A Appendix

A.1 Illustration of ViT model in detail

The ViT model that we used for the classification task is adapted from a public codebase[2]. The detailed structure of the ViT is depicted in Figure 12, where we only fine-tune the projection layers for query and value in the Self-Attention modules. In ViT-base, depth (L in Figure 12) is set to be 12 and dimension is 768. The total number of parameters of the ViT base model is 86.6M.

A.1.1 ImageNet21k to CIFAR-100 transfer

Firstly, we used a ViT model[3] pretrained on ImageNet21k for CIFAR-100 transfer. As this pretrained model has no classifier head available, a new classifier head to match the number of classes from 21k to 100 is added for classifing CIFAR100 dataset. All layers of the pretrained ViT model are frozen except the SuperLoRA parameters and the new classifier head. The result of SuperLoRA for CIFAR-10 is shown in Figure 5, achieving a significant reduction by 3 to 10 folds in the required number of parameters over LoRA.

A.1.2 ImageNet1k to CIFAR-100 transfer

To exclude extra fine-tuning budget introduced in the classifier head, automatic label matching is used for ViT model pretrained on ImageNet1k. Specifically, ViT-base model pretrained on ImageNet1k is loaded along with the pretrained classifier head. Then, feed all training data from CIFAR-100 into this pretrained model, and corresponding labels of CIFAR-100 in ImageNet1k are obtained by voting. When there is a tie, the label that has larger gap with the second voting label wins the label. If one label is taken, the label with second voting is assigned. In this way, all 100 classes in CIFAR-100 get their corresponding labels in ImageNet1k, and the classifier head can be frozen. Other settings are same as experiments above. The results can be found in Figure 11. Compared with Figure 5, 768×100 less parameters are fine-tuned, while most conclusions still hold true.

A.1.3 ImageNet1k to CIFAR-10 transfer

For transfer learning from ImageNet1k to CIFAR-10, we used a ViT base model[4] which is pretrained for ImageNet1k. The classifier head is frozen after selecting most relevant labels in ImageNet1k, *i.e.* [404, 436, 94, 284, 345, 32, 340, 510, 867], corresponding to [airliner, humming bird, siamese cat, ox, golden retriever, tailed frog, zebra, container ship, trailer truck]. It fine-tunes with 3000 steps at most and the best accuracy is reported.

Detailed ranks we tested are as follows:

- LoRA (2D): ranks: 1, 2, 4, 6, 8, . . . , 64, 128
- SuperLoRA (2D): groups: 1, 4, 8, 12; ranks: 1, 2, 4, 6, 8, . . . , 64, 128
- SuperLoRA (2D, reshape):

[2]`https://github.com/bwconrad/vit-finetune`
[3]`https://huggingface.co/google/vit-base-patch16-224-in21k`
[4]`https://huggingface.co/google/vit-base-patch16-224`

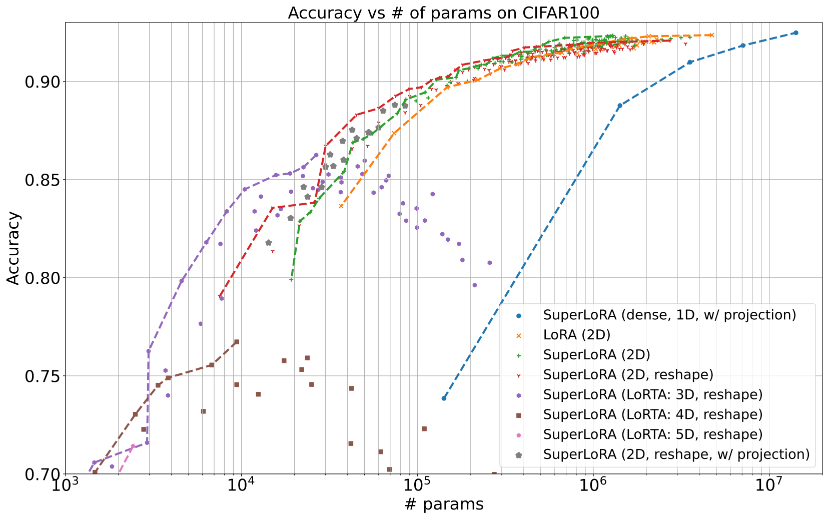


Figure 11: ImageNet1k to CIFAR100 transfer learning with frozen classifier head by automatic label matching.

  - groups: 1, 4, 12, ranks: 1, 2, 4, 6, 8, . . . , 64, 128;
  - group 8, ranks: 1, 2, 4, 6, 8, . . . , 24, 28, 32, 36, . . . , 64

- SuperLoRA (LoRTA: 3D, reshape): groups: 1, 4, 8, 12; ranks: 1–6, 8, 10, 12, . . . , 24
- SuperLoRA (LoRTA: 4D, reshape):
  - group 1; ranks: 1–6, 8, 10, 12, . . . , 22
  - group 4; ranks: 1–6, 8, 10, 12, . . . , 16
  - group 8; ranks: 1–6, 8, 10, 12, . . . , 18
  - group 12; ranks: 1–6, 8, 10, 12
- SuperLoRA (LoRTA: 5D, reshape):
  - groups 1, 4, 8; ranks: 1–6, 8
  - group 12; ranks: 1–6

The result of SuperLoRA for CIFAR-10 is shown in Figure 6, achieving a significant reduction by 3 to 10 folds in the required number of parameters over LoRA.

A.2 Illustration of diffusion model in detail

The classifier-free diffusion model [18] that we used for image generation is adapted from a public codebase[5]. Its U-Net structure is illustrated in Figure 13, which contains 21 attention modules, where the number of input/output channels of the attention modules is either 64 or

[5]https://github.com/coderpiaobozhe/classifier-free-diffusion-guidance-Pytorch

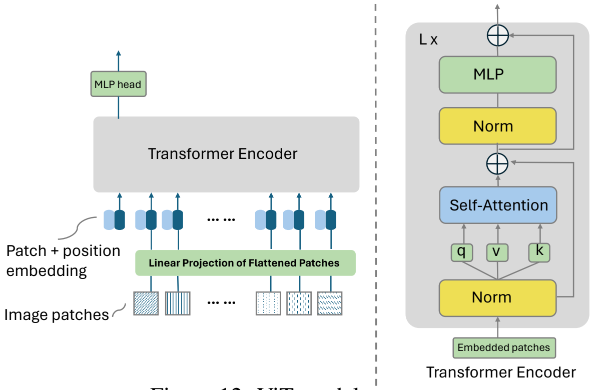


Figure 12: ViT model structure.

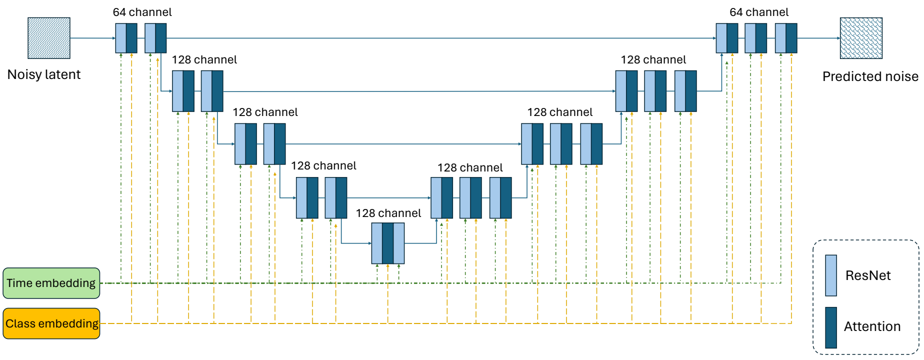


Figure 13: Classifier-free diffusion model structure.

128. We only fine-tune the query and value projection layers of those attention modules. The total number of parameters of the U-Net base model is 10.42M, including 300 parameters for the class embedding.

