57.28	68.79	51.80	39.75	51.92	55.24	60.04	55.38
DNN	68.63	61.54	52.35	61.34	73.17	58.62	45.49	59.39	61.53	69.10	60.85

4.1 CSJ の結果

CSJ の評価セット（男性 10 講演、CSJ-TESTSET1）により認識実験を行った。言語モデルは、CSJ の学会講演および模擬講演の書き起こしから作成した 3-gram を用いた。

GMM-HMM(ML 基準、MPE 基準) および DNN-HMM の隠れ層の数を 1〜5 に変えて構成した場合の単語認識精度を表 3 に示す。

隠れ層が 1 層のみの DNN-HMM でも、MPE 基準で学習した GMM-HMM と同等の認識精度となった。また、隠れ層を増やす毎に認識精度が向上し、隠れ層が 5 のモデルでは、GMM-HMM よりも 2.5 %高い認識精度となった。

4.2 京都大学 OCW CiRA シンポジウムの結果

CSJ と異なる音響条件・話題のデータに対する頑健さを評価するために、京都大学 OCW で公開されている CiRA シンポジウム (http://ocw.kyoto-u.ac.jp/ja/center-for-ips-cell-research-and-application-jp) の講演音声を用いて評価実験を行った。評価データには、6 名・6 講演を用いた。

言語モデルは、本タスクに適応したモデルを用いた [17]。CiRA シンポジウムの 6 講演に対する文字認識精度を表 4 に示す。

DNN-HMM の認識精度はすべての講演で GMM-HMM を上回り、平均で 5.3 %の向上があった。

4.3 京都大学 OCW 震災シンポジウム講演の結果

京都大学 OCW で公開されている別の講演である「大震災後を考える」シンポジウムシリーズ (http://ocw.kyoto-u.ac.jp/ja/opencourse/16) の講演音声を用いて認識実験を行った。評価データには、10 名・10 講演を用いた。

言語モデルは、CSJ のモデルと新聞記事から学習したモデルを線形補間することで作成した。震災シンポジウムの 10 講演に対する文字認識精度を表 5 に示す。

DNN-HMM の認識精度はすべての講演で GMM-HMM を上回り、平均で 5.4 %の向上があった。

4.4 話者適応の効果

ネットワークの追加学習による適応の効果について、特に平均の精度が低い震災シンポジウムのデータで評価した。適応手法としては、以下を比較した。

(1) 評価データの初期認識結果を用いてネットワークを再学習
(2) 評価データの類似話者のデータを用いてネットワークを再学習
(3) (1) と (2) の併用

類似話者は、MLLR により変換した平均ベクトルの距離を基準に 10 名を選択した。ただし、変換行列自体の距離（要素の差の自乗和）を用いても選択される話者は同じであった。

震災シンポジウムの 10 講演に対する教師なし適応実験の結果を表 6 に示す。

評価データの初期認識結果を用いた手法では、適応なしのモデルより 1.8 %精度が向上した。平均 60.85 %と品質の低いラベルを用いても、追加学習による適応に一定の効果があることがわかる。

類似話者を用いた手法では、2.0 %精度が向上し、適応データを用いた場合よりやや高い精度となった。

この二つの手法を話者・講演別に比較すると、おおむね類似話者による手法の方が精度が高かった。しかし、初期認識結果の精度が比較的高い話者については、認識結果を用いた手法の方が精度が高い傾向があった。したがって今後、認識信頼度を用いて追加学習に用いるデータの選別を行うことを検討する。

また、評価データの初期認識結果と類似話者の両方を用いた手法では、平均 3.4 %精度が向上し、組み合わせの効果が見られた。

5. おわりに

DNN-HMM を大規模な『日本語の話し言葉コーパス』(CSJ) を用いて学習し、評価を行った。DNN-HMM の認識精度は、CSJ 評価セットに対して GMM-HMM を 2.5 %上回った。また、CSJ と異なるシンポジウムの講演音声についても評価を行い CiRA シンポジウムで平均 5.3 %、震災シンポジウムで平均 5.4 %精度が向上した。震災シンポジウムでは、類似話者を用いた適応により、さらに 3.4 %精度が向上した。このように、

表 6 京都大学 OCW 震災シンポジウム講演の認識精度 (教師なし適応)

