Effective SVD-Based Deep Network Compression for Automatic Speech Recognition

Hao Fu[1(✉)], Yue Ming[1], Yibo Jiang[2], and Chunxiao Fan[1]

[1] Beijing University of Posts and Telecommunications, Beijing, China
{fuhao2013,yming,eetxwl}@bupt.edu.cn
[2] Ningbo Xitang Technologies Inc., Ningbo, China
jiangybtt@163.com

Abstract. Neural networks improve speech recognition performance significantly, but their large amount of parameters brings high computation and memory cost. To work around this problem, we propose an efficient network compression method based on Singular Value Decomposition (SVD), Simultaneous Iterative SVD Reconstruction via Loss Sensitive Update (SISVD-LU). Firstly, we analyse the matrices' singular values to learn the sparsity in every single layer and then we apply SVD on the most sparse layer to factorize the weight matrix into two or more matrices with least reconstruction errors. Secondly, we reconstruct the model using our *Loss Sensitive Update* strategy, which propagates the error across layers. Finally, we utilize *Simultaneous Iterative Compression* method, which factorizes all layers simultaneously and then iteratively minimize the model size while keeping the accuracy. We evaluate the proposed approach on the two different LVCSR datasets, AISHELL and TIMIT. On AISHELL mandarin dataset, we can obtain 50% compression ratio in single layer while maintaining almost the same accuracy. When introducing update, our simultaneous iterative compression can further boost the compression ratio, finally reduce model size by 43%. Similar experimental results are also obtained on TIMIT. Both results are gained with slight accuracy loss.

Keywords: Speech recognition · SVD-based compression · Loss sensitive update · Simultaneous iteration

1 Introduction

In the past few years, we have witnessed a rapid development of deep neural networks in the field of automatic speech recognition (ASR) [8,14,18,21,22]. However, the large size of neural network models leads to high computation and memory costs, which also makes it difficult to deploy the models in low resource devices. Frequently-used solution is to put the models on powerful cloud servers. But when network-connection is instable, this approach brings high latency, and even failure. Thus, neural networks compression for mobile devices attracts more and more attention.

Recent researches have proposed various methods to compress models, which can be efficiently executed directly on the embedded devices. Existing methods for neural network compression can be broadly divided into four categories: parameter quantization, pruning, knowledge distillation and low rank approximation.

Parameter quantization attempts to quantize the weights or activations of networks from 32 bit floating point into lower bit-width representations. With just two-three bits per parameter, these methods can get pretty good compression performance [6,12,15,23]. However it requires the algorithm computationally efficient while reducing runtime memory footprint.

Pruning is a forthright way to reduce network complexity. [11] pioneered the approach of pruning. They trained a full network and removed the neurons with the zero activations. The work in [20] exploited the sparseness in DNN, and presented a nice way to reduce the model size. [4] jointly learned weights and connections, using a hard threshold to remove the least important weights with small absolute values. Finally, they then fine-tuned to recover its accuracy. It has successfully pruned the heavy networks without performance loss. But it still need extra memory usage to index the non-zero value.

Knowledge distillation method first trains a heavier network, as "teacher" network, then trains a smaller "student" network through knowledge transfer. First attempts in this direction were made by [2], they investigated the model complexity- RMSE error. [5] then utilized the predicted probability distribution of the teacher model as "knowledge", introducing a more general technique for distilling the knowledge of a network.

Low-rank approximation is also widely studied [3,17,19]. In recent years, low-rank tensor approximation methods, e.g. Singular Value Decomposition (SVD), have been established as a new tool in matrix compression to address large-scale parameters problem. Reducing parameter dimensions by low-rank approximation saves storage and reduces time complexity simultaneously.