A.3 Illustration of grouping mechanism

Figure 2 illustrates several different cases of the grouping mechanism. Figure 2(a) is the conventional weight-wise grouping, used for typical LoRA. Each weight correction, *i.e.* $\Delta W_{\mathrm{v}\ell}$ and $\Delta W_{\mathrm{q}\ell}$ for value and query projections at layer ℓ, is individually represented by a rank-r decomposition: $A_g B_g^\top$ for group g. Figure 2(b) shows layer-wise grouping, where the LoRA unit in each group jointly adapts both value and query projections in each layer. When we stack multiple weight matrices in a naïve way, the 2D array will have unbalanced fan-in/fan-out shape, leading to inefficient low-rank decomposition. Figure 2(c) can solve this issue by reshaping the 2D array into a regular square shape before low-rank decomposition. As the reshaping is already breaking the geometric meaning of the original 2D weights, the grouping need not necessarily aligned with the weight boundary as shown in a general grouping case of Figure 2(d). Further, applying a projection function $\mathcal{F}(\cdot)$ as shown in Figure 2(e), the element distribution can be shuffled and mixed-up to relax the geometric restriction of original LoRA. LoRTA can further generalize the reshaping by folding the 2D array to any arbitrary M-dimensional tensor array by using the Tucker decomposition as shown in Figure 2(f). Relaxing the geometric constraint can improve the parameter efficiency as shown in this paper. We further make a geometric analysis of our grouping methods.

A.4 Geometric analysis

SuperLoRA adapts multiple attention modules at once, and relaxes the underlying geometric restrictions inherent to the 2D weights for each attention module, by employing grouping, reshaping, and projection (including shuffling). To better understand how SuperLoRA works differently from LoRA, geometric analysis is conducted for the classification task. Specifically, we pick 4 different methods with a comparable number of parameters around 100,000:

- LoRA (2D): #param 147,456, accuracy 0.9113;
- SuperLoRA (2D): #param 115,200, accuracy 0.9170;
- SuperLoRA (2D, reshape): #param 165,572, accuracy 0.9218;
- SuperLoRA (2D, reshape, w/ projection): #param 138,372, accuracy 0.9213.

The weight correction term ΔW is compared to the full dense FT case, which involves 14M parameters achieving an accuracy of 0.9290. We analyze three different geometric measures with respective to the FT weight ΔW_{dense}: i) left-singular similarity; ii) right-singular similarity; and iii) Euclidean distance. Letting U and V denote the left- and right-singular vectors of ΔW for each variant listed above, these metrics are defined as follows:

$$d_{\text{L}} = \frac{1}{\sqrt{k}} \| U_{\text{dense}}[:,:k]^\top U_{\text{variant}}[:,:k] \|_2, \tag{5}$$

$$d_{\text{R}} = \frac{1}{\sqrt{k}} \| V_{\text{dense}}[:k,:] V_{\text{variant}}[:k,:]^\top \|_2, \tag{6}$$

$$d_{\text{E}} = \frac{\| \Delta W_{\text{dense}} - \Delta W_{\text{variant}} \|_2}{\| \Delta W_{\text{dense}} \|_2}. \tag{7}$$

Note that d_{E} approaches to 0 when ΔW converges to the dense FT case, while d_{L} and d_{R} converge to 1.

The top $k = 5$ principal singular vectors are analyzed as shown in Figure 14. The ViT model has 12 attention modules, and we plot the total of 24 points for the query and value projection weights. The first row shows the query weights for d_{L} *vs.* d_{R}, d_{E} *vs.* d_{R}, and d_{E} *vs.* d_{L} from left to right across the columns. The second and third rows are for the value weights, and both query and value weights, respectively.

We see that the Euclidean distance d_{E} is significantly decreased for SuperLoRA, especially with reshaping applied. It explains the improved accuracy with reshaping. Although grouping, reshaping, and projection can break the geometric meaning of the original 2D weights, the subspace similarity is not completely lost. Especially for query weights, SuperLoRA shows higher right-singular similarity than LoRA. As the embedding vector passes through right-hand side of the weight, principal right-singular vectors perform as a low-rank subspace mapping of the input vector while the left-singular vectors work as mapping the subspace towards the output vector. While SuperLoRA with reshaping tends to preserve

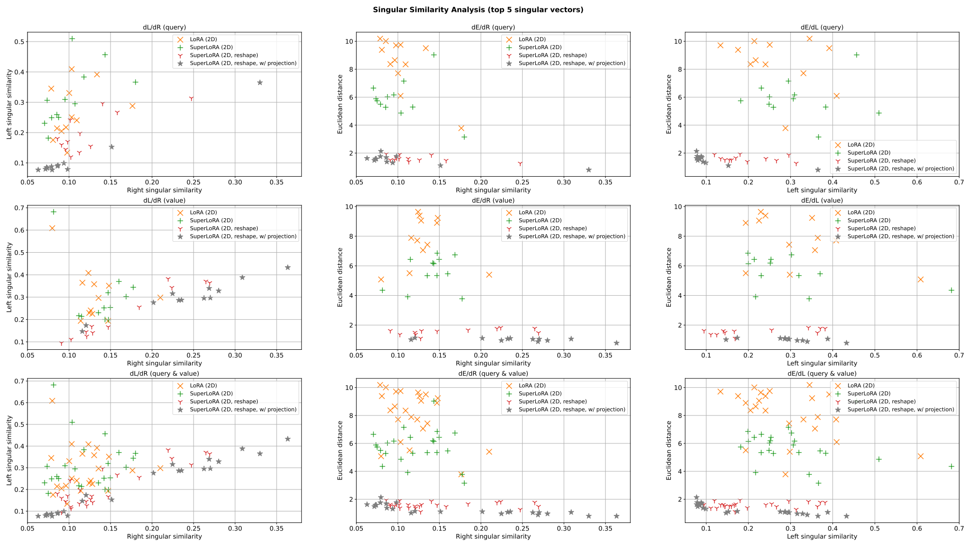


Figure 14: Geometric similarity analysis (top 5 principal singular vectors).

higher right-singular similarity, LoRA tends to preserve higher left-singular similarity. Further, it is found that the corrections for query and value weights behave differently with reshaping, *i.e.*, right-singular similarities for the value weights are much larger than for query weights.

A.5 Grouping effect on SuperLoRA (1D, dense, with projection)

As the fixed projection matrix is shared across all groups, the number of groups will affect the size of the projection matrix directly. To explore this influence, dense FT with projection is tested for different splitting, from 1 to 12 groups. According to Figure 15, using 1 group achieves the best overall accuracy and using 4 or 8 groups are comparable to a smaller projection ratio. When the projection matrix is too small, *e.g.*, with 12 groups, accuracy drops greatly. This confirms that jointly updating multiple attention modules is beneficial.

