BOTTLENECK LINEAR TRANSFORMATION NETWORK ADAPTATION FOR SPEAKER ADAPTIVE TRAINING-BASED HYBRID DNN-HMM SPEECH RECOGNIZER

Tsubasa Ochiai[1,2]*, Shigeki Matsuda*[2]*, Hideyuki Watanabe*[1]*, Xugang Lu*[1]*,*
Hisashi Kawai[1]*, and Shigeru Katagiri*[2]

[1] National Institute of Information and Communications Technology, Kyoto, Japan
[2] Graduate School of Engineering, Doshisha University, Kyoto, Japan

eup1105@mail4.doshisha.ac.jp, {shmatsud,skatagir}@mail.doshisha.ac.jp,
{hideyuki.watanabe,xugang.lu,hisashi.kawai}@nict.go.jp

ABSTRACT

Recently, a Hybrid DNN-HMM recognizer trained with the Speaker Adaptive Training (SAT) concept was successfully modified to a more effective speaker-adaptation-oriented recognizer whose DNN front-end adopted a Linear Transformation Network (LTN) Speaker Dependent (SD) module. However, the size of SD modules is still large, which incurs high storage costs and the risk of over-training. To alleviate this problem, we analyze the characteristics of an LTN module by focusing on the relation between its size and its feature-representation capability. Moreover, we propose a new SAT-based scheme for reducing the LTN size using SVD-based matrix compression. Evaluation experiments on the TED Talks corpus prove that our LTN size-reduction scheme not only maintains the adaptation performance of the original LTN-embedded, SAT-based DNN-HMM recognizer but also further increases it especially in cases where the speech data available for adaptation training are severely limited.

Index Terms— Speaker Adaptive Training, Deep Neural Network, Linear Transformation Network

1. INTRODUCTION

Speaker adaptation is one of the key technologies needed to achieve high-performance speech recognition. Its various embodiments have been vigorously investigated for the increasingly popular Hybrid DNN-HMM speech recognizers [1–17]. Among those embodiments, our method [1] demonstrated the effectiveness of the idea of embedding speaker-dependent (SD) modules into a large-scale DNN front-end of a Hybrid DNN-HMM recognizer and then training the DNN part based on the Speaker Adaptive Training (SAT) concept [18]. In addition, by adopting a Linear Transformation Network (LTN) [19–21], the method was successfully upgraded to a SAT-based Hybrid DNN-HMM recognizer whose SD module was defined with LTN (called SAT-DNN-LTN in this paper) [2].

In real-world situations where the amount of available data is finite, large-scale networks easily suffer from the over-training problem. Trained networks generally work for the data in hand but often fail for unseen data. This problematic phenomenon becomes more serious in such cases where only a severely limited amount of data is available in the speaker adaptation stage. In our previous methods, such as SAT-DNN-LTN [2], this problem was effectively controlled using the regularization concept. Nevertheless, the size of the used SD module is still large, and its reduction is obviously desirable because a small SD module further decreases the risk of over-training and reduces computational load as well as storage cost. Motivated by this understanding, we propose in this paper a new technique for reducing the size of the DNN front-end, or more precisely the size of the LTN SD module, in our latest SAT-DNN-LTN method without degradation of its classification power. Note that an LTN SD module is prepared for every training/target speaker and thus its computational and storage costs are high; furthermore, the size of the SD module should be strictly constrained due to the limited amount of speech data used in the speaker adaptation stage.

Several techniques for alleviating over-training by reducing the adaptable parameter size of LTN have been investigated, focusing on the sharing or restriction of the adaptable parameters [3–5]. In addition, Singular Value Decomposition (SVD)-based low-rank matrix conversion has also been investigated to reduce the number of adaptation parameters [6–8]. However, the best choice among these techniques and, more importantly, the relation between the size of adaptable parameters and their feature-representation capability in the LTN have not yet been clarified. Consequently, in this paper, we first theoretically analyze this relation and next propose, based on analysis results, a new SAT-based scheme for reducing the size of LTN while maintaining its feature-representation power. Evaluation experiments prove that our LTN size-reduction scheme not only maintains the adaptation performance of the original SAT-DNN-LTN recognizer but also further improves it even in cases where the speech data for adaptation are severely restricted.

2. PREVIOUS WORK: SAT-BASED HYBRID DNN-HMM RECOGNIZER EMBEDDING LTN SPEAKER-DEPENDENT MODULES

2.1. Overview

The SAT-DNN-LTN speech recognizer, which is a baseline for our development of an LTN size-reduction scheme in this paper, consists of the front-end DNN and the back-end HMM, and it is characterized by the LTN SD module that is switched for every speaker in the DNN part [2]. The back-end HMM is first designed using the GMM probability outputs estimated in the discriminative training using the mutual information loss, and it is used to produce the *senone* state allocation information for the SAT-based DNN optimization that is conducted under the cross entropy (CE) minimization criterion. After this SAT-based optimization of the whole DNN part, only the LTN SD module is replaced for a new target speaker and it is adapted using his/her speech data in the adaptation stage. In the successive paragraphs, we review the training procedures for the DNN part. The details of the remainder are shown in the literature [2].

