AUTOMATIC NODE SELECTION FOR DEEP NEURAL NETWORKS USING GROUP LASSO REGULARIZATION

Tsubasa Ochiai¹,*, Shigeki Matsuda¹, Hideyuki Watanabe², and Shigeru Katagiri¹

¹ Graduate School of Science and Engineering, Doshisha University, Kyoto, Japan
² Advanced Telecommunications Research Institute International, Kyoto, Japan
*eup1105@mail4.doshisha.ac.jp

ABSTRACT

We examine the effect of the Group Lasso (gLasso) regularizer in selecting the salient nodes of Deep Neural Network (DNN) hidden layers by applying a DNN-HMM hybrid speech recognizer to TED Talks speech data. We test two types of gLasso regularization, one for outgoing weight vectors and another for incoming weight vectors, as well as two sizes of DNNs: 2048 hidden layer nodes and 4096 nodes. Furthermore, we compare gLasso and L2 regularizers. Our experiment results demonstrate that our DNN training, in which the gLasso regularizer was embedded, successfully selected the hidden layer nodes that are necessary and sufficient for achieving high classification power.

Index Terms— Deep neural networks, Group Lasso regularization, Speech recognition

1. INTRODUCTION

Motivated by the recent rise of Deep Neural Networks (DNNs), we studied their utility for speech recognition by developing DNN-HMM hybrid speech recognizers [1, 2]. A DNN’s high capability is mainly derived from a deep layer structure and a massive amount of network weight parameters. Therefore, roughly speaking, the larger and deeper the network, the higher its performances are. However, a large network is obviously undesirable from the viewpoints of computational load and memory size; it is also unfavorable from the viewpoint of controlling training robustness to unseen data. To meet this requirement for finding a small, necessary, and sufficient DNN structure, several approaches have reshaped the network structure [3-5] or pruned the network nodes [6]. However, these methods assumed retraining or adapting a size-reduced network for high discriminative power. Therefore, there needs obviously exists for developing training methods that can automatically (without additional retraining and adaptation) find a small but sufficient DNN structure.

Seeking such development, we propose in this paper a new training scheme that embeds group Lasso (gLasso) regularization [2, 8] for hidden layer weight vectors. To examine its feasibility, we implement it in a large-scale DNN-HMM speech recognizer and conduct various evaluation experiments in a difficult task: TED Talks.

Quite recently, a gLasso-based idea similar to ours was reported [9]. However, we independently studied it and elaborated the characteristics of its scheme in large-scale experiments. In Section 4, we will discuss the relationship between our work and related works, including [9].

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.
vectors, which form the columns of weight matrix $W^l$; the incoming weight vectors form the rows of $W^l$.

In the figure, for example, if the norm of $w_i^l$ is nearly zero and the norm for $w_i^{l-1}$ exceeds zero, $\alpha^l$ is essentially based only on $z_i^{l-1}$, i.e., the outputs from the $l$th node in the $(l-1)$-th layer. In this situation, even if we remove node $\nu^1_{l-1}$, $\alpha^l$ has no large changes and retains the DNN performances that were gained before node pruning. On the other hand, again for example, if the norm of $w_i^l$ is nearly zero and the norm for $w_i^{l+1}$ exceeds zero, $\alpha^{l+1}$ is essentially based on $z_i^l$ and $b^l_2 (\approx z^l_0)$. In this situation, even if we prune node $\nu^2_l$, $\alpha^{l+1}$ has no large changes and the DNN performances accordingly remain almost the same. However, because the node is pruned based on its weight vector norm, its bias remains as a constant. Therefore, we have to remember here to shift the bias for the pruned node to the upper layer nodes.

As above, regardless of the weight vector grouping, if we can control the values of the weight vector norm in the DNN training, or more precisely, selectively minimize the norm values, we can control the values of the weight vector norm in the DNN training, enabled by their corresponding small norm values and only the salient nodes will be retained.