A.6 Linear *vs*. nonlinear projection

Besides linear projection, we also examined nonlinear projections. We use the "tanhshrink" operation, denoted by $\mathtt{tanhs}(x) := x - \tanh(x)$, after the fixed linear projection, resulting in the 'nonlinear' and 'nonlinear$_{\text{v2}}$' variants. Note that the 'v2' projection uses a Gaussian random vector rather than a binary random vector $\mathcal{B}$ for the Fastfood projection as shown in Figure 3. More specifically, we consider six variants for the projection function $\mathcal{F}(\cdot)$ in this paper:

- identity (no projection): $\mathcal{F}(x) = x$;
- shuffling: $\mathcal{F}(x) = x\mathit{\Pi}$;

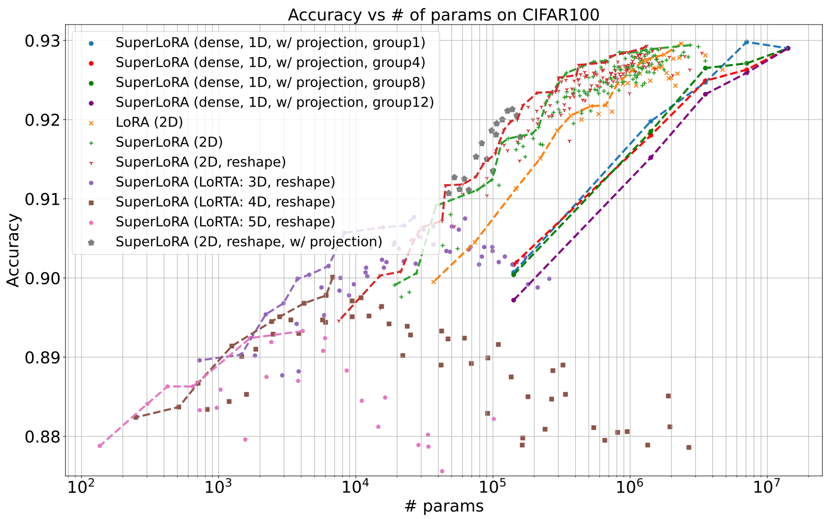


Figure 15: More groups (*i.e.* less fixed projection parameters) on SuperLoRA (1D, dense, w/ projection).

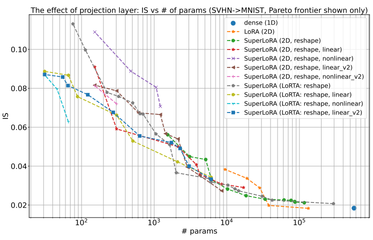
- linear: $\mathcal{F}(x) = x\mathcal{H}' \operatorname{diag}[\mathcal{G}]\, \Pi\, \mathcal{H} \operatorname{diag}[\mathcal{B}]$;
- linear$_{\text{v2}}$: $\mathcal{F}(x) = x\mathcal{H}' \operatorname{diag}[\mathcal{G}]\, \Pi\, \mathcal{H} \operatorname{diag}[\mathcal{G}']$;
- nonlinear: $\mathcal{F}(x) = \operatorname{tanhs}\big[x\mathcal{H}' \operatorname{diag}[\mathcal{G}]\, \Pi\, \mathcal{H} \operatorname{diag}[\mathcal{B}]\big]$;
- nonlinear$_{\text{v2}}$: $\mathcal{F}(x) = \operatorname{tanhs}\big[x\mathcal{H}' \operatorname{diag}[\mathcal{G}]\, \Pi\, \mathcal{H} \operatorname{diag}[\mathcal{G}']\big]$.

Here, Π performs a random permutation of a vector. The Walsh–Hadamard matrices $\mathcal{H}' \in \mathbb{R}^{N_{\text{in}} \times 2^N}$ and $\mathcal{H} \in \mathbb{R}^{2^N \times N_{\text{out}}}$ are left- and right-truncated versions of a regular Walsh–Hadamard matrix $\mathcal{H}_2^{\otimes N} \in \mathbb{R}^{2^N \times 2^N}$, where $[\cdot]^{\otimes N}$ denotes N-fold Kronecker power and $\mathcal{H}_2 = \frac{1}{\sqrt{2}}\left[\begin{smallmatrix}1 & 1\\ 1 & -1\end{smallmatrix}\right]$. Letting N_{in} and N_{out} be the number of elements for the input and output of the projection function $\mathcal{F}(\cdot)$ with a compression ratio of $\rho = N_{\text{in}}/N_{\text{out}}$, the exponent N is chosen as $N = \operatorname{ceil}[\log_2(\max(N_{\text{in}}, N_{\text{out}}))]$. In practice, the left-truncated Walsh–Hadamard matrix is realized by input zero-padding before fast Walsh–Hadamard transform. The random vector $\mathcal{G}$ is hence of size 2^N, and drawn from the normal distribution. Here, $\mathcal{G}' \in \mathbb{R}^{N_{\text{out}}}$ is another random vector drawn from the normal distribution while $\mathcal{B} \in \{\pm 1\}^{N_{\text{out}}}$ is a random vector drawn from the Rademacher distribution.

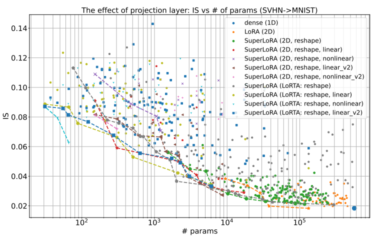
Figure 16 shows the comparison of several projection variants. Surprisingly, with the same number of parameters, the linear version outperforms the nonlinear versions in most cases.

A.7 Classification transfer task

We evaluated SuperLoRA with grouping with/without reshaping to square-like for 2D $\Delta W_{\text{group}_g}$, reshaping version for higher-order $\Delta W_{\text{group}_g}$ including 3D, 4D and 5D. The fixed projection layers are inserted to SuperLoRA with reshaping (2D version) and also dense. Original



(a) linear *vs*. nonlinear (IS, Pareto only)



(b) linear *vs*. nonlinear (IS, all points)

Figure 16: Comparison between Linear/Linear$_{\text{v2}}$/Nonlinear/Nonlinear$_{\text{v2}}$ projections.

weight-wise LoRA is also examined for comparison by setting the number of groups to the number of query and value weights (24 for 12-layer ViT-base) as all projection weights for ViT-base are equal size. Each correction weight is of size 768×768 as the projection weight for query/value, resulting in 14M parameters. Except for most cases, more ranks are needed to span the parameter axis well, including larger ranks from 34 to 128 and smaller ranks below 8 for LoRTA. Projection compression ratio is from $\rho \in \{0.5, 0.25, 0.1, 0.01\}$, and the fixed projection parameters are shared across all groups in our experiments.

We also examined another transfer learning task from ImageNet1k to CIFAR10. Most settings are same as Figure 5 for transfer learning from ImageNet21k to CIFAR100. The classifier head is frozen after selecting most relevant labels in ImageNet1k. Details are found in Appendix A.1.3. Classification results can be found in Figure 6. Even though only attention modules are adapted, overall performance is excellent, reaching an accuracy close to 0.99. Besides, SuperLoRA significantly outperforms original LoRA in terms of both classification accuracy and the parameter range it covers as the transfer learning. SuperLoRA (2D, reshape) shows at least 3-fold reduction in the required number of parameters compared to LoRA. Noticeably, when comparing the lowest-rank LoRA with around 0.97 accuracy, SuperLoRA (2D, reshape, w/ projection) improves the accuracy by about 1%, and moreover the required number of parameters can be greatly reduced by 10 folds with SuperLoRA (LoRTA: 3D, reshape) to maintain the comparable accuracy.

A.8 Image generation transfer task

A.8.1 Settings

The model we worked on is a classifier-free diffusion model [18] and the correction weights from LoRA variants are added to query and value projection matrices in the attention modules of U-Net backbone [36]. Note that the size of projection weights differs across layers for this U-Net structure, which allows us to examine the performance of SuperLoRA after breaking the boundaries of different weight matrices. More details of the diffusion model are described in Appendix A.2. For comparison, the original weight-wise LoRA and dense FT are also evaluated. For SuperLoRA variant, LoRA, LoNKr and LoRTA consider three versions: weight-wise, group-wise and group-reshaped. The scaling factor α of LoRA is fixed to 2.0 for all variants unless specified. 40 epochs with a batch size of 32 are carried out and results plotted are mainly from epoch 20 noticing convergence becomes stable around epoch 20. The maximum rank is set to 32 by default and a constraint $r < \min(d_1, d_2)$ is imposed.