Figure 1 illustrates the structure of the DNN part of SAT-DNN-LTN and the SAT-based training procedure. In the figure, we assume that our DNN has 7 layers $\{\mathrm{L}_l; l = 0, \ldots, 6\}$, and for simplicity, no biases are depicted. The LTN SD module can be inserted in any of the layers from L_1 to L_6. As an example, we illustrate the case where the LTN module is inserted in L_2 (the LTN layer is denoted by AL_2). The figure also schematically illustrates the SAT-based procedure

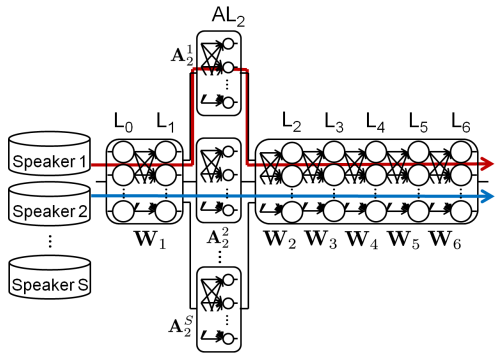

Fig. 1. DNN structure and SAT-based training procedure for SAT-DNN-LTN recognizer.

where SD modules $\{\mathbf{A}_2^s; s = 1, \cdots, S\}$ are switched along with the selection of training speaker, where $\mathbf{A}_2^s$ is the LTN matrix for training speaker s and S is the number of training speakers.

For explanation purposes, we denote the layer in which an LTN SD module is inserted by $\mathrm{L}_{l_{\mathrm{SD}}}$. Then, the computational procedure for the SD module layer is formulated as follows:

$$\begin{aligned}\mathbf{z}_{l_{\mathrm{SD}}} &= \sigma\Big(\mathbf{W}_{l_{\mathrm{SD}}}(\mathbf{A}_{l_{\mathrm{SD}}}\mathbf{z}_{l_{\mathrm{SD}}-1} + \mathbf{a}_{l_{\mathrm{SD}}}) + \mathbf{b}_{l_{\mathrm{SD}}}\Big) \\ &= \sigma\Big(\widehat{\mathbf{W}}_{l_{\mathrm{SD}}}\mathbf{z}_{l_{\mathrm{SD}}-1} + \widehat{\mathbf{b}}_{l_{\mathrm{SD}}}\Big), \end{aligned} \tag{1}$$

where $\mathbf{W}_{l_{\mathrm{SD}}}$ is the DNN's weight matrix in $\mathrm{L}_{l_{\mathrm{SD}}}$, $\mathbf{b}_{l_{\mathrm{SD}}}$ is the DNN's bias vector in $\mathrm{L}_{l_{\mathrm{SD}}}$, $\sigma(\cdot)$ is the activation function, $\mathbf{z}_{l_{\mathrm{SD}}}$ is the output from $\mathrm{L}_{l_{\mathrm{SD}}}$, $\mathbf{A}_{l_{\mathrm{SD}}}$ is the LTN's weight matrix in $\mathrm{L}_{l_{\mathrm{SD}}}$, $\mathbf{a}_{l_{\mathrm{SD}}}$ is the LTN's bias vector in $\mathrm{L}_{l_{\mathrm{SD}}}$, $\widehat{\mathbf{W}}_{l_{\mathrm{SD}}} = \mathbf{W}_{l_{\mathrm{SD}}}\mathbf{A}_{l_{\mathrm{SD}}}$, and $\widehat{\mathbf{b}}_{l_{\mathrm{SD}}} = \mathbf{W}_{l_{\mathrm{SD}}}\mathbf{a}_{l_{\mathrm{SD}}} + \mathbf{b}_{l_{\mathrm{SD}}}$.

2.2. Speaker Adaptive Training step

In the SAT procedure, we first initialize the DNN part by setting the network parameters of SI-DNN to $\{\mathbf{W}_l, \mathbf{b}_l\}(l = 1, \cdots, L)$, where L is the number of DNN layers excepting the input layer. Next, we insert the LTN SD module for training speaker s, of which the LTN weight matrix and bias vector are represented by $\mathbf{A}_{l_{\mathrm{SD}}}^s$ and $\mathbf{a}_{l_{\mathrm{SD}}}^s$, respectively, into $\mathrm{L}_{l_{\mathrm{SD}}}$ and train the LTN-inserted DNN in the CE minimization framework. We then repeat this SD module insertion and training along with changing training speakers from $s = 1$ to $s = S$ randomly. This SAT procedure is formulated as follows:

$$\begin{aligned}(\overline{\mathbf{\Lambda}}, \overline{\mathbf{A}}_{l_{\mathrm{SD}}}^{\mathrm{SDs}}, \overline{\mathbf{a}}_{l_{\mathrm{SD}}}^{\mathrm{SDs}}) = \\ \underset{(\mathbf{\Lambda}, \mathbf{A}_{l_{\mathrm{SD}}}^{\mathrm{SDs}}, \mathbf{a}_{l_{\mathrm{SD}}}^{\mathrm{SDs}})}{\arg\min}\ E(\mathbf{\Lambda}, \mathbf{A}_{l_{\mathrm{SD}}}^{\mathrm{SDs}}, \mathbf{a}_{l_{\mathrm{SD}}}^{\mathrm{SDs}}) + \frac{\beta}{2}R(\mathbf{A}_{l_{\mathrm{SD}}}^{\mathrm{SDs}}, \mathbf{a}_{l_{\mathrm{SD}}}^{\mathrm{SDs}}),\end{aligned} \tag{2}$$

where $\mathbf{\Lambda} = \{\mathbf{W}_1, \cdots, \mathbf{W}_L, \mathbf{b}_1, \cdots, \mathbf{b}_L\}$, $\mathbf{A}_{l_{\mathrm{SD}}}^{\mathrm{SDs}} = \{\mathbf{A}_{l_{\mathrm{SD}}}^1, \cdots, \mathbf{A}_{l_{\mathrm{SD}}}^S\}$, $\mathbf{a}_{l_{\mathrm{SD}}}^{\mathrm{SDs}} = \{\mathbf{a}_{l_{\mathrm{SD}}}^1, \cdots, \mathbf{a}_{l_{\mathrm{SD}}}^S\}$, E is the accumulated CE loss function, R is a regularization term based on L_2 norm [2], β is its regularization coefficient, and such overlined parameters as $\overline{\mathbf{\Lambda}}$ are the trained state of such corresponding parameter as $\mathbf{\Lambda}$.

2.3. Speaker adaptation step

In the speaker adaptation step, we first remove the SD modules for training speakers, i.e., $\mathbf{A}_{l_{\mathrm{SD}}}^{\mathrm{SDs}}$ and $\mathbf{a}_{l_{\mathrm{SD}}}^{\mathrm{SDs}}$. Then, inserting a new LTN SD module for target speaker t into $\mathrm{L}_{l_{\mathrm{SD}}}$, we train only the inserted SD module whose trainable parameters are $\mathbf{A}_{l_{\mathrm{SD}}}^t$ and $\mathbf{a}_{l_{\mathrm{SD}}}^t$, using the speech data of t. This adaptation procedure is formulated as follows:

$$\begin{aligned}(\overline{\mathbf{A}}_{l_{\mathrm{SD}}}^t, \overline{\mathbf{a}}_{l_{\mathrm{SD}}}^t) = \\ \underset{(\mathbf{A}_{l_{\mathrm{SD}}}^t, \mathbf{a}_{l_{\mathrm{SD}}}^t)}{\arg\min}\ E(\overline{\mathbf{\Lambda}}, \mathbf{A}_{l_{\mathrm{SD}}}^t, \mathbf{a}_{l_{\mathrm{SD}}}^t) + \frac{\beta}{2}R(\mathbf{A}_{l_{\mathrm{SD}}}^t, \mathbf{a}_{l_{\mathrm{SD}}}^t).\end{aligned} \tag{3}$$

Note that the trained DNN parameter $\overline{\mathbf{\Lambda}}$ is included in E but fixed in the minimization.

3. PROPOSED METHOD: BOTTLENECK LTN ADAPTATION WITH SVD-BASED MATRIX COMPRESSION

3.1. Property analysis of LTN-based adaptation scheme

To prepare for matrix/vector manipulation, we re-express the trainable matrices/vectors of DNN and LTN as follows: $\widehat{\mathbf{W}}_{l_{\mathrm{SD}}} = [\widehat{\mathbf{w}}_1 \cdots \widehat{\mathbf{w}}_{N_{l_{\mathrm{SD}}-1}}]$, $\mathbf{W}_{l_{\mathrm{SD}}} = [\mathbf{w}_1 \cdots \mathbf{w}_{N_{l_{\mathrm{SD}}-1}}]$, $\mathbf{A}_{l_{\mathrm{SD}}} = [\boldsymbol{\alpha}_1 \cdots \boldsymbol{\alpha}_{N_{l_{\mathrm{SD}}-1}}]$, $\widehat{\mathbf{w}}_j = [\widehat{w}_{1j} \cdots \widehat{w}_{N_{l_{\mathrm{SD}}}j}]^{\mathrm{T}}$, $\mathbf{w}_j = [w_{1j} \cdots w_{N_{l_{\mathrm{SD}}}j}]^{\mathrm{T}}$, $\boldsymbol{\alpha}_j = [\alpha_{1j} \cdots \alpha_{N_{l_{\mathrm{SD}}-1}j}]^{\mathrm{T}}$, where $N_{l_{\mathrm{SD}}}$ is the number of nodes in $\mathrm{L}_{l_{\mathrm{SD}}}$, T is transpose. Strictly speaking, such column vectors as $\mathbf{w}_j$ should have an index that expresses its corresponding insertion layer, as with $\mathbf{w}_j^{l_{\mathrm{SD}}}$, but for simplicity, we omit it.