To implement such training, we adopt gLasso regularization for the outgoing or incoming weight vectors in standard DNN training using Cross Entropy (CE) loss. Focusing on the outgoing weight vector case, we formalize our proposed training scheme below. The resulting formalization is basically common to that in the incoming weight vector case. Note that the target layer for grouping is different between types of grouping.

Our training scheme starts by defining regularized loss $L(\cdot)$ as follows:

$$L(\Lambda) = E(\Lambda) + \alpha R_{\text{group}}(\{W^l\}_{l=2}^L) + \beta \hat{R}_{\text{L2}}(W^1, \{b^l\}_{l=1}^L),$$

where $\Lambda$ is a set of trainable parameters of DNN, i.e., $\Lambda = \{\{W^l, b^l\}_{l=1}^L\}$, $E(\cdot)$ is the CE loss, $R_{\text{group}}(\cdot)$ and $\hat{R}_{\text{L2}}(\cdot)$ respectively the gLasso and L2 regularizers, and $\alpha$ and $\beta$ are positive constants that control the effects of their corresponding regularizers. Here the gLasso regularizer is specified as

$$\hat{R}_{\text{group}}(\{W^l\}_{l=2}^L) = \sum_{l=2}^L R_{\text{group}}(W^l),$$

where $R_{\text{group}}(W^l) = \sum_{j=1}^{N_l} \|w^l_j\|$, and the L2 regularizer is specified as

$$\hat{R}_{\text{L2}}(W^1, \{b^l\}_{l=1}^L) = R_{\text{L2}}(W^1) + \sum_{l=1}^L R_{\text{L2}}(b^l),$$

where $R_{\text{L2}}(W^l) = \frac{1}{2} \|W^l\|^2$ and $R_{\text{L2}}(b^l) = \frac{1}{2} \|b^l\|^2$.

### 2.3.1. Definition of regularized loss

The node selection in [2,2] is naturally expected to be done so that only the salient nodes that produce useful outputs for increasing the DNN’s performances are retained. If DNN training is done to concurrently minimize a loss function that reflects the DNN’s classification errors and a function that reflects the norm values of the outgoing/incoming weight vectors, the unnecessary nodes, which produce useless outputs (for the DNN performances), are automatically disabled by their corresponding small norm values and only the salient nodes will be retained.

To implement such training, we adopt gLasso regularization for the outgoing or incoming weight vectors in standard DNN training using Cross Entropy (CE) loss. Focusing on the outgoing weight vector case, we formalize our proposed training scheme below. The resulting formalization is basically common to that in the incoming weight vector case. Note that the target layer for grouping is different between types of grouping.

Our training scheme starts by defining regularized loss $L(\cdot)$ as follows:

$$L(\Lambda) = E(\Lambda) + \alpha R_{\text{group}}(\{W^l\}_{l=2}^L) + \beta \hat{R}_{\text{L2}}(W^1, \{b^l\}_{l=1}^L),$$

where $\Lambda$ is a set of trainable parameters of DNN, i.e., $\Lambda = \{\{W^l, b^l\}_{l=1}^L\}$, $E(\cdot)$ is the CE loss, $R_{\text{group}}(\cdot)$ and $\hat{R}_{\text{L2}}(\cdot)$ respectively the gLasso and L2 regularizers, and $\alpha$ and $\beta$ are positive constants that control the effects of their corresponding regularizers. Here the gLasso regularizer is specified as

$$\hat{R}_{\text{group}}(\{W^l\}_{l=2}^L) = \sum_{l=2}^L R_{\text{group}}(W^l),$$

where $R_{\text{group}}(W^l) = \sum_{j=1}^{N_l} \|w^l_j\|$, and the L2 regularizer is specified as

$$\hat{R}_{\text{L2}}(W^1, \{b^l\}_{l=1}^L) = R_{\text{L2}}(W^1) + \sum_{l=1}^L R_{\text{L2}}(b^l),$$

where $R_{\text{L2}}(W^l) = \frac{1}{2} \|W^l\|^2$ and $R_{\text{L2}}(b^l) = \frac{1}{2} \|b^l\|^2$.