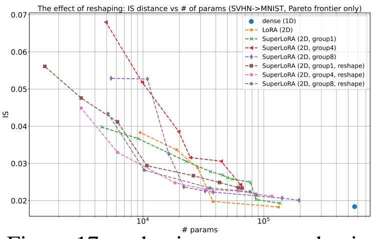


Figure 17: reshaping *vs*. non-reshaping

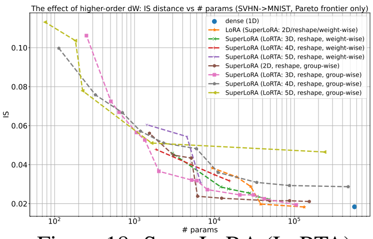


Figure 18: SuperLoRA (LoRTA)

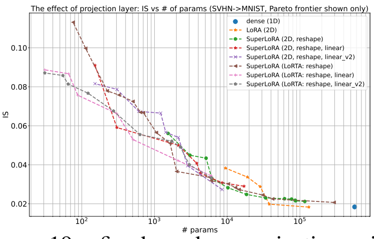


Figure 19: fixed random projection within group

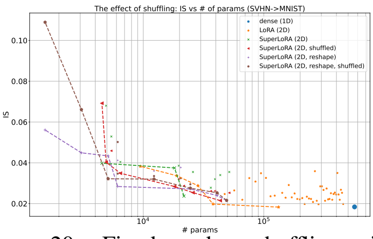


Figure 20: Fixed random shuffling within group.

To evaluate the quality of images generated by the fine-tuned diffusion model, we consider several metrics including Inception Score (IS) [37], Fréchet Inception Distance (FID) [17], Multi-Scale Intrinsic Distance (MSID) [40], Kernel Inception Distance (KID) [6], Recall and Precision [27]. Except for the recall and precision metrics, all metrics should be lower for higher-quality image generations.

A.8.2 Reshaping effect

To evaluate the importance of reshaping, we compare group-wise SuperLoRA with and without reshaping in Figure 17. For weight-wise LoRA, most weight matrices corrected are square already. For all splitting with groups $G = 1$, 4 and 8, we confirmed that reshaping shows smaller number of parameters and better IS compared with their corresponding non-reshaping counterparts. This indicates that reshaping ΔW to regular tensor array (square, cube, and hyper-cube) is vital for SuperLoRA fine-tuning to prevent unbalanced skew tensors when adapting multiple weights at once.

A.8.3 LoRTA

LoRTA reshapes ΔW_{all} to high-order tensor. We evaluated 3D, 4D and 5D, as data points become much less when the dimension is too small for all planes when order is larger than 5D. From Figure 18, the higher the order of tensor folding, the less data points we have. In both weight-wise and group-wise version, 5D LoRTA reduces the least parameter it requires. Especially for group-wise LoRTA, 5D LoRTA requires less than 80 parameters to produce a result compared with beyond 1000 for 2D LoRTA and beyond 200 for 3D LoRTA, while original LoRA needs about 10^4 parameters, about 120-fold more parameters. To achieve a comparable IS of LoRA having 10^4 parameters, LoRTA (3D) just needs 2×10^3 parameters, *i.e*. 5-fold reduction.

A.8.4 Projection effect

SuperLoRA can use a projection layer $\mathcal{F}$ which is randomly initialized but fixed at both finetuning and inference. Linear fastfood projection and nonlinear projection with tanshrink applied after the linear projection matrix are evaluated. Besides, a modified version of fastfood projection with random Gaussian instead of random binary $\mathcal{B}$ is also tested for both linear and nonlinear versions, denoted as $\text{linear}_{\text{v2}}$ and $\text{nonlinear}_{\text{v2}}$ respectively. The projection matrix is shared across all groups. We evaluated number of groups $G \in \{1,4\}$, rank $r \in \{1,4,8\}$ and projection ratio $\rho \in \{0.01, 0.1, 0.5\}$ on SuperLoRA (2D, reshape) and SuperLoRA (LoRTA, reshape) for 3D, 4D and 5D tensor.

Figure 19 demonstrates with smaller projection ratio, required parameters for both SuperLoRA (2D, reshape) and SuperLoRA (LoRTA, group-wise) are pushed to extremely low-parameter regimes. The least parameter required becomes only about 30, compared with 10,000 for original LoRA. Surprisingly, linear version for both methods shows better performance than nonlinear version which are attached in Appendix A.6. Besides, in extremely low-parameter regimes, higher rank with projection layer for SuperLoRA (LoRTA, group-wise) works better than small ranks itself, showing promising direction to explore projection layer in extremely low-parameter regime. In terms of linear *vs.* $\text{linear}_{\text{v2}}$, $\text{linear}_{\text{v2}}$ shows better performance in higher-parameter area while linear works better in lower-parameter area, even better than SuperLoRA (LoRTA) without projection.

A.8.5 Shuffling effect

As a simple projection, we studied a random shuffling to distribute ΔW_{group} before adding it to corresponding W. We evaluated SuperLoRA (2D) and SuperLoRA (2D, reshape) with/without shuffling for groups $G \in \{1,4,8,16]\}$ and ranks $r \in \{1,4,8\}$, where the shuffled indices are shared across all groups. The shuffling corresponds to one of fastfood projection modes by setting projection ratio to $\rho = 1$ with only permutation matrix Π. As shown in Figure 20, shuffling inside groups had no harm on IS. It even improved IS for SuperLoRA (2D) in most cases.

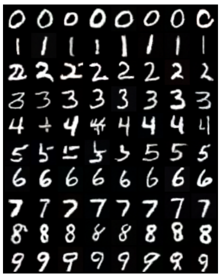
A.9 Visualization

To better understand the superiority of SuperLoRA, especially in low-parameter regimes, we visualize a set of generated images from SuperLoRA, as well as dense FT and LoRA, from a range of parameter setting: high-parameter ($>$ 70,000), middle-parameter (from 5,000 to 10,000), low-parameter (around 1,000) and extremely-low parameter ($<$ 100) regimes. We selected one image with the best IS for each hyper-parameter setting we have tested under same level of parameter amount. Figure 21 shows that all generated images by the transfer learning model from SVHN to MNIST are close to images from MNIST dataset itself with black-white background, removing most domain information of color SVHN. SuperLoRA (2D, group8, rank13) in Figure 21(c) shows competitive results with LoRA (rank8) using 5,000 less parameters.

For the middle-parameter regimes, Figure 22 shows visualization of LoNKr, SuperLoRA (2D, reshape), LoRTA (3D, reshape), LoRTA (4D, reshape) and SuperLoRA (2D, reshape, projection). More domain information with colorful digits and background occur occasionally. There are also some missing digits presented in middle-parameter area.

(a) dense
#param 565,248
IS 0.0184

(b) LoRA $r = 8$
#param 75,776
IS 0.03025


(c) SuperLoRA
#param 70,720
IS 0.0305

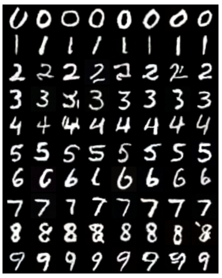
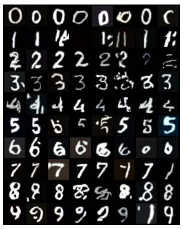
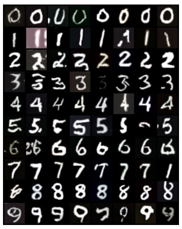

(d) SuperLoRA
#param 73,728
IS 0.0263

Figure 21: Visualization of generated images under high-parameter level ($> 70{,}000$).


(a) LoNKr
#param 5,112
IS 0.036

(b) SuperLoRA
#param 10,752
IS 0.0294

(c) LoRTA
#param 8,160
IS 0.0272


(d) LoRTA
#param 11,100
IS 0.036

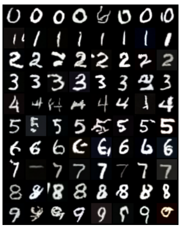

(e) SuperLoRA w/ p
#param 8,512
IS 0.0273

Figure 22: Visualization of generated images under middle-parameter level ($[5{,}000, 20{,}000]$).