ID	4-4	4-7	7-3	8-2	8-4	8-5	10-7	13-1	16-2	17-8	計
時間 (分)	13.2	8.5	9.8	70.5	37.3	34.1	28.5	38.9	45.6	6.7	293.2
適応なし	68.63	61.54	52.35	61.34	73.17	58.62	45.49	59.39	61.53	69.10	60.85
(1) 教師なし適応（認識結果）	71.46	64.93	52.23	62.94	74.19	59.95	48.20	61.16	63.90	70.66	62.67
(2) 教師なし適応（類似話者）	71.73	65.79	53.59	63.34	72.86	60.78	48.64	62.49	63.43	69.26	62.89
(3) 両方	73.19	66.13	54.94	65.00	74.81	62.28	49.34	64.11	64.45	69.41	64.24

DNN-HMM は精度面で GMM-HMM より有意に高い性能を示した。

また、同じサーチパラメータのもとで、GMM-HMM が実時間の 4.5 倍の認識時間を要したのに対し、DNN-HMM では 1.7 倍しかかからず、速度面でも優位となった。学習時も、ラティスを用いる GMM-HMM の識別学習より手続きが単純である。GPU を用いると学習時間も並列化した GMM-HMM の学習と同等程度に抑えることができる。

今後は、話者適応に関してさらに検討を行う予定である。

参考文献

[1] G.E.Hinton, S.Osindero and Y.Teh: A fast learning algorithm for deep belief nets, *Neural Computation*, Vol. 18, pp. 1527–1554 (2006).

[2] G.E.Hinton, L.Deng, D.Yu, G.Dahl, A.Mohamed, N.Jaitly, A.Senior, V.Vanhoucke, P.Nguyen, T.Sainath and B.Kingsbury: Deep neural networks for acoustic modeling in speech recognition, *IEEE Signal Processing Magazine*, Vol. 29, No. 6, pp. 82–97 (2012).

[3] A.Mohamed, G.Dahl and G.Hinton: Acoustic modelling using deep belief networks, *IEEE Trans. Audio, Speech, & Language Proc.*, Vol. 20, No. 1, pp. 14–22 (2012).

[4] G.E.Dahl, D.Yu, L.Deng and A.Acero: Context-dependent pre-trained deep neural networks for large vocabulary speech recognition, *IEEE Trans. Audio, Speech, & Language Proc.*, Vol. 20, No. 1, pp. 30–42 (2012).

[5] F.Seide, G.Li and D.Yu: Conversational speech transcription using context-dependent deep neural networks, *Proc. Interspeech*, pp. 437–440 (2011).

[6] N.Morgan: Deep and wide: Multiple layers in automatic speech recognition, *IEEE Trans. Audio, Speech, & Language Proc.*, Vol. 20, No. 1, pp. 7–13 (2012).

[7] G.S.V.S.Sivaram and H.Hermansky: Sparse multilayer perceptron for phoneme recognition, *IEEE Trans. Audio, Speech, & Language Proc.*, Vol. 20, No. 1, pp. 23–29 (2012).

[8] P.J.Bell, M.J.F.Gales, P.Lanchantin, X.Liu, Y.Long, S.Renals, P.Swietojanski and P.C.Woodland: Transcriptions of multi-genre media archives using out-of-domain data, *Proc. SLT*, pp. 324–329 (2012).

[9] G.E.Hinton: Training products of experts by minimizing contrastive divergence, *Neural Computation*, Vol. 14, pp. 1711–1800 (2002).

[10] G.E.Hinton: A Practical Guide to Training Restricted Boltsmann Machines, http://www.cs.toronto.edu/~hinton/absps/guideTR.pdf (2012).

[11] D.E.Rumelhart, G.E.Hinton and R.J.Williams: Learning representations by back-propagating errors, *Nature*, Vol. 323, No. 6088, pp. 533–536 (1986).

[12] V.Mnih: Cudamat: a CUDA-based matrix class for python, *Department of ComputerScience, University of Toronto, Tech. Rep. UTML TR* (2009).

[13] S.Seide, G.Li and D.Yu: Feature engineering in context-dependent deep neural networks for conversational speech transcription, *Proc. ASRU*, pp. 24–29 (2011).

[14] Y.Xiao, Z.Zhang, S.Cai, J.Pan and Y.Yan: A initial attempt on task-specific adaptation for deep neural network besed large vocabulary continuous speech recognition, *Proc. Interspeech* (2012).

[15] H.Liao: Speaker adaptation of context dependent deep neural networks, *Proc. ICASSP*, Vol. 1, pp. 7947–7951 (2013).

[16] D.Yu, K.Yao, H.Su, G.Li and F.Seide: KL-Divergence regularized deep neural network adaptation for improved large vocabulary speech recognition, *Proc. ICASSP*, Vol. 1, pp. 7893–7897 (2013).

[17] 秋田祐哉，河原達也：オープンコースウェアの講演を対象とした音声認識に基づく字幕付与，音講論 (2013 春)，Vol. 2-9-9 (2006).