### 2.3.2. Effects of gLasso regularization

The status of $\Lambda$ that corresponds to the minimum of $L(\Lambda)$ will provides high classification power with selected network nodes. To find that status, we minimize $L(\Lambda)$ using the standard gradient-based optimization procedure.

In the above gradient-based loss minimization for $L(\Lambda)$, the gradient terms of the regularizers play a key role in accelerating the minimization. In the L2 regularizer case, the gradient of $\hat{R}_{\text{L2}}(\cdot)$ is given as follows, for example, in terms of $w^l_j$:

$$\frac{\partial \hat{R}_{\text{L2}}(W^l)}{\partial w^l_j} = w^l_j.$$  (5)

In contrast, the gradient of the gLasso regularizer becomes

$$\frac{\partial \hat{R}_{\text{group}}(W^l)}{\partial w^l_j} = \frac{w^l_j}{\|w^l_j\|}.$$  (6)

Eq. (6) clearly indicates the effects of gLasso regularization; the (“outgoing” in this case) weight vector rapidly converges to zero when its norm is small. As a result, the node outputs, which are less useful for the minimization of $L(\cdot)$, are disabled in an early stage of the loss minimization, and the retained node outputs will be dominantly used for the loss minimization.

### 2.4. Node Selection Procedure

If the weight vector norm becomes sufficiently small after the above loss minimization, we can simply prune its corresponding nodes using a threshold value ($\theta$). The node selection procedure is summarized as follows: 1) calculate the gLasso0 norm (column norm) or the gLasso1 norm (row norm) for all of the hidden layer nodes, 2) prune the nodes whose norm values are less than $\theta$, and 3) shift the bias for the pruned node to the upper layer nodes, especially in the gLasso case. (See [2,2] for the definitions of gLasso0 and gLasso1.)

### 3. EXPERIMENTS

#### 3.1. Settings

To evaluate our proposed DNN training and node selection method, we conducted speech recognition experiments using our DNN-HMM hybrid speech recognizer over a TED Talks corpus that consists of lecture speech data spoken by 793 speakers. For the evaluation experiments, we divided the data into the following three sets: training data by 774 speakers, validation data by 8 speakers, and testing data by 11 speakers. The total length of the training data was about 62 hours, the number of senone classes was 3,944, and the vocabulary size was 179, 604.

We represented the input speech as a series of acoustic feature vectors, each of which was based on Mel-scale Frequency Cepstrum Coefficients and of 429 dimensions. See [1] for the details of the input pattern definitions.

In our DNN-HMM hybrid speech recognizer, the front-end DNN part was the 6-layer Multi-Layer Perceptron ($L = 6$ in [2,1]) with a sigmoid activation function, and the post-end HMM part used a context-dependent acoustic model and a 3-grm language model. We initialized the DNN part by the RBM pre-training [10]. Independently of the DNN part, we developed a HMM part using Boosted MMI training [11].

The main target of our DNN training was to increase the senone classification accuracy (SCA), whose categories were indicated by the DNN outputs. However, we also evaluated the training results using the word error rate (WER) obtained by the post-end HMM part of the hybrid recognizer.
3.2. Evaluation procedures

We ran gradient-based training for minimizing $L(\theta)$ in (2) several times, along with changing such hyper-parameters as $\alpha$ and $\beta$ in (2). Nevertheless, for simplicity, we restricted $\beta$ to $\beta = 0.1\alpha$ or $\beta = 0$. In every training run, we repeated twenty epochs, in which each of the whole set of training samples was used only once. Among the different trained statuses of $\Lambda$, we selected the one that produced the highest SCA value over the validation data, evaluated the SCA values for the selected status of $\Lambda$ over the testing data, and measured the WER values over the testing data using the selected DNN in the hybrid recognizer.