When the number of parameters is as low as 1,000, even though only few choices left like LoNKr and LoRTA, one can always stretch hyper-parameter settings from middle-parameter level coupled with fixed linear projection layer to compress the tensor size. In this way, the strength of middle-parameter level gets extended to low-parameter area. As shown in Figure 23, compared with the visualization from middle-parameter results, more missing digits and more colorful backgrounds are presented.

Finally, we also visualized a few images from extremely-low parameter level less than 100 in Figure 23. Surprisingly, domain transfer in those images is somewhat realized from SVHN to MNIST even with such an extremely few parameter case such as 31, which is more than four orders of magnitude smaller than dense FT.

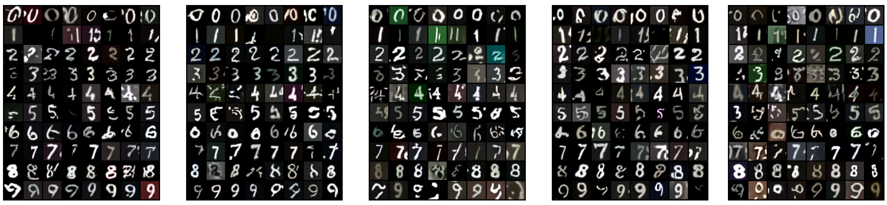
A.10 LLM transfer task

A.10.1 More metrics on LLM task

Except BLEU score, 4 other metrics are also used to examine the effectiveness of SuperLoRA on LLM, including NIST, METEOR, CIDEr and ROUGE_L as shown in Figures 24 to 27.

A.10.2 Geometric Analysis

As in Appendix A.4, we also analyze the geometric similarity for LLM task. Data points we selected for the analysis are:



(a) LoNKr #param 1,080 IS 0.0471

(b) LoRTA #param 1,060 IS 0.0565

(c) SuperLoRA w/ p #param 832 IS 0.0607

(d) LoRTA #param 76 IS 0.1131

(e) LoRTA w/ p #param 31 IS 0.0871

Figure 23: Visualization of generated images under low-parameter level (1,000) and extremely-low level (< 100).

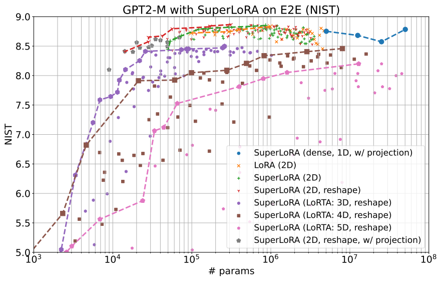


Figure 24: NIST scores

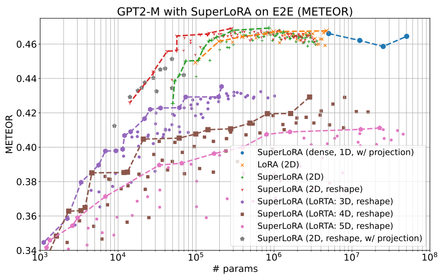


Figure 25: CIDER scores

- LoRA (2D): #param 98,304, BLEU 69.02, NIST 8.7224, METEOR 44.3, ROUGE_L 70.54, CIDEr: 2.396;
- SuperLoRA (2D): #param 159,744, BLEU 69.83, NIST 8.7789, METEOR 46.23, ROUGE_L 71.37, CIDEr: 2.4719;
- SuperLoRA (2D, reshape): #param 113,520, BLEU 69.81, NIST 8.7726, METEOR 46.56, ROUGE_L 71.42, CIDEr: 2.5007;
- SuperLoRA (2D, reshape, w/ projection): #param 49,152, BLEU 67.56, NIST 8.5668, METEOR 45.13, ROUGE_L 69.63, CIDEr: 2.3515.

Full dense FT is also used to compare in the weight term ΔW, which has 50M trainable parameters, and returns BLEU 69.53, NIST 8.7839, METEOR 46.45, ROUGE_L 71.53 and CIDEr 2.4973. Noticeably, full dense FT does not achieve the best results compared with SuperLoRA. One interesting observation from Figure 28 is, different from Figure 14, learned weight updates from LoRA are clearly splitted into two sets, similar/unsimilar to dense FT ones.

A.11 Transfer learning from SVHN to MNIST

A.11.1 Grouping effect (complete results)

Scatter plots of all metrics (FID, IS, KID, MSID, Improved Precision and Improved Recall) are given in Figure 29. Except IS, all metrics show many examples performing better

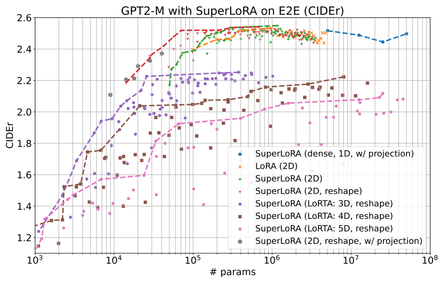


Figure 26: CIDEr scores

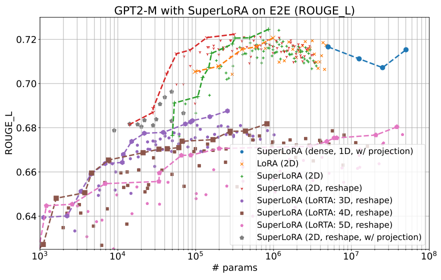


Figure 27: ROUGE_L scores

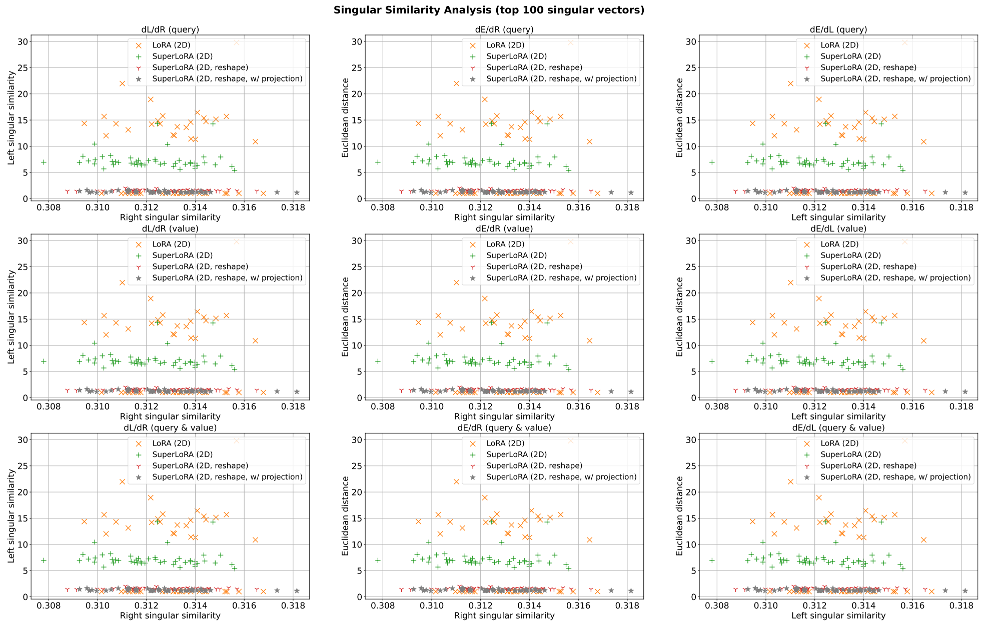


Figure 28: Geometric similarity analysis (top 100 principal singular vectors).