Then, through simple matrix/vector manipulations, we reach the following relation:

$$\widehat{\mathbf{w}}_j = \sum_{i=1}^{N_{l_{\mathrm{SD}}-1}} \alpha_{ij}\mathbf{w}_i. \tag{4}$$

Here, we can easily find that the relation $\widehat{\mathbf{w}}_j \in S(\mathbf{W}_{l_{\mathrm{SD}}})$ holds, where $S(\mathbf{W}_{l_{\mathrm{SD}}})$ is the column space spanned by the column vectors of $\mathbf{W}_{l_{\mathrm{SD}}}$. Similarly, we can obtain the following bias vector relation: $\mathbf{W}_{l_{\mathrm{SD}}}\mathbf{a}_{l_{\mathrm{SD}}} \in S(\mathbf{W}_{l_{\mathrm{SD}}})$. Accordingly, the potential adaptability of LTN, or in other words, the size of the parameter space searched by LTN, is determined by the dimension of $S(\mathbf{W}_{l_{\mathrm{SD}}})$, that is the real rank of $\mathbf{W}_{l_{\mathrm{SD}}}$. A question arising here concerns the *effective rank* of $\mathbf{W}_{l_{\mathrm{SD}}}$. If it is smaller than the real rank of $\mathbf{W}_{l_{\mathrm{SD}}}$, it is fundamentally possible to construct a low-dimensional parameter space that has the same feature-representation capability as $S(\mathbf{W}_{l_{\mathrm{SD}}})$. Then, because such use of a smaller number of basis vectors reduces the number of adaptable parameters, the new basis vectors, or the new smaller number of parameters, are expected to stably increase the effect of speaker adaptation, especially in cases where the available speech data for adaptation training are severely limited.

3.2. Low rank approximation of weight matrix using SVD

To find the above (smaller number of) basis vectors, we consider the application of SVD to the weight matrix of the SD module layer. For notation simplicity, omitting SD module layer index l_{SD}, we consider weight matrix $\mathbf{W}$ ($\in R^{m\times n}$) in this subsection. Then, $\mathbf{W}$ is decomposed as $\mathbf{W} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}}$, where $\mathbf{\Sigma}$ ($\in R^{m\times n}$) is the rectangular diagonal matrix whose diagonal elements are singular values σ_i, and $\mathbf{U}$ ($\in R^{m\times m}$) and $\mathbf{V}$ ($\in R^{n\times n}$) are the orthogonal matrices produced by SVD. In the same way, for matrix $\widetilde{\mathbf{W}}$ whose rank is k ($\leq$ rank $\mathbf{W}$), we obtain $\widetilde{\mathbf{W}} = \widetilde{\mathbf{U}}\widetilde{\mathbf{\Sigma}}\widetilde{\mathbf{V}}^{\mathrm{T}}$, where $\widetilde{\mathbf{\Sigma}}$ ($\in R^{k\times k}$) is the diagonal matrix produced by retaining the k largest singular value elements (while removing the remainder) in $\mathbf{\Sigma}$, $\widetilde{\mathbf{U}}$ ($\in R^{m\times k}$) is the matrix produced by retaining only the k column vectors of $\mathbf{U}$, each corresponding to the k largest singular values, and $\widetilde{\mathbf{V}}$ ($\in R^{n\times k}$) is the matrix that is similarly produced from $\mathbf{V}$. Finally, using $\widetilde{\mathbf{W}}$, we can obtain the following approximation of $\mathbf{W}$:

$$\widetilde{\mathbf{W}} = \widetilde{\mathbf{U}}\widetilde{\mathbf{\Sigma}}\widetilde{\mathbf{V}}^{\mathrm{T}} \approx \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}} = \mathbf{W}. \tag{5}$$

Eq. (5) shows that the space spanned by the column vectors of $\mathbf{W}$ can be approximated by the k basis vectors of $\widetilde{\mathbf{U}}$.

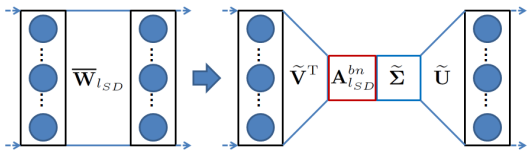

Fig. 2. Adaptation procedure consisting of SVD-based weight matrix-size reduction and bottleneck LTN insertion. For simplicity, no biases are depicted.