In addition to experiments with our proposed training associated with the gLasso regularizer, we also tested (for comparison purposes) the training using the L2 regularizer for all of the weight matrices and all of the bias vectors. Then, the loss in (2) was replaced in this additional case by

$$L(\Lambda) = E(\Lambda) + \beta \sum_{l=1}^{L} \left( R_{L2}(\mathbf{W}^l) + R_{L2}(\mathbf{b}^l) \right). \quad (7)$$

To elaborate the effects of our proposed training method, we also tested two different DNN sizes: one with 2048 nodes for every hidden layer (DNN$_{2048}$) and another with 4096 nodes for every hidden layer (DNN$_{4096}$). For both DNN$_{2048}$ and DNN$_{4096}$, we evaluated the following four types of training: 1) with the gLasso regularizer for the outgoing hidden layer weight vectors (gLasso$_{O}$), 2) with the gLasso regularizer for the incoming hidden layer weight vectors (gLasso$_{I}$), 3) with the L2 regularizer for the outgoing hidden layer weight vectors (L2$_{O}$), and 4) with the L2 regularizer for the incoming hidden layer weight vectors (L2$_{I}$). Particularly in the L2$_{O}$ and L2$_{I}$ cases, we conducted common L2-regularized training (which corresponds to the training using (2)) but selected different nodes, based on the differences in grouping. Moreover, as stated in 2.4, we used in all four cases the L2 regularizer for all of the bias vectors; We also used the L2 regularizer for the input layer weights in the gLasso$_{I}$ case, and for the output layer weights in the L2$_{O}$ case.

For convenience, we collectively refer to both the gLasso$_{O}$ and gLasso$_{I}$ training cases as gLasso and L2 for both the L2$_{O}$ and L2$_{I}$ cases.

3.3. Results

3.3.1. Fundamental performances

In Table 1 we summarize the SCA and WER values that we achieved for DNN$_{2048}$ in the four training cases: gLasso$_{O}$, gLasso$_{I}$, L2$_{O}$, and L2$_{I}$. SCA$_{B}$ and WER$_{B}$ represent the SCA and WER values (%), both obtained before the node selection; the SCA and WER values were obtained after the node selection for SCA$_{A}$ and WER$_{A}$.

First, we focus on the results in the SCA$_{B}$ and WER$_{B}$ columns. Regardless of the weight vector grouping type and the regularizer type, DNN training achieved such accurate classification performances as an SCA of about 46% and a WER less than 19%. They also show that the gLasso procedures achieved the performance that were competitive with the L2 procedures, which are widely used in the neural network training.

### Table 1. Achieved classification performances (%) in DNN$_{2048}$

| Regularizer | SCA$_{B}$ | WER$_{B}$ | SCA$_{A}$ | WER$_{A}$ |
|-------------|-----------|-----------|-----------|-----------|
| gLasso$_{O}$ | 46.2      | 18.7      | 46.2      | 18.8      |
| gLasso$_{I}$ | 46.1      | 18.2      | 46.1      | 18.2      |
| L2$_{O}$    | 45.9      | 18.6      | 22.8      | 42.6      |
| L2$_{I}$    | 45.9      | 18.6      | 8.2       | 100.0     |

Next, we observe the characteristics of the weight vector norm values that were obtained by the training. Figure 2 shows a histogram of the outgoing weight vector norm values in gLasso$_{O}$ or L2$_{O}$ cases. Training was done for DNN$_{2048}$. Top: gLasso$_{O}$, Bottom: L2$_{O}$.

Due to space limitations, we omit the introduction of the results in the gLasso$_{I}$ and L2$_{I}$ cases. However, we obtained norm value distribution that closely resembled that in Figure 2.

We return to Table 1 and compare the results in the SCA$_{B}$ and WER$_{B}$ columns with those in the SCA$_{A}$ and WER$_{A}$ columns.

