Densely Connected Multi-Layer Perceptron

Jinsung Yoon
20242376

Jaehwan Lee
20242318

Jemin Jeon
20180154

Abstract

In order to properly emulate real neurons, we introduce densely-connected multi-layer perceptron, namely DCMLP, using concat-based skip connections. In this paper, we study the effects of skip connections on various skip range restrictions and pruning by analyzing test accuracy and weight sparcity. Experiments on FashionMNIST and VoxCeleb show that although shallow layer models do not benefit from skip connections, deeper models benefit from skip connections, typically on long-range skip connections.

1 Introduction

The multi-layer perceptron (MLP), which mimics connections between neurons, has brought significant advancements in deep learning. However, MLP falls short of faithfully emulating neurons. MLP typically transmits information from the preceding layer to the adjacent next layer. However, neurons not only relay information to adjacent neurons but also interact with neurons in distant regions. In this paper, we study modeling this behavior of neurons in multi-layer perceptron architecture and analyze its feasibility. We formulate the inter-connection behavior of neurons via skip-connection. Studies on skip or residual connections in a neural network have been conducted and showed the effectiveness of these additional flows on the network's performance[3][4]. Through this, we propose a more generalized densely connected MLP architecture, namely Densely Connected Multi-layer Perceptron(DCMLP). Our DCMLP network involves two variations; (1) neurons not only convey information from the preceding layer to the adjacent next layer but also extends information exchange to distant layers, and (2) pruning is employed to curtail parameter count while reinforcing connections with different neurons. By calculating the sparsity of pruned weights, we can analyze the importance of specific connections including non-skip connections. Additionally, we investigate the performance under various skip-layer restrictions(maximum number of layers a node can skip) and find the optimal condition.

2 Method

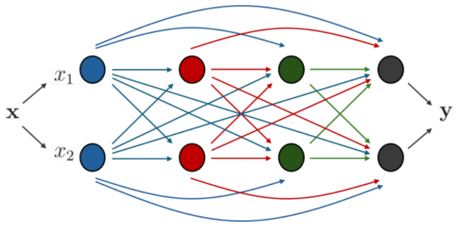
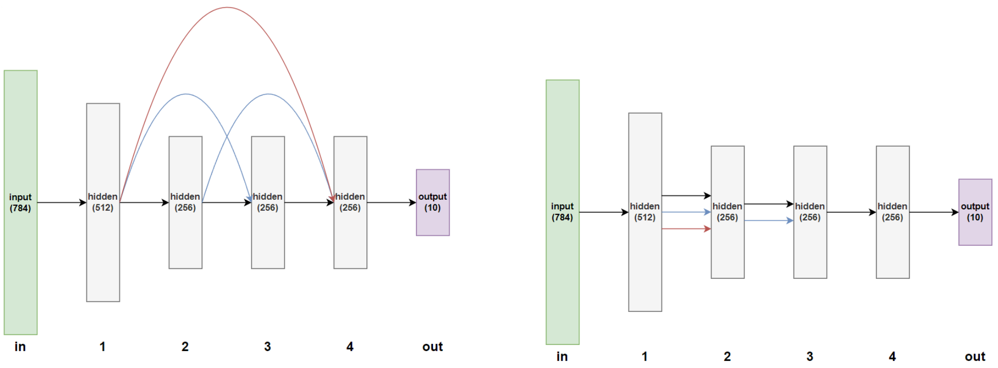


Figure 1: DCMLP model architecture

DeepLearning (CSED/AIGS 538): Final Report

$$\begin{aligned}
\mathbf{H}_1 &= f(\mathbf{W}_{11}\mathbf{x} + \mathbf{b}_1), & (1)\\
\mathbf{H}_2 &= f(\mathbf{W}_{22}\mathbf{H}_1 + \mathbf{W}_{21}\mathbf{x} + \mathbf{b}_2) & (2)\\
\mathbf{H}_3 &= f(\mathbf{W}_{33}\mathbf{H}_2 + \mathbf{W}_{32}\mathbf{H}_1 + \mathbf{W}_{31}\mathbf{x} + \mathbf{b}_3) & (3)\\
\mathbf{y} &= \mathbf{W}_4\mathbf{H}_3. & (4)
\end{aligned}$$