3.3. Adaption procedure using bottleneck LTN

From the results of 3.1 and 3.2, we propose a new adaptation scheme, which adopts a size-reduced LTN SD module, for improving the SAT-DNN-LTN recognizer. The proposed scheme is illustrated in Figure 2 and formalized as follows:

(i) It first conducts the SAT-based training of the entire DNN, based on the procedure in 2.2.

(ii) It next approximates, with the SVD-based matrix decomposition, the weight matrix of SD module $\overline{\mathbf{W}}_{l_{\mathrm{SD}}}$ as $\overline{\mathbf{W}}_{l_{\mathrm{SD}}} \approx \widetilde{\mathbf{U}}_{l_{\mathrm{SD}}}\widetilde{\mathbf{\Sigma}}_{l_{\mathrm{SD}}}\widetilde{\mathbf{V}}^{\mathrm{T}}_{l_{\mathrm{SD}}}$. Only the k basis vectors, which correspond to the k largest singular values of $\overline{\mathbf{W}}_{l_{\mathrm{SD}}}$, are selected in $\widetilde{\mathbf{\Sigma}}_{l_{\mathrm{SD}}}$, and therefore $\overline{\mathbf{W}}_{l_{\mathrm{SD}}}$ is approximately replaced by the bottleneck network $\widetilde{\mathbf{U}}_{l_{\mathrm{SD}}}\widetilde{\mathbf{\Sigma}}_{l_{\mathrm{SD}}}\widetilde{\mathbf{V}}^{\mathrm{T}}_{l_{\mathrm{SD}}}$.

(iii) It inserts a new small-sized LTN into the bottleneck SD module layer whose size (number of nodes) is reduced from n to k. This situation is illustrated in the right-side picture in Figure 2. To distinguish this small-sized LTN from the LTN originally used in the SAT-DNN-LTN framework, we refer to this newly inserted LTN as *bottleneck LTN*, and denote it by weight matrix $\mathbf{A}^{\mathrm{bn}}_{l_{\mathrm{SD}}}$ and bias vector $\mathbf{a}^{\mathrm{bn}}_{l_{\mathrm{SD}}}$. Accordingly, we also refer to the layer into which the bottleneck LTN is inserted as *bottleneck layer*, with *bottleneck size* for k. Then, the outputs from SD module layer $\mathrm{L}_{l_{\mathrm{SD}}}$ are given as follows:

$$\mathbf{z}_{l_{\mathrm{SD}}} = \sigma\Big(\widetilde{\mathbf{U}}_{l_{\mathrm{SD}}}\widetilde{\mathbf{\Sigma}}_{l_{\mathrm{SD}}}(\mathbf{A}^{\mathrm{bn}}_{l_{\mathrm{SD}}}\widetilde{\mathbf{V}}^{\mathrm{T}}_{l_{\mathrm{SD}}}\mathbf{z}_{l_{\mathrm{SD}}-1} + \mathbf{a}^{\mathrm{bn}}_{l_{\mathrm{SD}}}) + \mathbf{b}_{l_{\mathrm{SD}}}\Big). \quad (6)$$

(iv) It finally adapts only the bottleneck LTN using the speech data of a target speaker.

In the above scheme definition, we inserted the bottleneck LTN between $\widetilde{\mathbf{V}}^{\mathrm{T}}_{l_{\mathrm{SD}}}$ and $\widetilde{\mathbf{\Sigma}}_{l_{\mathrm{SD}}}$ (See Figure 2 and (6)). However, in principle, it can also be inserted between $\widetilde{\mathbf{\Sigma}}_{l_{\mathrm{SD}}}$ and $\widetilde{\mathbf{U}}_{l_{\mathrm{SD}}}$. We pre-experimentally investigated both ways of insertion and found that the way of (6) worked more stably than its counterpart. Therefore, we adopt it in this paper.

3.4. SAT-based DNN retraining using bottleneck LTN

Based on the definition of bottleneck LTN, it is expected to fundamentally retain the same degree of feature-representation capability as the original weight matrix $\overline{\mathbf{W}}_{l_{\mathrm{SD}}}$ has, provided its size (i.e., bottleneck size) exceeds a certain necessary level. However, removing the basis vectors, which correspond to small singular values, possibly makes the capability of $\widetilde{\mathbf{U}}_{l_{\mathrm{SD}}}\widetilde{\mathbf{\Sigma}}_{l_{\mathrm{SD}}}\mathbf{A}^{\mathrm{bn}}_{l_{\mathrm{SD}}}\widetilde{\mathbf{V}}^{\mathrm{T}}_{l_{\mathrm{SD}}}$ lower than that of $\overline{\mathbf{W}}_{l_{\mathrm{SD}}}\mathbf{A}_{l_{\mathrm{SD}}}$; this is because if those singular values are not truly zero, their corresponding basis vectors actually contribute to widening the adaptable parameter space. Therefore, the simple size reduction achieved by letting only the large singular value basis vectors remain may not necessarily be sufficient for initializing the bottleneck LTN for the successive adaptation; moreover, if needed, some countermeasures should be added to solve this insufficiency. One possible solution is to re-conduct the SAT-based training of the entire DNN part between (ii) and (iii) in our scheme shown above. The SAT-based DNN training here is basically the same as that in the original SAT-DNN-LTN recognizer, with the only difference being that the training here involves the small-sized bottleneck LTNs. Note that, similar to the full-sized LTN in the SAT-DNN-LTN framework, the bottleneck LTN is switched along with the speech data selection for every training speaker. This additional SAT procedure will correct the degradation of the SAT concept caused by the bottleneck LTN insertion that deteriorates the SAT-optimized SD module.