We executed node selection by setting $\theta$ to $10^{-2}$, which is clearly far from the norm values in either of the two separate regions generated by the gLasso training. Using this threshold, we pruned 3161 nodes from the five hidden layers in the gLasso$_{O}$ case (the number of retained hidden layer nodes = 7079) and 3368 nodes in the gLasso$_{I}$ (number of retained hidden layer nodes = 6872) case. The performances gained after the node selection (in the SCA$_{A}$ and WER$_{A}$ columns) are basically the same with those gained before the node selection (in the SCA$_{B}$ and WER$_{B}$ columns). The comparison here clearly proves that, regardless of the type of weight vector grouping, gLasso training sorted out the weight vectors by selectively reducing the norm values for some weight vectors and allowed node selection simply based on thresholding.

As Figure 3 shows, pruning a certain amount of nodes is not easy from L2-based DNNs by setting a threshold. For comparison, we removed (in ascending order) the small-norm nodes from the DNN produced by the common L2-regularized training in the L2$_{O}$ case until the number of removed nodes became identical to the gLasso$_{O}$ case: 3161; similarly, in the L2$_{I}$ case, we removed 3368 nodes from the L2$_{I}$-based DNN, which was the same with the network used in the L2$_{O}$ case. The scores in the SCA$_{A}$ and WER$_{A}$ columns of Table 1 were obtained using these size-reduced L2-based DNNs. They are much worse than those gained before the node selection and clearly show that both the L2$_{O}$ and L2$_{I}$ cases failed to sort out the weight vectors (as a result, their corresponding nodes).

To further analyze the effects of regularizer selection, we observed the changes in the SCA scores along with gradually removing hidden layer nodes, by 100 nodes, in increasing order of their norm values. Figure 4 shows the SCA scores, each of which is a function...
and DNN

4096

found the following: 1) Node selection was stably realized with the higher hidden layers as the 3rd and 4th.

training progress. 2) It occurred in such comparatively early epochs a trend that mainly reduces the weight vector norm values in the hidden layers (1st through 5th) of the gLasso

cal axis. If the nodes do not exist in the small norm value region, the indicates the number of removed nodes; the SCA scores for the verti-

SCA scores gradually decrease as nodes are removed. On the other hand, if the nodes exist in the small norm value region, the SCA values remain high and later decrease. The curves in the figure clearly prove that the L2 procedure corresponds to the former situation and the gLasso procedure achieves the latter desirable situation.

3.3.2. Properties of gLasso procedure

When introducing gLasso’s training scheme in 2,3 we stated that the disposable nodes are disabled in some early training stage. To investigate this point, we observed the number of disposable hidden layer nodes for every hidden layer as a function of the training epoch. Here we define a disposable node as one whose norm value is smaller than 10−2. Figure 4 illustrates the observed numbers for all five hidden layers (1st through 5th) of the gLasso0-based DNN2048. We found the following: 1) Node selection was stably realized with the training progress. 2) It occurred in such comparatively early epochs as the 1st through 5th epochs. 3) It dominantly occurred in such higher hidden layers as the 3rd and 4th.

The results in Figure 4 suggest that the gLasso procedure shows a trend that mainly reduces the weight vector norm values in the higher hidden layers. To scrutinize this trend, in both the DNN2048 and DNN4096 cases, we pruned the disposable nodes using the 10−2 threshold and observed the number of selected (retained) nodes for every hidden layer. Figure 5 illustrates those numbers. Regardless of the weight vector grouping (outgoing vs. incoming) or the DNN size (DNN2048 vs. DNN4096), we found that the gLasso procedure suppressed the weight vector norm values, specifically in the 3rd and 4th hidden layers, and achieved a size-reduced and reshaped network structure that was shared in all of the tested cases of gLasso0, gLasso1, DNN2048, and DNN4096.

4. RELATED WORK

An orthodox approach for the size reduction of a large matrix is to use Singular Value Decomposition. A large weight matrix in DNN is decomposed into small matrices and re-expressed (in the sense of low-rank approximation) by replacing the small eigenvalues of the decomposed diagonal matrix by zero 4,4. This approach allows parameter reduction without serious degradation of the original discriminative power of the network. However, it does not reduce the number of hidden layer nodes, and therefore it is probably insufficient to achieve a desirable amount of network parameter reduction, especially in the case of using the network with the huge number of hidden layer nodes.