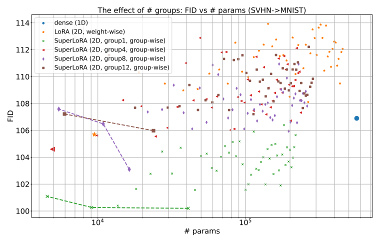
than dense FT, while worse according to the visualization results, indicating IS is a more reasonable quantitative metric in this case.

A.11.2 Reshaping effect (complete results)

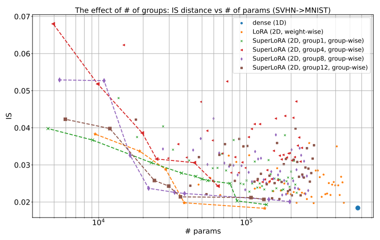
Complete results for reshaping, with scatter plots for all metrics, are shown in Figure 30.

A.11.3 SuperLoRA (LoNKr, complete results)

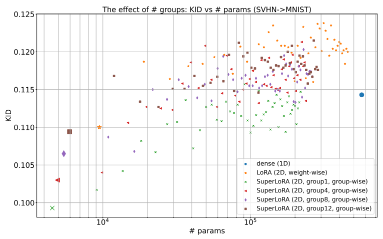
Complete results for SuperLoRA (LoNKr), with scatter plots for all metrics, are shown in Figure 31.



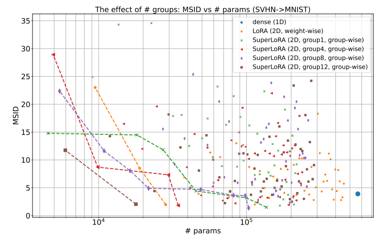
(a) weight-wise *vs*. group-wise (FID)



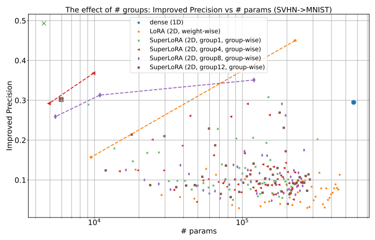
(b) weight-wise *vs*. group-wise (IS)



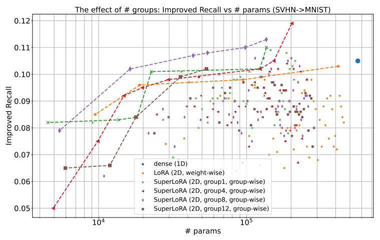
(c) weight-wise *vs*. group-wise (KID)



(d) weight-wise *vs*. group-wise (MSID)



(e) weight-wise *vs*. group-wise (Improved Precision)



(f) weight-wise *vs*. group-wise (Improved Recall)

Figure 29: Complete comparison between weight-wise LoRA and group-wise SuperLoRA.

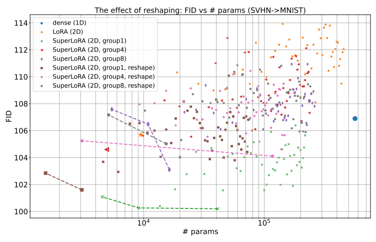
A.11.4 SuperLoRA (LoRTA, complete results)

Complete results for SuperLoRA (LoRTA), with scatter plots for all metrics, are shown in Figure 32.

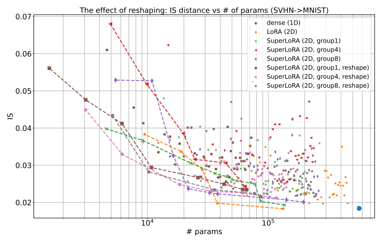
A.12 Transfer learning from MNIST to SVHN

A.12.1 Grouping effect

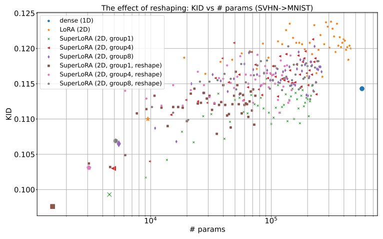
Transfer learning from MNIST to SVHN is also tested. Figure 33 shows that some metrics cannot function when transferred from a simpler dataset to a more complicated one, *e.g*. FID, IS, KID and Improved Precision, where some ill-posed cases appear. Besides this, we can still find from the Pareto frontiers that SuperLoRA extends LoRA to low-parameter regime and works better occasionally in terms of IS, MSID, Improved Precision and Improved Recall.



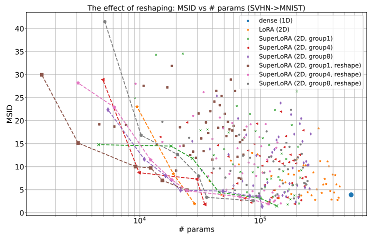
(a) reshaping *vs*. non-reshaping (FID)



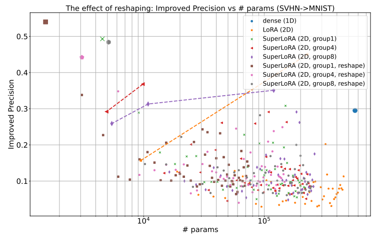
(b) reshaping *vs*. non-reshaping (IS)



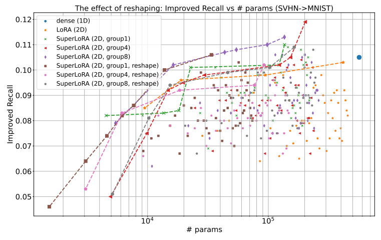
(c) reshaping *vs*. non-reshaping (KID)



(d) reshaping *vs*. non-reshaping (MSID)



(e) reshaping *vs*. non-reshaping (Improved Precision)



(f) reshaping *vs*. non-reshaping (Improved Recall)

Figure 30: Complete comparison between reshaping and non-reshaping SuperLoRA.

A.12.2 Reshaping effect

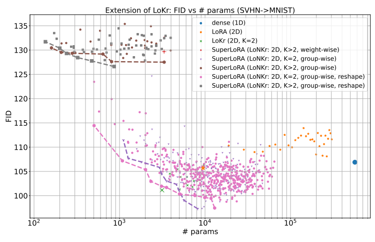
Figure 34 shows that SuperLoRA with reshaping works better than non-reshaping in most cases in transfer learning from MNIST to SVHN, consistent with the results in transfer learning from SVHN to MNIST.

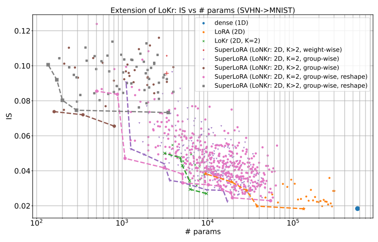
A.12.3 SuperLoRA (LoNKr)

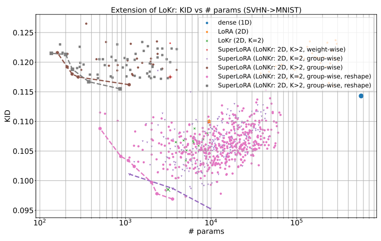
Figure 35 demonstrates the results of SuperLoRA (LoNKr). From MSID figure, we can see that, LoNKr extends LoKr to low-parameter regime, and achieves a better MSID.

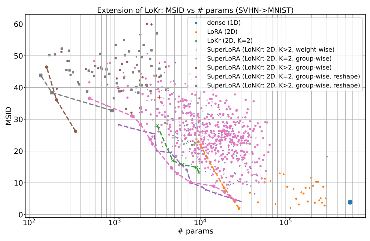
A.12.4 SuperLoRA (LoRTA)

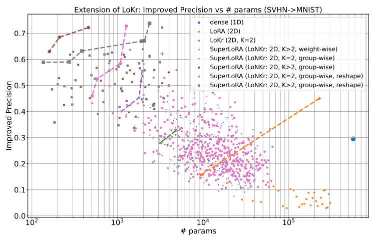
From FID and KID in Figure 36, LoRTA pushes required parameters from 10^4 to 10^2 compared with LoRA, providing more flexibility when the memory is limited.