4. EXPERIMENTS

4.1. Conditions

We tested our proposed scheme under the supervised adaptation setups in the same experiment environments as in our previous studies on the original SAT-DNN and SAT-DNN-LTN recognizers [1, 2]. We prepared three datasets: training (300 speakers), validation (10 speakers), and testing (28 speakers). The total length of the training data was about 75 hours (although we previously reported 150 hours [1, 2], it should have read 75 hours). The validation data were used for setting hyper-parameters. The average length of the testing data was about 8.5 minutes.

The 39-dimensional acoustic feature vector, which consisted of MFCCs, log-power, and their first and second derivative, was concatenated to the 429 (39×11)-dimensional input to DNN. The DNN used had 7 layers and consisted of 429 input nodes, 4909 output nodes, and 2048 nodes in every hidden layer.

For comparison purposes, following our previous training procedures [1, 2], we first developed the baseline SI-DNN recognizer and the SAT-DNN-LTN recognizer that had no SVD-based bottleneck layer. Then, using the above SI-DNN and SAT-DNN-LTN recognizers as baselines, we also developed the following recognizers:

(a-1) SI-SVD: developed by replacing the SD module layer of SI-DNN with an SVD-based low-rank weight matrix.

(a-2) SI-SVD-RET: developed by reapplying the SI training to SI-SVD.

(b-1) SAT-SVD: developed by replacing the SD module layer of SAT-DNN-LTN with an SVD-based low-rank weight matrix.

(b-2) SAT-SVD-RESAT: developed by reapplying the SAT-based training to SAT-SVD, as described in 3.4.

(c-1) SA-SI-SVD, SA-SI-SVD-RET[1], SA-SAT-SVD, and SA-SAT-SVD-RESAT: Prefix SA indicates that the corresponding recognizers were speaker-adapted. The adaptation was done using the bottleneck LTN inserted into the SD module layer.

In the experiments, we allocated the SD module only to the layer L_2 ($l_{\mathrm{SD}} = 2$), which was shown to be the most effective for the SD module insertion [2]. We also set all other training conditions in the same way as in the previous experiment [2].

4.2. Results: Properties of LTN-based adaptation

Figure 3 shows the word error rates (WERs) of three types of recognizers, i.e., SI-SVD, SA-SI-SVD, and SA-SAT-SVD. The rates were obtained along with the different bottleneck sizes of 64, 128, 256, 512, 1024, and 2048; each of the rates was the average of the results obtained through the four-times cross-validation over the speech data of all 28 testing speakers. The WERs of SA-SI-SVD and SA-SAT-SVD, at the right-side of the figure, were exactly the same as those of SA-SI-LTN and SA-SAT-LTN (for these two, the speaker adaptation was done without SVD-based matrix conversion), respectively.

[1]SA-SI-SVD-RET basically corresponds to the recognizer in [6], although there are such small differences as the positioning of the SD modules.

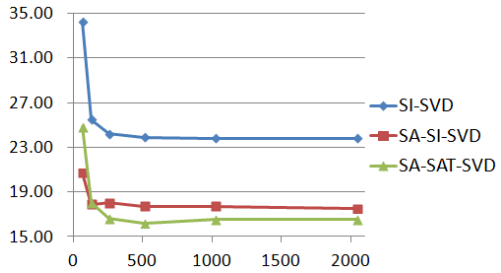

Fig. 3. Relationship between bottleneck size and recognition performance (word error rate [%]).

Note that in these cases, k was kept to 2048, although $\overline{\mathbf{W}}_2$ was decomposed based on SVD. As theoretically expected, SVD did not change the feature-representation capability of the matrix when all of the decomposed basis vectors were maintained. Figure 3 also shows that the original 2048-dimensional SD module matrix $\overline{\mathbf{W}}_2$ can be sufficiently approximated by such low-rank matrices as the 512-dimensional matrix: The WERs of all three recognizers had little degradation until their bottleneck sizes were reduced to 512. The results clearly support our analysis in 3.1 that the effective rank of $\overline{\mathbf{W}}_2$ is rather low and the bottleneck LTN having an appropriately selected size can sufficiently retain the adaptation capability of the original high-dimension matrix. Due to space limitations, we omit the results of the other recognizers shown in the list of 4.1 but they showed the same trend as in Figure 3.

