Supplementary Material for Straightforward Layer-wise Pruning for More Efficient Visual Adaptation

Ruizi Han and Jinglei Tang(✉)

College of Information Engineering, Northwest A&F University, Yangling, China
rzh@nwafu.edu.cn, tangjinglei@nwsuaf.edu.cn

1 More Experiment Details

Pre-trained Backbones. We conducted experiments utilizing the ViT-B/16 [2] and Swin-B [5] models built with the Timm [11] library, both of which were pre-trained on ImageNet21K [1].
Code Implementation. We employed PyTorch to conduct our primary experiments on an NVIDIA RTX-A5000 GPU and evaluated the model throughput using an NVIDIA RTX3090 GPU. The pytorch-like pseudo-code for calculating model throughput is as following Algorithm 1.
Data Augmentation. We resized the VTAB-1k [12] images to 224×224 and then normalized them using the mean and variance of the ImageNet, following [4, 7].
Training Details. Optimizer and hyper-parameters are shown in Tab. 2.

2 More Results

Detailed Results for Convpass. Pruning results for the Convpass [4] on VTAB-1k are depicted in Fig. 1, along with its corresponding model accuracy. The figure illustrates that an acceptable model accuracy can be maintained even with a significant number of pruned model structures for certain datasets, particularly SVHN, EuroSAT, and dspr-Loc. This finding aligns with the trend of SC_index [9] depicted in Fig. 2, which indicates a gradual decline in SC_index for both the Convpass and RepAdapter [7] on these datasets as the number of layers decreases.
Analysis of Silhouette Coefficient Index. Variation statistics on the relationship between the number of layers for Convpass and RepAdapter and SC_index values across all datasets are displayed in Fig. 2. The horizontal axis represents the number of layers within the model, while the vertical axis represents the SC_index computation from the current layer output features after t_SNE [8] dimensionality reduction. In each sub-figure, dashed lines represent the pruning thresholds of the two methods with respect to the current data, and pruning ceases at the point in which the number of layers falls below this threshold. The SC_index variation significantly differs for various datasets. Datasets

Algorithm 1: Calculating Model Throughput

```
# Define an input tensor
# x.shape = [B C H W]
x = torch.randn(64, 3, 224, 224)
batch_size = x.shape[0]
# Waits for all kernels to complete
torch.cuda.synchronize()
tic1 = time.time()
# Inference begin
for i in range(100):
    model(x)
torch.cuda.synchronize()
tic2 = time.time()
time = tic2 - tic1
# Throughput calculate
throughput = 100 × batch_size/time
```

Table 1: Comparison of different DepGraph [3] settings and SLS on VTAB-1k [12]. Pruning Adapter indicates that Depgraph prunes the Adapter parameters in the model while Ignoring Adapter means pruning only the pre-trained parameters in the model. †To maintain training stability, we reduced the learning rate in the Full retraining setting.

Pruning Method	Retraining Setting	Avg Acc.	Nat.	Spe.	Str.
DepGraph (Pruning Adapter)	Adapter+Head	74.5	79.3	82.3	61.8
DepGraph (Ignoring Adapter)	Full†	74.0	79.1	83.0	59.9
DepGraph (Ignoring Adapter)	Adapter+Head	75.0	79.7	83.4	61.9
SLS	Adapter+Head	75.4	79.9	84.5	61.9

with high initial values and a gradual decrease in the number of layers, for instance, SVHN and EuroSAT, demonstrate significantly better pruning performance compared to those with lower starting values, like Sun397, or datasets with a rapid decline such as Pets.

Results of Different DepGraph Settings Experiments were conducted to determine the effectiveness of the DepGraph [3] pruning method in pruning Adapter parameters and the retraining strategy. As DepGraph is a group pruning method, excluding certain modules could potentially affect the process of building the dependency graph and, in turn, impact the pruning results. The experimental outcomes are presented in Tab. 1. We observed a decrease in DepGraph's performance after pruning the adapter when compared to ignoring it. This outcome aligns with expectations, as pruning the adapter reduces the number of tunable parameters during the retraining process. The DepGraph with full retraining setting performed lower than the setting of using only the Adapter and Head, which may be due to overfitting caused by excessive parameters.

Table 2: Optimizer and hyper-parameters for 1-CLR [10] and TFS retraining strategies.

optimizer	batch size	learning rate	weight decay	epochs	lr decay	warm-up epochs
AdamW [6]	64	1e-3	1e-4	100	cosine	10