Our work builds on previous research in the area of low rank decomposition, called ***Simultaneous Iterative SVD Reconstruction via Loss Sensitive Update*** **(SISVD-LU)**. Initially, a large model trained without constraints is produced. We conduct the first phase of our method to learn the importance of each weight matrix in different layers. We keep the essential information remained (indicated by larger singular values), and surpress less useful ones. Then, we update the reconstructed network in a optimal procedure so that the removed information can be compensated. In the end, reconstruction and update are iteratively performed to further reduce network complexity and keep the accuracy at a acceptable level.

Our work is different from the previous works in follow aspects:

1. Most methods [3,7,19] approximate a tensor by minimizing the reconstruction error of the original parameters, while ignoring the accumulate errors. Our update mechanism emphasizes the ultimate network objective goal by

applying across-layer loss sensitive update. Furthermore, we iterate the process which is different from their methods.
2. Compared with [13], we compress the Time Delay Neural Network (TDNN) for ASR instead of Convolutional Neural Network (CNN). We also aware the importance of global loss, but we further explore the case of single-layer compression, and propose our exclusive update.
3. Prior approaches are usually evaluated on over-parameterized models, such as AlexNet [10], VGG [16], or very large output full-connection layers. Our method can get about 50% compression rate while only applied in the relatively small hidden layers.

The rest of this paper is organized as follows. Section 2 details every phase of our proposed method. Experimental results are presented and discussed in Sect. 3. In Sect. 4, we summarize our work.

2 Effective SVD-Based Deep Network Compression

Our proposed method, called Simultaneous Iterative SVD Reconstruction via Loss Sensitive Update (SISVD-LU), including three phases:

2.1 Inner-Layer Reconstruction Using SVD

In our proposed method, a full-trained deep neural network model is firstly obtained without resource constraints. Then, we decompose the weights matrix $W^{(l)}$ between the l-th and $(l+1)$-th layers via matrix factorization (MF) to reduce the parameter size.

We formulate *the Matrix Approximation* problem as follow.

$$W = \hat{W} + \varepsilon \tag{1}$$

as Eq. 1 shows, the weight matrix W is subject to the *Additive-Residual* model. Where the weight matrix $W^{(l)}$ is generalized as W $(\in \mathbb{R}^{M\times N})$ with rank r, $\varepsilon \in \mathbb{R}^{M\times N}$ is the reconstructed residual matrix. And $\hat{W}$ is the approximate low-rank matrix. We can view this procedure as capturing the main patterns of W while eliminating much of "noise".

We use Singular Value Decomposition (SVD) to solve this rank minimization problem. The matrix $W^{(l)}$ has a representation of the form:

$$W^{(l)} = U\varSigma_r V^\top \tag{2}$$

where U and V are orthogonal matrices $UU^T = VV^T = I$, and $\varSigma_r$ is a diagonal matrix, $\varSigma_r = diag(\sigma_1, \sigma_2, ..., \sigma_r)$, are called *singular values*. The size of the original matrix $W^{(l)}$ is $M \times r$. The resulting decomposition submatrix U, $\varSigma_r$, V size $M \times r$, $r \times r$, $r \times N$, respectively. Here r denotes the number of the nonzero singular values.

It is found that the singular value decreases particularly fast. In many cases, the sum of top 10% of singular values accounts for more than 99% of the sum of all singular values [19]. For compression task, a small preserved rank $k$will be chosen. We sort singular values in a descending order and pick the largest k ($k \ll r$) singular vectors in U and V with corresponding eigenvalue in Σ to approximate W.

$$\begin{aligned} \hat{W} &= \hat{U}\hat{\Sigma}_k\hat{V}^\top \\ &= \hat{U}\hat{\Sigma}^{\frac{1}{2}} \cdot (\hat{V}\hat{\Sigma}^{\frac{1}{2}})^\top \end{aligned} \tag{3}$$

The approximation of SVD is controlled by the decay along the eigenvalues in Σ_k. This procedure changes the number of parameters from $M \times N$ to $k \times (M+N)$. So the *Compression Ratio* $\mathcal{R}$ is defined as $\mathcal{R} = \dfrac{k \times (M+N)}{M \times N}$.