4.3. Results: Adaptation using a small amount of speech data

As discussed in 3.1 and 3.4, the use of the bottleneck LTN may have a conflicting two-side effect. Its adaptable parameter reduction possibly increases the adaptability of SD modules, especially over the limited amount of adaptation data, but it also raise the danger of fundamentally decreasing the adaptability. To clarify this effect, we conducted adaptation experiments using two different sizes of bottleneck LTNs: one with $k = 512$ and the other with $k = 2048$. From the results in 4.2, the bottleneck LTN with $k = 512$ was shown to retain the representation capability of the original 2048-dimensional weight matrix $\overline{\mathbf{W}}_2$, while the bottleneck LTN with $k = 2048$ was also shown to be equivalent to $\overline{\mathbf{W}}_2$. Note that the bottleneck LTN with $k = 2048$ was produced with SVD but actually its structure was not a bottleneck. The experiment was conducted over the speech data, each longer than 6 minutes, of 18 testing speakers; for each testing (adaptation target) speaker, his/her first 3 minutes of utterances were reserved for adaptation, and the data of the remaining time were used for testing.

Table 1 shows the size conditions and gained WERs ([%]) of the evaluated recognizers. In the table, for each recognizer, we show the bottleneck size (bn size), the number of adaptable parameters of the SD module (# param), and the WERs for 4 different lengths of adaptation speech data, i.e., 180, 60, 30, and 15 seconds. Note that the length of testing speech data was set to the fixed value described in the above paragraph. Because the SI-SVD and SI-SVD-RET recognizers did not have the speaker adaptation mechanism, the same WERs gained without the adaptation are listed in all of the corresponding columns.

From the table, we can make three observations:

(i) The SAT-based recognizer retrained with SAT (SA-SAT-SVD-RESAT with $k = 512$) achieved the best WERs (bold-faced scores) for all of the adaptation speech data lengths, although the number of the adaptation parameters was only 6.25% of the original 2048-dimensional weight matrix.

Table 1. Experimental results (parameter sizes and word error rates [%]) obtained with supervised adaptation using different amounts of adaptation data.

recognizer	bn size	# param	180 s	60 s	30 s	15 s
SI-SVD	2048	4.2 M	24.0	24.0	24.0	24.0
SI-SVD	512	0.26 M	24.1	24.1	24.1	24.1
SA-SI-SVD	2048	4.2 M	18.8	20.3	21.1	21.5
SA-SI-SVD	512	0.26 M	18.9	20.4	21.3	21.7
SA-SAT-SVD	2048	4.2 M	17.8	19.3	20.5	21.3
SA-SAT-SVD	512	0.26 M	18.0	19.5	**20.4**	21.3
SI-SVD-RET	512	0.26 M	24.0	24.0	24.0	24.0
SA-SI-SVD-RET	512	0.26 M	18.9	20.2	21.3	21.7
SA-SAT-SVD-RESAT	512	0.26 M	**17.6**	**18.9**	**20.4**	**20.8**

(ii) The SAT-based recognizer (SA-SAT-SVD) constantly outperformed its counterpart SI-based recognizer (SA-SI-SVD) for all of the settings of adaptation speech data lengths and bottleneck sizes.

(iii) The SD module size reduction ($k = 2048 \rightarrow k = 512$) basically decreases the adaptation capability of its baseline non-reduced module in both the SI- and SAT-based recognizers.

Similarly to our approach, SVD-based matrix approximation has been studied in recent works with the aim of reducing the adaptation parameter size [6–8]. However, it was not fully clarified whether adaptation incorporating the parameter-size reduction achieved greater adaptation performance compared to adaptation without such size reduction, especially in cases where the speech data available for adaptation training were severely limited. Our results show that a simple use of SVD-based size reduction is not sufficient for improving this performance, but our novel SAT-based reinitialization procedure for the DNN front-end clearly helps the SAT-based bottleneck LTN to boost its adaptation power. In addition to the SAT-based reinitialization, we actually examined the SI retraining for the SI-SVD recognizer, but this alternative reinitialization was not effective for increasing the adaptation performances of the SA-SI-SVD recognizer. The difference in reinitialization effect probably arises from the training difference that the SAT-based approach reinitializes the bottleneck LTN together with the other DNN part in order to increase the adaptation capability of the LTN module but the SI-based approach gives no consideration to the adaptation.

5. CONCLUSION

In this paper, we theoretically analyzed the role of the SD module weight matrix in representing the capability of speaker adaptation, and based on the analysis results, we proposed the LTN-based bottleneck speaker adaptation scheme for our previous SAT-DNN-LTN recognizer. We then elaborated its effectiveness in a difficult TED Talks task, and successfully demonstrated its ability to improve WERs. Specifically, our adaptation scheme improved, using a radically reduced SD module of only 6.25% of its original size, the WERs of the baseline SAT-DNN-LTN recognizer, even in cases where a severely limited amount of adaptation data was available.