Figure 1. shows the concept of DCMLP architecture. For simplicity, the network has 2 hidden layers. While nodes of one layer are densely connected to the following layer, additional connections are applied to connect to subsequent layers' nodes. Equations (1) to (4) denote the example network architecture. $\mathbf{x}$, $\mathbf{y}$ denotes the input and output respectively, $\mathbf{H}_i$ the output of i-th hidden layer, $\mathbf{W}_{ij}, i \in \{1, ..., m\}, j \in \{1, ..., i\}$ the weight matrix of i-th hidden layer multiplied to the output of $(j-1)$-th layer. $f(\cdot)$ the ReLU activation function. To find the optimal model structure for DCMLP, we set a 'range' hyperparameter. The range denotes the maximum number of layers a skip connection can span. For instance, if the range r=1, it corresponds to a typical MLP. If r=2, skip connections extend up to the second next layer. Figure 2. shows the experimental group, DCMLP, with the control group, MLP. As shown in Figure 2(a), the DCMLP, blue lines represent skip connections that skip one layer whose range is 2, and red lines skip two layers whose range is 3. We keep the parameters of models same to remove the effect of variance in number of parameters. To maintain the parameters, we modified the base MLP structure by adding duplicate connections as shown in Figure 2(b).



(a) (b)

Figure 2: Architecture of DC-MLP and MLP on various ranges

2.1 Skip Connection

Concatenation-based skip connection preserves the integrity of features compared to simple summation[4]. Also, it can inject different importance to these skip connections which accord with our objective. Thus, we apply connections in a way shown in equation (5) which is slightly different from that of DenseNet. However, the connection behavior is conducted in the same way. Here, $\mathbf{x}_i$ refers to the feature produced in layer i, and H refers to non-linear transformation. Separating weight matrices linked to input features and features from previous layers gives ease to analyzing weight sparsity.

$$\mathbf{x}_\ell = \sum_{i=0}^{\ell-1} H_{\ell,i+1}(\mathbf{x}_i) \tag{5}$$

2.2 Pruning

To reduce the increased number of parameters due to the added skip connections and to determine the importance of the connections, we used importance-based pruning. We partially adopted the pruning method from [2]. The pruning operates within the optimization cycle, removing parameters with low importance from the model. Importance is defined as the magnitude of each parameter in the given epoch. We adopted a global pruning approach, allowing less important layers to be pruned

	FashionMNIST	VoxCeleb
MLP	90.08	91.76
MLP r=2	89.99	92.39
DCMLP r=2	89.97	91.92
MLP r=2 (prune)	90.21	92.11
DCMLP r=2 (prune)	90.14	92.03
MLP r=3	90.00	92.13
DCMLP r=3	89.99	91.87
MLP r=3 (prune)	90.16	92.42
DCMLP r=3 (prune)	90.11	91.85

Table 1: Test accuracy(%) of shallow models

	FashionMNIST	VoxCeleb
DeepMLP	10.00	16.15
DeepMLP r=11 3 conn. (prune)	10.00	16.15
DeepDCMLP r=11 3 conn. (prune)	89.57	90.66
DeepDCMLP 1-to-any (prune)	89.57	90.99
DeepDCMLP any-to-13 (prune)	89.52	90.92

Table 2: Test accuracy(%) of deep models

more extensively. Additionally, we performed pruning four times during the optimization process. The pruning rates at each stage were [10, 40, 60, 80], respectively. Consequently, only 20% of the parameters in the prunable layers remain after training is completed.

3 Experiments

Experiments are conducted on two conditions; models with shallow layers(4 hidden layers) and with deep layers(13 hidden layers). Deep layer models are denoted as DeepMLP and DeepDCMLP. Both conditions were experimented with FashionMNIST and VoxCeleb[7], with test accuracy and sparcity of weights calculated for each.

3.1 Experiment Setup

For training dataset, we used VoxCeleb1 identification split for speaker identification task and FashionMNIST for image classification task. For VoxCeleb, we cherry-picked 10 speakers among 1251. 80 dimension log mel coefficients were used for input features. The Fashion MNIST dataset consists of 10 categories of fashion images. Each image is composed of 28x28 pixels, and the training dataset contains 60,000 images, while the test dataset contains 10,000 images Models were trained with 32 batch size, SGD with 0.5 momentum, and learning rate of 1e-2. VoxCeleb and FasionMNIST were trained for 50 epochs and 100 epochs respectively. For shallow model, models consist of 4 hidden layers followed by ReLU activations with dimension sets (512, 256, 256, 256) except for the input layer(784 for image, 80 for speech). Skip connections are added only on hidden layers. For deep model, models consist of 13 hidden layers with 128 dimensions. Other configurations follow shallow model.