The demonstration of SVD is presented in the right part of Fig. 1.

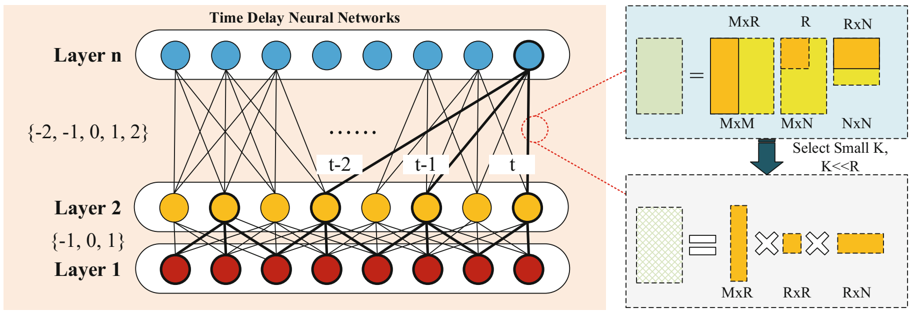

Fig. 1. Architecture of SVD-based network reconstruction. Left part is the baseline TDNN model structure, right part illustrates the process of SVD.

2.2 Across-Layer Loss Sensitive Update

From a across-layer perspective, the inner-layer decomposition causes cumulative errors and destroys the coupling of the layers. We build the *Loss-sensitive Update* recipe.

In a vanilla neural network, the input features are presented as $X = [x_1, x_2, ..., x_m]$, where $x_i \in \mathbb{R}^d$, where m is the number of feature vectors and d is dimension of a feature vector. After forward propagation, the output of the l-th layer can be written as:

$$y_i^{(l)} = \sigma(z_i^{(l)}),\ where\ z_i^{(l)} = \sum_{j=1}^{n^{(l-1)}} w_{ij}^{(l-1)} x_j^{(l-1)} \tag{4}$$

Where $w_{ij}^{(l-1)}$ is the element of weight matrix $W^{(l-1)}$. $n^{(l-1)}$ denotes the number of neurons in the $l-1$ layer. And $\sigma(\cdot)$ is a non-linear transformation called "activation function". The commonly used forms of this function are: tanh, sigmoid or the rectifier linear unit (ReLU) etc. The hidden state and its result after activation are denoted as vectors $z_i^{(l)}$ and $y_i^{(l)}$, respectively. Expanding the expression to Eq. 5.

$$\mathrm{y}_i^{(l)} = \sigma(w^{(l-1)} \cdots \sigma(w^{(2)}\sigma(w^{(1)}x^{(1)} + b^{(1)}) + b^{(2)}) \cdots + b^{(l-1)}) \tag{5}$$

We can see more clearly how the global error accumulated after decomposition. Most existing reconstruction focus on how to reduce the error of inner-layer reconstruction, as showed in Eq. 6. Here $||\cdot||_F$ as Frobenius norm. In this way, the loss of global accuracy is often ignored.

$$\mathcal{C}_1 = \min_{\hat{W}^{(l)}} \frac{1}{2}\left\| W^{(l)} - \hat{W}^{(l)} \right\|_F^2 \tag{6}$$

Single-layer reconstruction weakens the strong associations between layers, which are built through forward and backward propagation. Hence, we solve the reconstruction problem for a broader scope, aiming at preserving the global modeling capabilities of networks, such as classification ability or regression ability. Our loss function is modeled as Eq. 7.

$$\begin{aligned}\mathcal{C}_2 &= \min_{\hat{W}^{(l-1)}} \frac{1}{2}\left\| Y^{(l)} - \sigma(\hat{W}^{(l-1)}X) \right\|_F^2 \\ &= \min_{\hat{w}_{ij}{}^{(l-1)}} \sum_{i=1}^{n^{(l)}} \sum_{j=1}^{n^{(l-1)}} \frac{1}{2}\left\| y_i^{(l)} - \sigma(\hat{w}_{ij}^{(l-1)}x_j) \right\|_2^2\end{aligned} \tag{7}$$