For the SAT-based recognizer, we applied the SVD-based matrix-size reduction after the SAT-based DNN training. However, it is possible to apply it before the SAT-based training. This latter approach will be an interesting future study.

ACKNOWLEDGEMENTS

This work was supported in part by JSPS Grants-in-Aid for Scientific Research No. 26280063, MEXT-Supported Program "Driver-in-the-Loop", and Grant-in-Aid for JSPS Fellows. The authors appreciate their financial support.

6. REFERENCES

[1] T. Ochiai, S. Matsuda, X. Lu, C. Hori, and S. Katagiri, "Speaker adaptive training using deep neural networks," in *Proc. ICASSP*, 2014, pp. 6399–6403.

[2] T. Ochiai, S. Matsuda, H. Watanabe, X. Lu, C. Hori, and S. Katagiri, "Speaker adaptive training for deep neural networks embedding linear transformation networks," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, 2015, pp. 4605–4609.

[3] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *Proc. ASRU*, 2011, pp. 24–29.

[4] P. Swietojanski and S. Renals, "Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models," in *Proc. SLT*, 2014, pp. 171–176.

[5] Y. Zhao, J. Li, J. Xue, and Y. Gong, "Investigating online low-footprint speaker adaptation using generalized linear regression and click-through data," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, 2015, pp. 4310–4314.

[6] J. Xue, J. Li, D. Yu, M. Seltzer, and Y. Gong, "Singular value decomposition based low-footprint speaker adaptation and personalization for deep neural network," in *Proc. ICASSP*, 2014, pp. 6359–6363.

[7] K. Kumar, C. Liu, K. Yao, and Y. Gong, "Intermediate-layer dnn adaptation for offline and session-based iterative speaker adaptation," in *Proc. Interspeech*, 2015, pp. 1091–1095.

[8] S. Li, X. Lu, Y. Akita, and T. Kawahara, "Ensemble speaker modeling using speaker adaptive training deep neural network for speaker adaptation," in *Proc. Interspeech*, 2015, pp. 2892–2896.

[9] T. Tan, Y. Qian, M. Yin, Y. Zhuang, and K. Yu, "Cluster adaptive training for deep neural network," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, 2015, pp. 4325–4329.

[10] M. Delcroix, K. Kinoshita, T. Hori, and T. Nakatani, "Context adaptive deep neural networks for fast acoustic model adaptation," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, 2015, pp. 4535–4539.

[11] C. Wu and M. Gales, "Multi-basis adaptive neural network for rapid adaptation in speech recognition," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, 2015, pp. 4315–4319.

[12] H. Liao, "Speaker adaptation of context dependent deep neural networks," in *Proc. ICASSP*, 2013, pp. 7947–7951.

[13] D. Yu, K. Yao, H. Su, G. Li, and F. Seide, "KL-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition," in *Proc. ICASSP*, 2013, pp. 7893–7897.

[14] O. Abdel-Hamid and H. Jiang, "Fast speaker adaptation of hybrid NN/HMM model for speech recognition based on discriminative learning of speaker code," in *Proc. ICASSP*, 2013, pp. 7942–7946.

[15] S. Xue, O. Abdel-Hamid, H. Jiang, and L. Dai, "Direct adaptation of hybrid DNN/HMM model for fast speaker adaptation in LVCSR based on speaker code," in *Proc. ICASSP*, 2014, pp. 6339–6343.

[16] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *Proc. ASRU*, 2013, pp. 55–59.

[17] S. M. Siniscalchi, J. Li, and C.-H. Lee, "Hermitian based hidden activation functions for adaptation of hybrid HMM/ANN models," in *Proc. Interspeech*, 2012.

[18] T. Anastasakos, J. McDonough, R. Schwartz, and J. Makhoul, "A compact model for speaker-adaptive training," in *Proc. ICSLP*, 1996, vol. 2, pp. 1137–1140.

[19] J. Neto, L. Almeida, M. Hochberg, C. Martins, L. Nunes, S. Renals, and T. Robinson, "Speaker-adaptation for hybrid HMM-ANN continuous speech recognition system," in *Proc. EUROSPEECH*, 1995, pp. 2171–2174.

[20] R. Gemello, F. Mana, S. Scanzio, P. Laface, and R. De Mori, "Linear hidden transformations for adaptation of hybrid ANN/HMM models," *Speech Communication*, vol. 49, no. 10-11, pp. 827–835, 2007.

[21] B. Li and K. C. Sim, "Comparison of discriminative input and output transformations for speaker adaptation in the hybrid NN/HMM systems," in *Proc. Interspeech*, 2010, pp. 526–529.