(a) SuperLoRA (LoNKr, FID)


(b) SuperLoRA (LoNKr, IS)


(c) SuperLoRA (LoNKr, KID)


(d) SuperLoRA (LoNKr, MSID)


(e) SuperLoRA (LoNKr, Improved Precision)

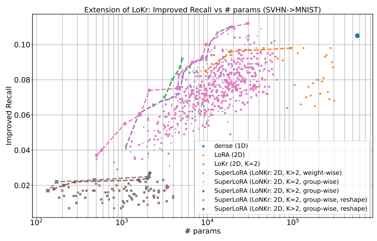

(f) SuperLoRA (LoNKr, Improved Recall)

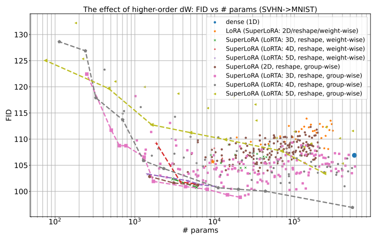
Figure 31: Complete results for LoNKr.

A.13 Effect of groups in LoNKr and LoRTA

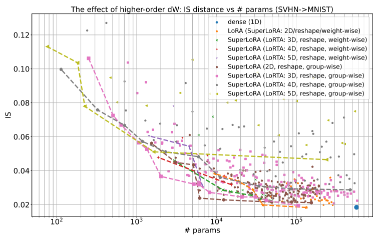
From Figure 37 and Figure 38, LoNKr and LoRTA behave differently in terms of the number of groups: for LoNKr, fewer groups are better (than more groups) in low-parameter regime, while they are comparable in high-parameter regime. However, LoRTA prefers less groups.

A.14 Effect of split K in LoNKr

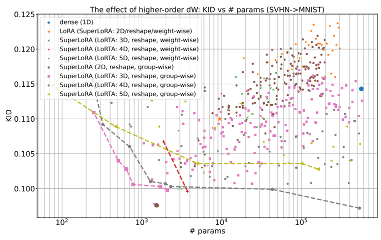
As shown in Figure 39, larger K works better than smaller ones in the low-parameter regime.



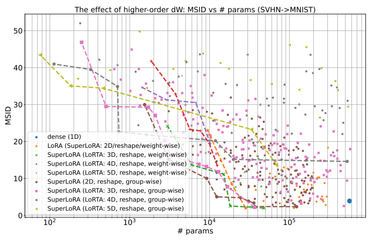
(a) SuperLoRA (LoRTA, FID)



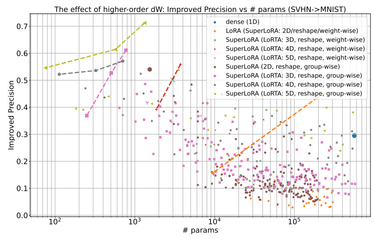
(b) SuperLoRA (LoRTA, IS)



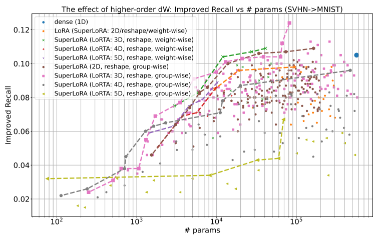
(c) SuperLoRA (LoRTA, KID)



(d) SuperLoRA (LoRTA, MSID)

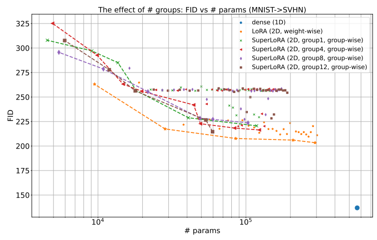


(e) SuperLoRA (LoRTA, Improved Precision)

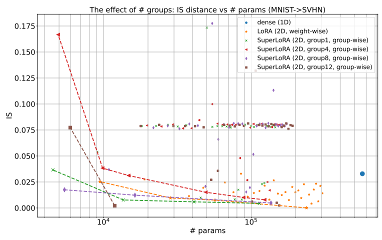


(f) SuperLoRA (LoRTA, Improved Recall)

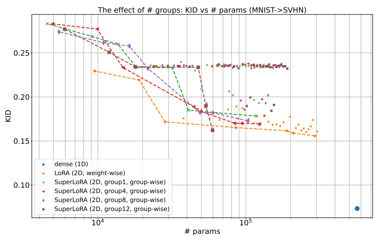
Figure 32: Complete results for LoRTA.



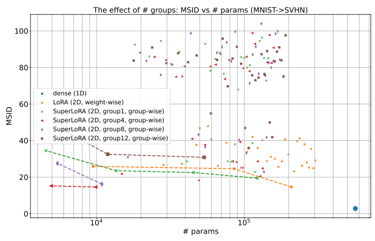
(a) weight-wise *vs*. group-wise (FID)



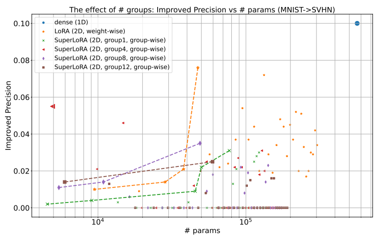
(b) weight-wise *vs*. group-wise (IS)



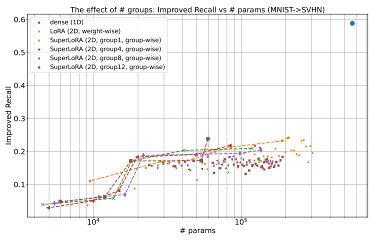
(c) weight-wise *vs*. group-wise (KID)



(d) weight-wise *vs*. group-wise (MSID)

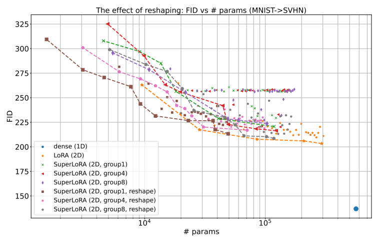


(e) weight-wise *vs*. group-wise (Improved Precision)

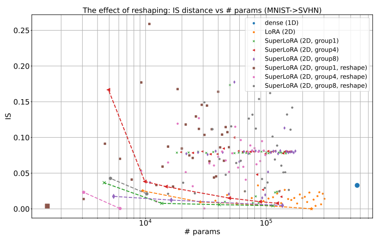


(f) weight-wise *vs*. group-wise (Improved Recall)

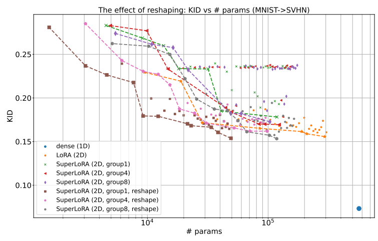
Figure 33: Complete comparison between weight-wise LoRA and group-wise SuperLoRA for transfer learning from MNIST to SVHN.