In order to further constrain the complexity of the model, we add the L1-regularization term to the objective function, inducing model to be sparse. Our final objective function is Eq. 8.

$$\begin{aligned}\mathcal{L} &= \min_{\hat{W}^{(l-1)}} \frac{1}{2}\left\| Y^{(l)} - \sigma(\hat{W}^{(l-1)}X) \right\|_F^2 + \Psi_\lambda(\hat{W}) \\ s.t. \quad & \Psi_\lambda(\hat{W}) = \lambda \sum_{i=1}^{n^{(l)}} \sum_{j=1}^{n^{(l-1)}} ||\hat{w}_{ij}||_1\end{aligned} \tag{8}$$

Then we backpropagate loss using *Stochastic Gradient Descent* (SGD). Note that SVD will insert a bottleneck layer in the middle of the original layer. In backpropagation phase, we keep the structure.

According to the different scopes of global loss backpropagation, we propose the following methods:

- **Scheme 1** Fix the decomposition layers, only update the remaining layers, we call it *Exclusive Update.*

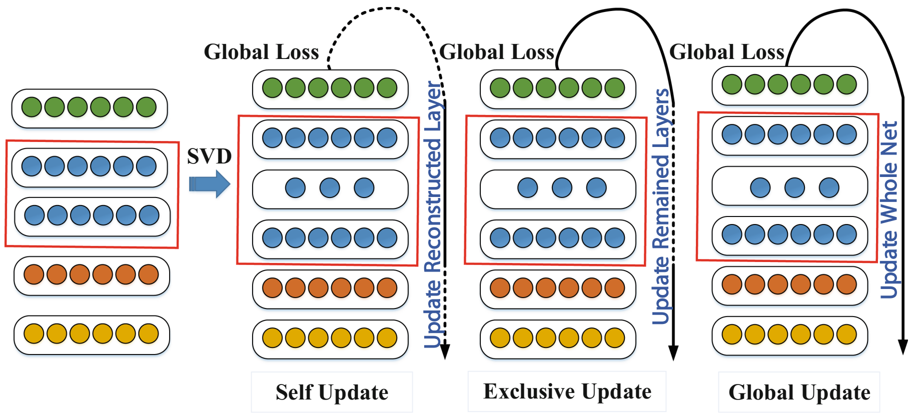

Fig. 2. Different update schemes of global loss backpropagation. Here, the rounded rectangle represents a hidden layer, and the circle represents neuron. The dashed line indicates that backpropagation does not change the parameters of the specific layers, and the solid line indicates that backpropagation will update the parameters those layers.

- **Scheme 2** Only update decomposition layers, keep the remaining layers unchanged, we call it *Self Update*.
- **Scheme 3** Update the whole reconstructed model, we call it *Global Update*.

The range of error back-propagation is controlled by the learning rate of each layer. If the learning rate is set to zero, this layer parameter is not updated. Demonstration of different global loss backpropagation schemes is presented in Fig. 2.

2.3 Iterative Compression

As previous section summarized, across-layer reconstruction can utilize SVD-based compression performance to make the neural networks small and fast enough. Based on the above analysis, we find that iteratively apply inner-layer decomposition and across-layer reconstruction procedure will bring high compression ratio with low accuracy loss. We perform the iterative compression in two different ways:

- ***Layerwise Compression***: Conduct network reconstruction after compression of single layer during every iteration.
- ***Simultaneous Compression***: Compress the whole networks at same time, and fine-tuning follows iteratively.

The back-propagation of the cumulative errors can retrieve the discriminal ability damage of models and preserve original relationships across the networks. Our proposed method reduces network complexity and keep the accuracy in a acceptable level. Figure 3 describes two iterative compression flows.

