# Appendix

## A.1. Additional Experiments on LORS

We would like to showcase the potential of LORS through further experiments, it shows its effectiveness in more tasks like image classification, different modules such as encoders, and across all weights of Transformers. In fact, We managed to achieve all above goals simultaneously: we applied LORS$^{\mathrm{T}}$ to Transformers [7] within a vision encoder and used it for the classification task on CIFAR-100 [4].

| Attention using LORS$^{\mathrm{T}}$ | FFN using LORS$^{\mathrm{T}}$ | Parameters each layer | Top-1(%) | Top-5(%) |
|---|---|---|---|---|
| | | 100% | 63.66 | 84.23 |
| ✓ | | 89.7% | 63.51 | 84.85 |
| | ✓ | 66.2% | 63.93 | 84.69 |
| ✓ | ✓ | 47.5% | 63.97 | 85.10 |

Table 1. Effects of LORS$^{\mathrm{T}}$ on Transformer-based DeiT encoder.

| Attention Param Groups | FFN Param Groups | Rank $r$ per group | Top-1(%) | Top-5(%) |
|---|---|---|---|---|
| $\{0_{\times 12}\}$ | $\{0_{\times 12}\}$ | 32 | 59.13 | 83.11 |
| $\{1_{\times 12}\}$ | $\{0_{\times 12}\}$ | 32 | 59.98 | 83.52 |
| $\{0_{\times 12}\}$ | $\{1_{\times 12}\}$ | 32 | 60.43 | 83.41 |
| $\{1_{\times 12}\}$ | $\{1_{\times 12}\}$ | 32 | 62.30 | 84.33 |
| $\{1_{\times 9}, 2, 4, 6\}$ | $\{1_{\times 12}\}$ | 32 | 62.92 | 83.83 |
| $\{1_{\times 12}\}$ | $\{6, 4, 2, 1_{\times 9}\}$ | 32 | 62.94 | 84.50 |
| $\{1_{\times 9}, 2, 4, 6\}$ | $\{6, 4, 2, 1_{\times 9}\}$ | 32 | 63.97 | 85.10 |

Table 2. Effect of LORS$^{\mathrm{T}}$ on DeiT with different configurations.

Specifically, DeiT-Tiny [6], whose encoder is comprised of 12 Transformer layers, is trained from scratch for 300 epochs in 1 hour on CIFAR-100 with 8 V100 GPUs. We resized images from $32 \times 32$ to $56 \times 56$, divided them into $14 \times 14$ patches as commonly done and kept original training settings except for retaining only feasible augmentations (Mixup [9], Cutmix [8], and RandomFlip).

Table 1 shows the main results. When applying LORS$^{\mathrm{T}}$ to all weights in each Tranformer, the total parameters of DeiT-Tiny's encoder are reduced to 47.5% of the original, while achieving better accuracy.

Table 2 shows an ablation study on the parameter allocation over layers. $\{1_{\times 9}, 2, 4, 6\}$ indicates that the first 9 layers of the encoder use LORS$^{\mathrm{T}}$ with 1 group of parameters, while the last 3 layers use 2, 4, and 6 groups, respectively. The last row is our default setting, it assigns more parameters to self-attention in the last 3 layers and to FFN in the first 3 layers. This selection comes from the visualization of features input to each layer. We find that low-layer ones appear variable and complex, while high-layer ones appear similar and simple, as shown in Figure 1. We hypothesize that the attention module needs more parameters to discern relationships between similar features, and FFN requires more parameters to process raw complex information. We select configurations for aforementioned AdaMixer experiments in a similar way. However, this empirical approach may not achieve optimal performance.

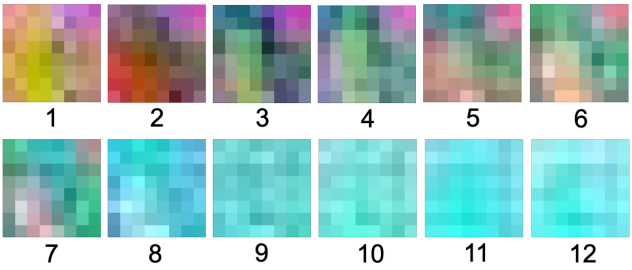


Figure 1. Visualizing input features of each layer in DeiT-Tiny.

## A.2. Discussion of LORS with regard to RNN

When LORS is applied to a stacked structure with only shared parameters, such a structure indeed degenerates into an RNN [3, 5]. However, The first row in Table **??** is still a hybrid recurrent architecture since it applies LORS to only part of AdaMixer [2] decoders' weights, so we performed the first 4 rows in Table 2 to facilitate this discussion. Its first row applied LORS to all weights in Transformer using no private parameters, fully degenerating into an RNN, and its performance is the worst. Improvement occurred weakly in the second and third rows with hybrid recurrent states, but significantly in the fourth row. Adding private parameters to all layers seems better than a pure RNN. These private parameters can surely be generated by a function of the previous layer, which we think could be achieved by a single LORS$^{\mathrm{A}}$. A promising attempt might be integrating LORS$^{\mathrm{A}}$ instead of LORS$^{\mathrm{T}}$ into Transformers.

## A.3. Validating the importance of self-attention and FFN in the performance of Transformers

A natural question is whether self-attention and FFN are both crucial for Transformers' performance, as this impacts the persuasiveness of the additional LORS experiments that rely on them. Table 3 shows that loading ImageNet [1] pretrained weights significantly affects the performance, highlighting the importance of both components.

| ATTN Pretrained Init | FFN Pretrained Init | Top-1(%) | Top-5(%) |
|---|---|---|---|
| ✓ | ✓ | 78.85 | 92.73 |
| ✓ | | 65.28 | 86.40 |
| | ✓ | 64.19 | 84.69 |
| | | 63.66 | 84.23 |

Table 3. Effect of whether pretrained weights loaded on attention module and feedforward module.

## References


[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 1

[2] Ziteng Gao, Limin Wang, Bing Han, and Sheng Guo. Adamixer: A fast-converging query-based object detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5364–5373, 2022. 1

[3] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 1

[4] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 1

[5] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986. 1

[6] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021. 1

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 1

[8] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019. 1

[9] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 1