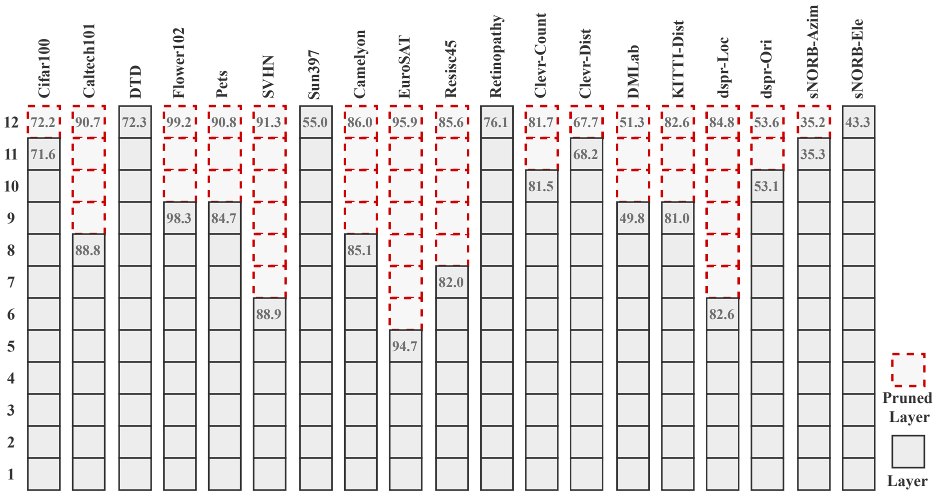

Fig. 1: The pruning outcomes of Convpass on VTAB-1K and the corresponding precision of the model before and after pruning.

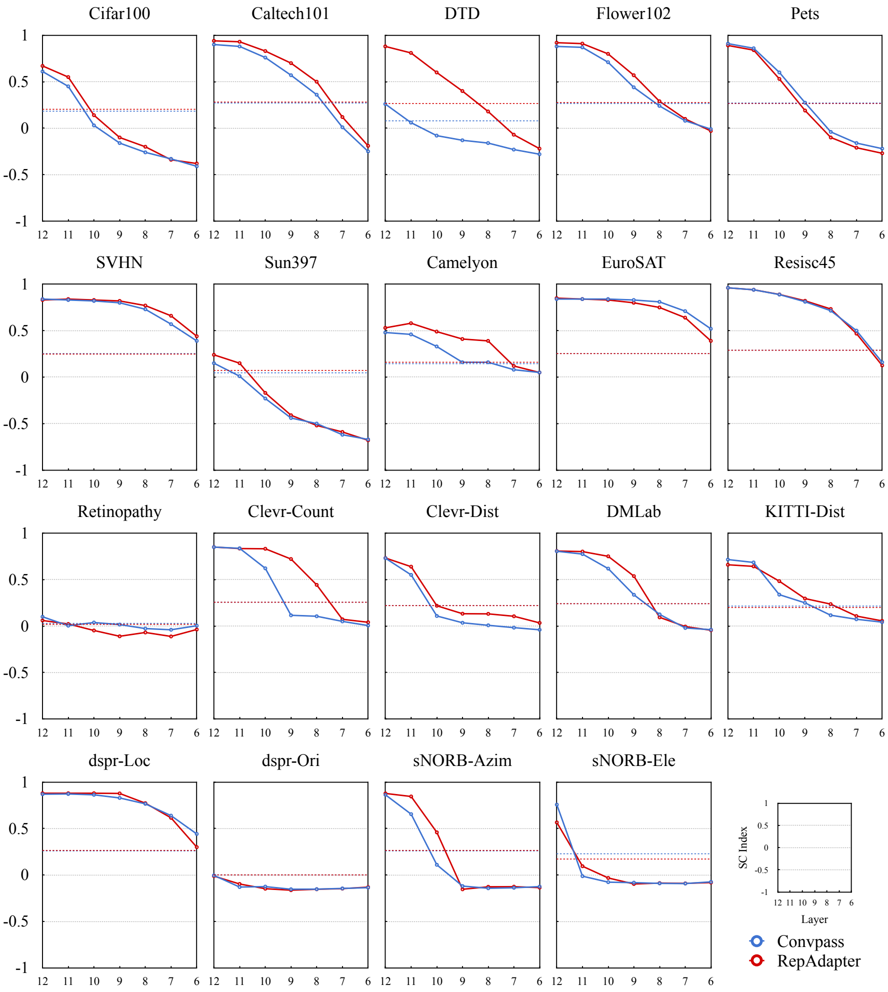

Fig. 2: The relationship between the number of layers for Convpass and RepAdapter and SC_index values on VTAB-1k.

References

1. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
2. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
3. Fang, G., Ma, X., Song, M., Mi, M.B., Wang, X.: Depgraph: Towards any structural pruning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16091–16101 (2023)
4. Jie, S., Deng, Z.H.: Convolutional bypasses are better vision transformer adapters. arXiv preprint arXiv:2207.07039 (2022)
5. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 10012–10022 (2021)
6. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
7. Luo, G., Huang, M., Zhou, Y., Sun, X., Jiang, G., Wang, Z., Ji, R.: Towards efficient visual adaption via structural re-parameterization. arXiv preprint arXiv:2302.08106 (2023)
8. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(11) (2008)
9. Rousseeuw, P.J.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of computational and applied mathematics **20**, 53–65 (1987)
10. Smith, L.N., Topin, N.: Super-convergence: Very fast training of neural networks using large learning rates. In: Artificial intelligence and machine learning for multi-domain operations applications. vol. 11006, pp. 369–386. SPIE (2019)
11. Wightman, R.: Pytorch image models. `https://github.com/rwightman/pytorch-image-models` (2019). `https://doi.org/10.5281/zenodo.4414861`
12. Zhai, X., Puigcerver, J., Kolesnikov, A., Ruyssen, P., Riquelme, C., Lucic, M., Djolonga, J., Pinto, A.S., Neumann, M., Dosovitskiy, A., et al.: A large-scale study of representation learning with the visual task adaptation benchmark. arXiv preprint arXiv:1910.04867 (2019)