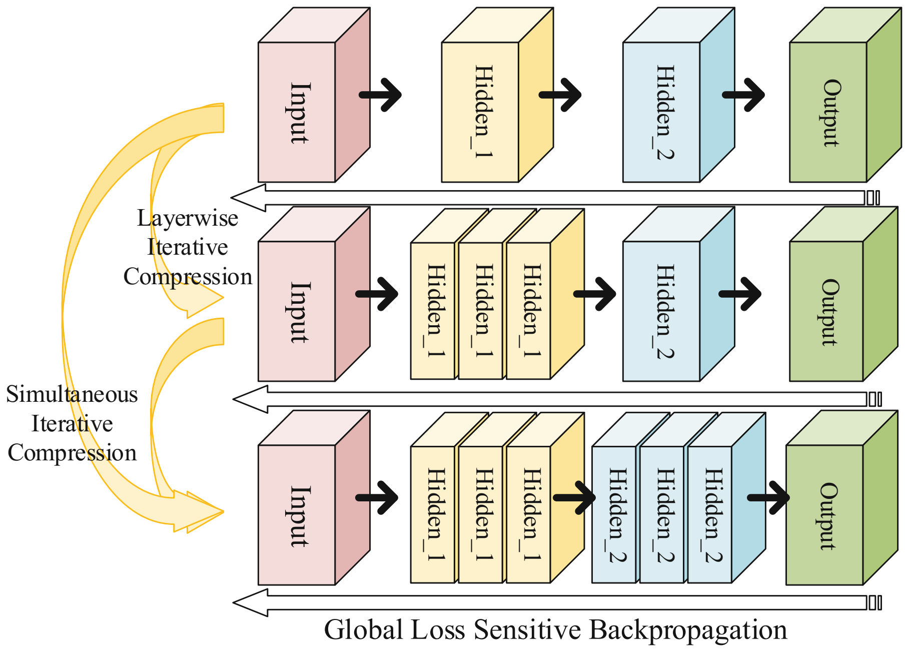
Input
Hidden_1
Hidden_2
Output
Layerwise Iterative Compression
Simultaneous Iterative Compression
Global Loss Sensitive Backpropagation

Fig. 3. Two iterative compression approaches: simultaneous compression, layerwise compression. Layerwise compression will get the bottom state from top step by step. Simultaneous compression will go to the bottom state directly. Here, the cube represents the weight matrix, it is decomposed into three smaller matrices by SVD.

3 Experiments

In this section, we evaluate the effectiveness of our approach on two different LVCSR corpus, AISHELL [1] for Mandarin ASR and TIMIT for English ASR.

3.1 Experimental Setup

Architecture of TDNN. Since speech signal has the temporal dynamics property, an acoustic model is required to have the ability to capture the long term dependencies between acoustic events. In a standard DNN, the initial layer learns the entire temporal context. Whereas, the TDNN architecture learns in a hierarchical structure. Narrow contexts are learnt by low layers and the deeper, layers learn from a wider temporal context. Hence the higher layers have the ability to learn wider temporal relationships. The structure of TDNN is depicted in the left part of Fig. 1.

3.2 Mandarin LVCSR Task on AISHELL

AISHELL corpus is a 170-h Mandarin speech corpus [1]. The corpus includes training set (150 h), development set (10 h) and test sets (5 h).

The input features consist of two parts, including 13-dimensional Mel frequency cepstral coefficients (MFCC) and 3-dimensional pitch features. Mean normalization and double deltas are applied on the above features before feeding into the training pipeline. The resulting GMM-HMM model has 2952 senones. During the training of TDNN-based acoustic models, we input high resolutional (40-dimensional) MFCC and 3-dimensional pitch features. Audio augmentation [9] and i-Vector (100-dimensional) based DNN adaptation are applied. Subsampling window is applied on MFCC and pitch features to splice neighboring frames.

Our baseline system is constructed based on the corresponding recipe. We used time delay neural network (TDNN) as our baseline. In our experiments, it contains 6 hidden layers, 850 hidden nodes per layer, using ReLU as the activation function. The output layer consisted of 2952 units.