3.2 Results on Performance

Table 1. shows the test accuracy of trained shallow models. Naive MLP has parameters of 306,954. Models in below two sections separated by horizontal line have 504,074 and 635,402 parameters respectively. As shown in the table, all models observed similar performance regardless of skip connections. Besides, there was a slight decrease in performance when skip-connections were applied. Table 2. shows the test accuracy of trained deep models. DeepDCMLP with 3 connections(third row) has random 3 skip connections (2 to 10, 2 to 14, 6 to 14). The last two models have skip connections from first hidden layer to any other layers(1-to-any), and any hidden layers to last hidden layer(any-to-13). Unlike shallow models, deep models showed significant performance gap between naive MLP and DCMLP. DeepMLP with no skip-connections, and DeepMLP with 3 duplicate connections couldn't manage to converge from which we surmise that gradient vanishing occured, whereas DeepDCMLP showed decent performance although slight decrease compared to shallow models. This implies that skip connection is beneficial for deep models which complies to our prior knowledge.

3.3 Results on Sparsity

Further experiment on deep layer models were conducted and the results are shown in appendix B. Sparcity is measured by calculating the ratio of weight elements being 0 over total number of weights in a matrix.

	in $\rightarrow 1$	$1 \rightarrow 2(i)$	$1 \rightarrow 2(ii)$	$2 \rightarrow 3(i)$	$2 \rightarrow 3(ii)$	$3 \rightarrow 4$	$4 \rightarrow out$
FashionMNIST	80.89	**86.40**	**86.28**	**71.09**	**71.44**	68.32	**36.95**
VoxCeleb	39.68	**91.47**	**91.54**	**74.79**	**74.82**	71.36	**34.34**

Table 3: Sparcity(%) of weights of MLP r=2

	in $\rightarrow 1$	$1 \rightarrow 2$	$1 \rightarrow 3$	$2 \rightarrow 3$	$2 \rightarrow 4$	$3 \rightarrow 4$	$4 \rightarrow out$
FashionMNIST	81.22	**82.38**	**89.60**	**70.54**	**72.29**	67.66	**30.70**
VoxCeleb	40.15	**87.75**	**94.04**	**75.19**	**75.86**	72.17	**31.88**

Table 4: Sparcity(%) of weights of DCMLP r=2

	in $\rightarrow 1$	$1 \rightarrow 2(i)$	$1 \rightarrow 2(ii)$	$1 \rightarrow 2(iii)$	$2 \rightarrow 3(i)$	$2 \rightarrow 3(ii)$	$3 \rightarrow 4$	$4 \rightarrow out$
FashionMNIST	79.08	**87.41**	**87.34**	**87.22**	**68.77**	**68.43**	66.23	**36.09**
VoxCeleb	36.52	**90.58**	**90.79**	**90.66**	**69.33**	**69.53**	66.03	**34.92**

Table 5: Sparcity(%) of weights of MLP r=3

	in $\rightarrow 1$	$1 \rightarrow 2$	$1 \rightarrow 3$	$1 \rightarrow 4$	$2 \rightarrow 3$	$2 \rightarrow 4$	$3 \rightarrow 4$	$4 \rightarrow out$
FashionMNIST	80.17	**82.47**	**88.30**	**87.89**	**69.09**	**69.11**	65.63	**23.87**
VoxCeleb	36.43	**86.62**	**91.93**	**91.23**	**70.71**	**70.83**	68.28	**25.04**

Table 6: Sparcity(%) of weights of DCMLP r=3

Table 3. through 6. shows the sparsity of weights in shallow layer models. In MLP when r is 3, the sparsity of the three layers connecting 1 to 2 is similar whereas in DCMLP, the sparsity from 1 to 2 is low, while 1 to 3 and 1 to 4 have higher sparsity. This indicates that skip connections were not as important as regular adjacent layer connections. Table 7. through 10. shows the sparsity of weights in the deep layer DCMLP scheme. Here, regular connections denote linear connections between adjacent layers. **F** stands for Fashion MNIST and **V** stands for VoxCeleb. As shown in Table 7. and 8., for DeepDCMLP with skips from the first hidden layer to any other hidden layers(1-to-any), regular connections show a consistent tendency of sparsity(84-85%) of weights in hidden layers 1 to 8 and start to decrease after layer 8. This holds for both datasets. Interestingly, similar tendency was also observed for skip connections. While skip-connections from layer 1 up to 8 showed sparsity of 84-85%, farther connections(9-13) showed more reduced sparsity. From our intuition that less sparsity implies less redundant weights, we conjecture that longer skip connections have more valuable weights. Table 9. and 10. shows sparsity of weights of DeepDCMLP with skips from any hidden layer to last hidden layer(any-to-13). Parallel to our previous conjecture, features from earlier layers contributed more to the model.