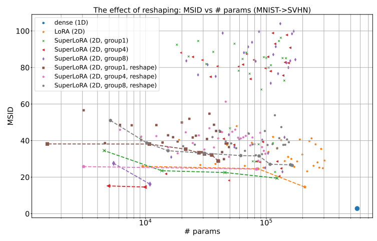
(a) reshaping *vs*. non-reshaping (FID)



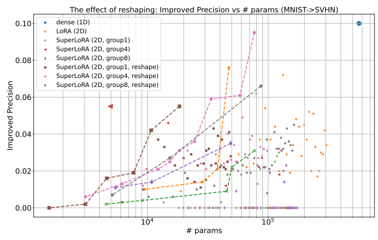
(b) reshaping *vs*. non-reshaping (IS)



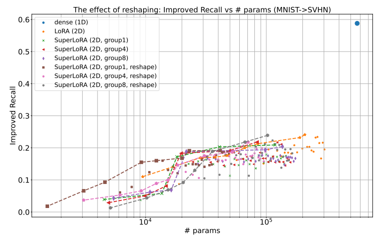
(c) reshaping *vs*. non-reshaping (KID)



(d) reshaping *vs*. non-reshaping (MSID)

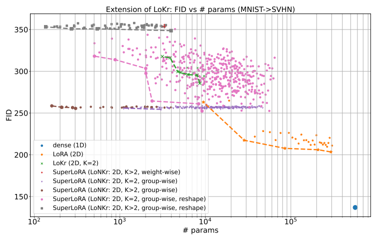


(e) reshaping *vs*. non-reshaping (Improved Precision)

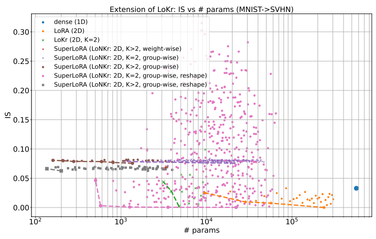


(f) reshaping *vs*. non-reshaping (Improved Recall)

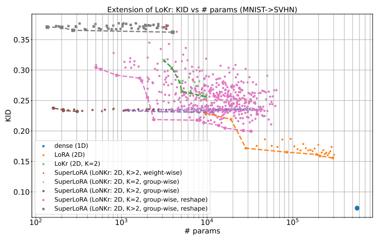
Figure 34: Complete comparison between reshaping and non-reshaping SuperLoRA for transfer learning from MNIST to SVHN.



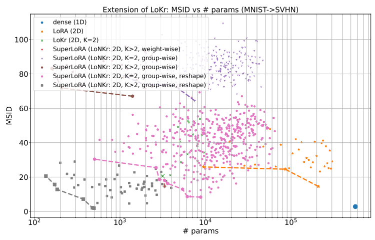
(a) SuperLoRA (LoNKr, FID)



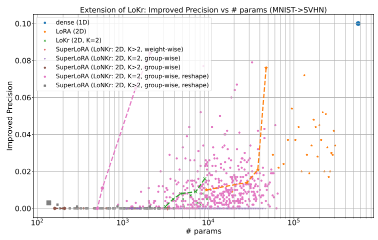
(b) SuperLoRA (LoNKr, IS)

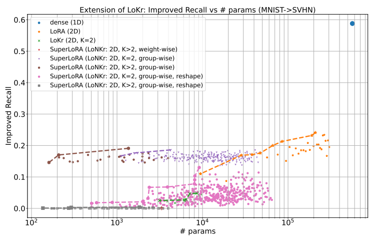


(c) SuperLoRA (LoNKr, KID)



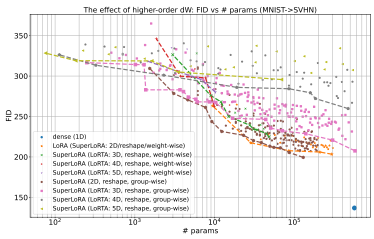
(d) SuperLoRA (LoNKr, MSID)



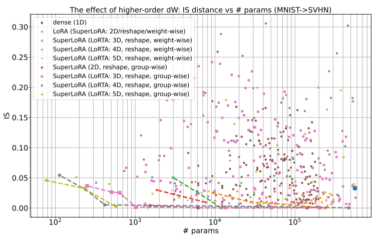


(e) SuperLoRA (LoNKr, Improved Precision) (f) SuperLoRA (LoNKr, Improved Recall)

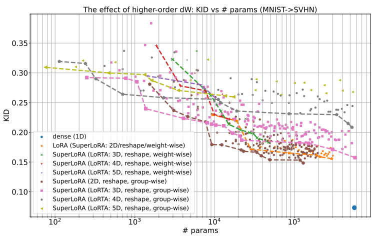
Figure 35: Complete results of SuperLoRA (LoNKr) for transfer learning from MNIST to SVHN.



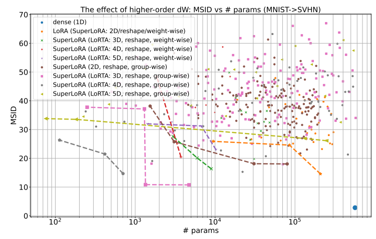
(a) SuperLoRA (LoRTA, FID)



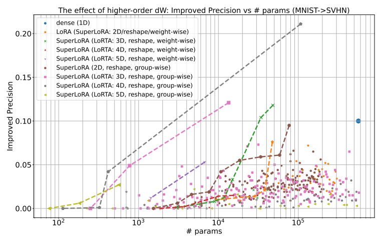
(b) SuperLoRA (LoRTA, IS)



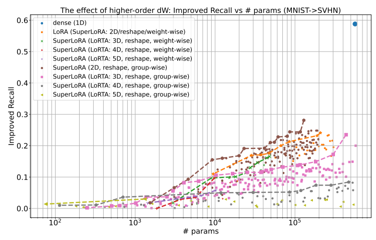
(c) SuperLoRA (LoRTA, KID)



(d) SuperLoRA (LoRTA, MSID)

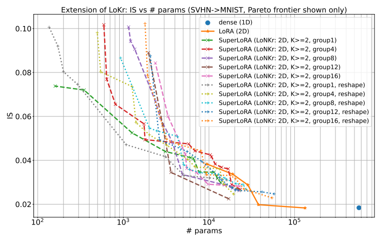


(e) SuperLoRA (LoRTA, Improved Precision)

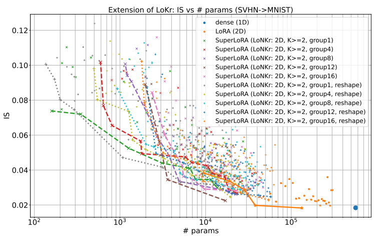


(f) SuperLoRA (LoRTA, Improved Recall)

Figure 36: Complete results of LoRTA for transfer learning from MNIST to SVHN.

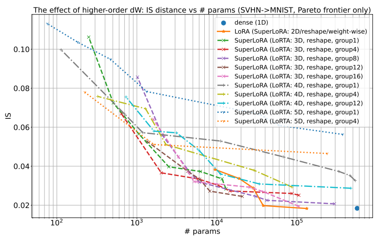


(a) SuperLoRA (LoNKr, Pareto frontier only)

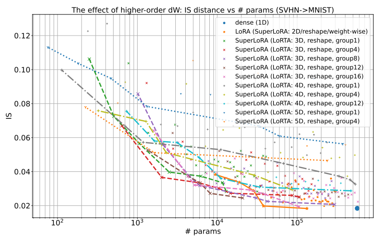


(b) SuperLoRA (LoNKr)

Figure 37: Effect of groups in LoNKr.

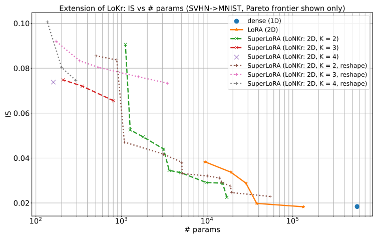


(a) SuperLoRA (LoRTA, Pareto frontier only)

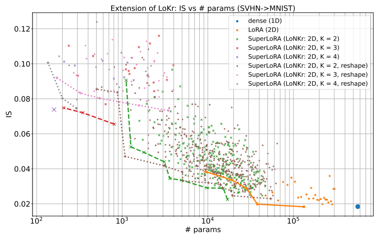


(b) SuperLoRA (LoRTA)

Figure 38: Effect of groups in LoRTA.



(a) SuperLoRA (LoNKr, Pareto frontier only)



(b) SuperLoRA (LoNKr)

Figure 39: Effect of K in LoNKr.