After we obtain a full-trained (train with no resource constraints) deep neural network model, further reduction of footprint is conducted by an SVD-based compression.

Rank Selection. We notice that rank selection affects the compression rate as well as the accuracy. Too high rank will result in insufficient compression, while too low may make the accuracy recovery difficult or impossible.

To explore the implicit information each layer contains, we apply SVD on the same layer with different ranks, and different layers with the same rank.

Tables 1 and 2 summarizes the experimental results. The first column describes the setup of the model, and the number in bracket means that how many singular values we keep after SVD. The third column is the number of parameters in each model. For example, in the original DNN model the number of parameters is $315 \times 850 + (850 \times 850) \times 6 + 850 \times 2952 \approx 6.78M$. Baseline GMM model has $100K$ gaussians in total.

Table 1. Results for SVD reconstruction **on the same single layer** preserving **different ranks** in AISHELL task (without fine-tune). Numbers inside brackets represent the preserved singular values. The digits in the table represent the **word error rate** (WER) (%). NoP denotes number of parameters

Acoustic model	test	dev	NoP
6th(128)	10.08	8.33	0.21M
6th(256)	8.62	7.4	0.42M
6th(512)	8.49	7.28	0.83M

From Table 1, we can see that model size of our original DNN model is nearly twice as GMM model. We reduce WER at 30% relatively by replacing GMM model with DNN model. Using different preserved ranks in the same layer lead to a nonlinear loss of precision.

Table 2. Results for SVD reconstruction **on different layers** preserving **the same rank** in AISHELL task (without fine-tune)

Acoustic model	test	dev	Prior NoP	Post NoP
GMM	12.10	10.40	- - -	- - -
TDNN	8.45	7.20	6.78M	- - -
2nd hidden layer(256)	9.99	8.47	0.68M	0.41M
3rd hidden layer(256)	8.87	7.62	0.68M	0.41M
4th hidden layer(256)	8.97	7.6	0.68M	0.41M
5th hidden layer(256)	8.67	7.42	0.68M	0.41M
6th hidden layer(256)	**8.62**	**7.4**	**0.68M**	**0.41M**
Output layer(256)	14.01	13.84	2.39M	0.92M
All hidden layers(256)	**22.12**	**19.58**	**6.78M**	**3.89M**

The following rows in Table 2 reveal the effect of the proposed approach. When we keep only top 50% singular values (the SVD-256 case) on the matrices of hidden layers, there are slight changes compared with original model. But when it comes to compress the output layer, keep top 50% singular values will cause obvious performance reduction. Therefore, different scaled weight matrices should keep different ranks.

As it were, SVD in the most sparse layer can effectively compress model size, even without reconstruction update. We can at most gain 50% compression ratio after applying SVD, while maintaining almost the same accuracy.

Across-Layer Loss Sensitive Update. We conduct several experiments, the results are described in Table 3. For single layer decomposition, the *Self Update* scheme and the *Global Update* scheme both work good for single-layer (or a small amount of layers). For multi-layer decomposition, *Global Update* scheme shows more advantages. Furthermore, we explore if train for more epoches, the *Self Update* can get better results. But it leads to a painful long time consumption, and doesn't look better than *Global Update* case (result: WER on dev set is 7.35%, on test set is 8.61%, re-train for 3 epoch,).

We believe that it is owing that the cumulative losses from SVD can affect the entire network, *Global Update* is more reasonable to give reconstructed model a larger adjustable range. On the other hand, since SVD maintains the principal components of the original weight matrix, we can take it for a pre-training procedure.

As a periodic summary, *Global Update* is suitable for more scenarios.