For directly pruning the hidden layer nodes, the L1 norm of the weight vectors was used as an important function. It resembles the weight-grouping-based norm that we use in our proposed training procedure 6. However, because such mechanism of reducing the norms in the training procedure as ours was not implemented, DNN restructuring had to rely on additional retraining results.

To find a desirable DNN structure, the Genetic Algorithm (GA) has been applied 12. It has high potential for generating a target structure, but it often needs an enormous amount of computational resources.

Compared with the above related studies, our proposed training that embeds the gLasso regularizer is characterized by two main advantages: 1) Even if no additional training is conducted, it can easily produce node-pruned DNNs without performance degradation, and 2) its computation time is almost the same as conventional L2 regularization.

Motivated by the gLasso procedure defined in linear regression 7, an approach similar to ours was proposed 9. The gLasso regularizer was defined for the outgoing vector grouping of DNN and its effect was evaluated in various small- and middle-sized tasks that classify fixed-dimensional patterns. Compared to this work, we studied in this paper both the incoming and outgoing vector groupings and also elaborated our proposed scheme in a large-scale speech recognition environment. Here, incoming vector grouping can be considered a filtering function and will be appropriate for node selection or restructuring for Convolutional Neural Networks.

5. CONCLUSION

We investigated the feasibility of the gLasso procedure, i.e., DNN training associated with the gLasso regularizer, for hidden layer weight vectors in a large-scale speech recognition task with the DNN-HMM hybrid speech recognizer in a TED Talks task. We elaborated the nature of gLasso training by applying it to both outgoing and incoming weight vector groupings and compared the gLasso and L2 procedures, where the L2 regularizer was used for all of the weight matrixes and all of the bias vectors. From the experiment results, we clearly demonstrated that our gLasso procedure automatically (without additional training) disabled less useful hidden layer nodes without degradation in DNN classification performance and successfully produced a small, necessary, and sufficient DNN structure.
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. 6349-6353.

[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. ICASSP, 2015, pp. 4605-4609.

[3] J. Xue, J. Li, and Y. Gong, “Restructuring of deep neural network acoustic models with singular value decomposition,” in Proc. INTERSPEECH, 2013, pp. 2365-2369.

[4] 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.

[5] T. Ochiai, S. Matsuda, H. Watanabe, X. Lu, H. Kawai, and S. Katagiri, “Bottleneck linear transformation network adaptation for speaker adaptive training-based hybrid DNN-HMM speech recognizer,” in Proc. ICASSP, 2016, pp. 5015-5019.

[6] T. He, Y. Fan, Y. Qian, T. Tan, and Kai Yu, “Reshaping deep neural network for fast decoding by node-pruning,” in Proc. ICASSP, 2014, pp. 245-249.

[7] M. Yuan and Y. Lin, “Model selection and estimation in regression with grouped variables,” Journal of the Royal Statistical Society: Series B (Statistical Methodology), vol. 68, no. 1, pp. 49-67, 2006.

[8] N. Simon, J. Friedman, T. Hastie, and R. Tibshirani, “A sparse-group lasso” Journal of Computational and Graphical Statistics, vol. 22, no. 2, pp. 231-245, 2013.

[9] S. Scardapane, Danilo Commiinniello, Amir Hussain, and Aurelio Uncini, “Group Sparse Regularization for Deep Neural Networks,” arXiv:1607.00485v1 (2016).

[10] G. E. Hinton, S. Osindero, and Y. Teh, “A fast learning algorithm for deep belief nets,” Neural Computation, vol. 18, pp. 1527-1554, 2006.

[11] D. Povey, D. Kanovsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah, “Boosted MMI for model and feature space discriminative training,” in Proc. ICASSP, 2008, pp. 4057-4060.

[12] T. Shinozaki and S. Watanabe, “Structure discovery of deep neural network based on evolutionary algorithms,” in Proc. ICASSP, 2015, pp. 4979-4983.