4 Conclusion

We experimented effects of concat-based skip connections and pruning on multi-layer perceptron architecture. Experiments show that in shallow layer models, skip connections are not beneficial whereas additional experiments on deep layer models show that skip connections are critical for performance. The results also prove our hypothesis that neurons not only utilize adjacent but also distant features. Consequently, our study introduces more generalized and flexible MLP architecture that uses dense skip-connections and pruning.

References

[1] Cortes, C., Gonzalvo, X., Kuznetsov, V., Mohri, M., Yang, S.: Adanet: Adaptive structural learning of artificial neural networks. In: International conference on machine learning. pp. 874–883. PMLR (2017)

[2] Han, S., Pool, J., Tran, J., Dally, W.: Learning both weights and connections for efficient neural network. Advances in neural information processing systems **28** (2015)

[3] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)

[4] Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4700–4708 (2017)

[5] Liu, H., Simonyan, K., Yang, Y.: Darts: Differentiable architecture search. arXiv preprint arXiv:1806.09055 (2018)

[6] Mostafa, H., Wang, X.: Parameter efficient training of deep convolutional neural networks by dynamic sparse reparameterization. In: International Conference on Machine Learning. pp. 4646–4655. PMLR (2019)

[7] Nagrani, A., Chung, J.S., Zisserman, A.: Voxceleb: a large-scale speaker identification dataset. arXiv preprint arXiv:1706.08612 (2017)

Appendices

A Related Works

[1] AdaNet propose an algorithm that adaptively learns network architecture by incrementally adding units and layers. Unlike our method, which gradually prunes the network, AdaNet focuses on expansion. [4] DenseNet introduces direct connections between layers to alleviate information loss and gradient vanishing. While DenseNet is CNN-based and uses growth rates to reduce parameters, our DC-MLP uses pruning and is MLP-based. [2] presents a three-step pruning technique to remove unimportant connections and retrain the remaining sparse network. Our approach differs by pruning connections between all layer pairs, not just adjacent ones. [5] DARTS proposes a gradient-based architecture search that optimizes network structure. Unlike DARTS, which searches within a predefined set of operations, our method prunes nodes without specifying operations beforehand. [6] introduces a technique for training sparse models directly, reallocating parameters throughout the process. This contrasts with traditional methods that start with dense models. Our approach similarly focuses on pruning but applies it to MLPs.

B Sparcity of weights in deep layer models

	in → 1	1 → 2	2 → 3	3 → 4	4 → 5	5 → 6	6 → 7	7 → 8	8 → 9	9 → 10	10 → 11	11 → 12	12 → 13	13 → *out*
F	79.58	84.91	84.58	84.75	85.21	85.26	84.63	85.03	83.61	**80.07**	**75.00**	**69.67**	**65.51**	29.92
V	47.52	83.85	84.34	84.27	84.61	84.51	83.97	84.30	83.94	**80.83**	**76.71**	**72.00**	**69.96**	31.25

Table 7: Sparcity(%) of DeepDCMLP 1-to-any regular connections

	1 → 3	1 → 4	1 → 5	1 → 6	1 → 7	1 → 8	1 → 9	1 → 10	1 → 11	1 → 12	1 → 13
F	85.10	84.58	85.14	85.11	84.89	84.08	**81.38**	**76.10**	**71.48**	**69.23**	**71.14**
V	84.65	84.13	84.14	84.55	84.66	84.28	**82.10**	**78.67**	**75.53**	**74.83**	**73.30**

Table 8: Sparcity(%) of DeepDCMLP 1-to-any skip connections

	in → 1	1 → 2	2 → 3	3 → 4	4 → 5	5 → 6	6 → 7	7 → 8	8 → 9	9 → 10	10 → 11	11 → 12	12 → 13	13 → *out*
F	79.18	**62.50**	**70.67**	**77.53**	**82.29**	84.65	83.65	84.01	84.69	83.97	84.05	84.44	84.41	30.39
V	48.03	**66.99**	**74.39**	**79.40**	**82.82**	83.46	83.29	83.66	83.43	83.22	83.72	83.61	83.50	33.52

Table 9: Sparcity(%) of DeepDCMLP any-to-13 regular connections

	1 → 13	2 → 13	3 → 13	4 → 13	5 → 13	6 → 13	7 → 13	8 → 13	9 → 13	10 → 13	11 → 13
F	**67.66**	**71.91**	**75.89**	**79.50**	**82.45**	83.74	83.91	84.25	84.05	84.57	84.10
V	**73.67**	**75.14**	**77.84**	**80.04**	**83.03**	83.47	83.98	83.45	83.94	83.85	83.73

Table 10: Sparcity(%) of DeepDCMLP any-to-13 skip connections