Iterative Compression. We perform experiments to compare our two iterative compression methods. Here "Aggressive" means directly push model to a relatively small size and Global Update repeatedly. "Gradual" mode means

Table 3. Comparisons of different update schemes after weight matrices reconstruction in AISHELL task

Acoustic model	Fine-tune Scheme	WER	
		test	dev
TDNN, Baseline	- - -	8.45	7.20
6th Hidden Layer(256)	Exclusive Update	8.59	7.36
	Self Update	**8.49**	**7.31**
	Global Update	8.55	7.36
All Hidden Layers(256)	Exclusive Update	9.45	8.08
	Self Update	8.37	7.22
	Global Update	**8.33**	**7.14**

compress to a moderate size first and iterative push to a smaller size, fine-tune is executed after every iteration. Table 4 shows our comparison results.

Table 4. Comparisons of different iterative compression schemes for combination of reconstruction and update in AISHELL task

Iterative scheme	Mode	test	dev
Layerwise compression	**Input to Output**	**8.59**	**7.36**
	Output to Input	8.63	7.41
Simultaneous compression	**Aggressive(850-128)**	**8.49**	**7.31**
	Gradual(850-512-256-128)	8.55	7.36

From the exhibited results, we found that the *Simultaneous Compression* looks better than *Layerwise Compression*. Moreover, proceeding from input to output rather that the reverse order produces better results for layerwise compression. "Agressive" mode for Simultaneous compression shows more effectiveness.

Results. Here we summarize the best result on AISHELL in Table 5 with SISVD-LU.

We get the best result when we factorize all hidden layers with rank 256 and iterative global update. The parameter size of the final compressed model is $315 \times 850 + (850 \times 128 + 128 \times 850) \times 6 + 850 \times 2952 \approx 3.89M$, which is 57% of the original size.

3.3 English LVCSR Task on TIMIT

We demonstrate scalability of the proposed low-rank decomposition on a different dataset. So our second task is to train a same-structure TDNN model for English ASR corpus TIMIT to verify our conclusion is universal.

Table 5. Experimental results on the Mandarin AISHELL Corpus using our SISVD-GLU approach

Acoustic model		test	dev
TDNN, Baseline	- - -	8.45	7.20
All Hidden Layers(512)	Before Iterative Update	8.54	7.32
	After Iterative Update	8.42	7.21
All Hidden Layers(256)	Before Iterative Update	22.12	19.58
	After Iterative Update	**8.32**	**7.12**
All Hidden Layers(128)	Before Iterative Update	34.55	30.71
	After Iterative Update	8.49	7.31

The two corpora are significantly different in language and duration. TIMIT contains a total of 6300 sentences (5.4 h), consisting of 10 sentences spoken by each of 630 speakers from 8 major dialect regions of the United States.

In our experiments, we use 13 dimensional features space maximum likelihood linear regression (fMLLR) features and then concatenate the neighboring 5 frames (11 frames in total) as the input feature. Note that, we don't use i-Vector in following experiments. To have a fair comparison, we construct a same TDNN architecture (number of hidden units and number of layers are the same) as the one used on last task.

We first train a full-trained model and then perform SISVD-LU. Table 6 shows us the results.

Table 6. Experimental results on the English TIMIT corpus using our SISVD-GLU approach

Acoustic model		test	dev
TDNN, Baseline	- - -	18.4	16.4
All Hidden Layers(512)	Before Iterative Update	18.7	17.4
	After Iterative Update	18.4	16.9
All Hidden Layers(256)	Before Iterative Update	23.3	21.4
	After Iterative Update	17.8	16.2
All Hidden Layers(128)	Before Iterative Update	45.2	41.5
	After Iterative Update	**17.7**	**16.1**

After compression, the accuracy after compression suffers great reduction. Yet, our iterative global update can recall the loss back. Those results fully support our method.

4 Conclusion

In this paper, we have proposed an effective SVD-based compression method. The loss sensitive update has conducted after SVD reconstruction, and repeat this combination of two operations. For the single layer, by performing our compression method, we can gain 50% compression ratio after applying SVD while maintaining almost the same accuracy. For a whole model, our iterative update procedure can boost the compression ratio, in the same time, without accuracy loss. We verify our strategies in two very different datasets, TIMIT for English ASR and AISHELL [1] for Mandarin ASR.

The experimental results support our conclusion. Though we only investigate the SVD compression method, the outcome of this paper provokes us to generalize our conclusion in other matrix manipulation related methods or combine it with other compression methods.

References

1. Bu, H., Du, J., Na, X., Wu, B., Zheng, H.: Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline. arXiv preprint arXiv:1709.05522 (2017)
2. Bucilu, C., Caruana, R., Niculescu-Mizil, A.: Model compression. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 535–541. ACM (2006)
3. Denton, E.L., Zaremba, W., Bruna, J., LeCun, Y., Fergus, R.: Exploiting linear structure within convolutional networks for efficient evaluation. In: Advances in Neural Information Processing Systems, pp. 1269–1277 (2014)
4. Han, S., Pool, J., Tran, J., Dally, W.: Learning both weights and connections for efficient neural network. In: Advances in Neural Information Processing Systems, pp. 1135–1143 (2015)
5. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
6. Jacob, B., et al.: Quantization and training of neural networks for efficient integer-arithmetic-only inference. arXiv preprint arXiv:1712.05877 (2017)
7. Jaderberg, M., Vedaldi, A., Zisserman, A.: Speeding up convolutional neural networks with low rank expansions. arXiv preprint arXiv:1405.3866 (2014)
8. Kim, S., Hori, T., Watanabe, S.: Joint CTC-attention based end-to-end speech recognition using multi-task learning. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4835–4839. IEEE (2017)
9. Ko, T., Peddinti, V., Povey, D., Khudanpur, S.: Audio augmentation for speech recognition. In: Sixteenth Annual Conference of the International Speech Communication Association (2015)
10. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
11. LeCun, Y., Denker, J.S., Solla, S.A.: Optimal brain damage. In: Advances in Neural Information Processing Systems, pp. 598–605 (1990)
12. Li, F., Zhang, B., Liu, B.: Ternary weight networks. arXiv preprint arXiv:1605.04711 (2016)

13. Lin, S., Ji, R., Guo, X., Li, X., et al.: Towards convolutional neural networks compression via global error reconstruction. In: IJCAI, pp. 1753–1759 (2016)
14. Maas, A.L., et al.: Building DNN acoustic models for large vocabulary speech recognition. Comput. Speech Lang. **41**, 195–213 (2017)
15. Rastegari, M., Ordonez, V., Redmon, J., Farhadi, A.: XNOR-Net: imagenet classification using binary convolutional neural networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 525–542. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_32
16. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
17. Sindhwani, V., Sainath, T., Kumar, S.: Structured transforms for small-footprint deep learning. In: Advances in Neural Information Processing Systems, pp. 3088–3096 (2015)
18. Xiong, W., et al.: The microsoft 2016 conversational speech recognition system. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5255–5259. IEEE (2017)
19. Xue, J., Li, J., Gong, Y.: Restructuring of deep neural network acoustic models with singular value decomposition. In: Interspeech, pp. 2365–2369 (2013)
20. Yu, D., Seide, F., Li, G., Deng, L.: Exploiting sparseness in deep neural networks for large vocabulary speech recognition. In: 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4409–4412. IEEE (2012)
21. Zeyer, A., Doetsch, P., Voigtlaender, P., Schlüter, R., Ney, H.: A comprehensive study of deep bidirectional LSTM RNNs for acoustic modeling in speech recognition. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2462–2466. IEEE (2017)
22. Zhang, Y., Chan, W., Jaitly, N.: Very deep convolutional networks for end-to-end speech recognition. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4845–4849. IEEE (2017)
23. Zhou, A., Yao, A., Guo, Y., Xu, L., Chen, Y.: Incremental network quantization: Towards lossless CNNs with low-precision weights. arXiv preprint arXiv:1702.03044 (2017)