Supplementary Material
Group Orthogonalization Regularization for Vision Models Adaptation and Robustness

Yoav Kurtz
yoavkurtz@mail.tau.ac.il
Noga Bar
nogabar@mail.tau.ac.il
Raja Giryes
raja@tauex.tau.ac.il

School of Electrical Engineering
The Iby and Aladar Fleischman Faculty of Engineering
Tel Aviv University
Tel Aviv, Israel

1 Ablation Study

In this section, we conduct an ablation study to assess the impact of different hyperparameters on the effectiveness of our proposed weight regularization technique. Specifically, we examine the effects of the orthogonalization group size, its interaction with GN, and the magnitude of regularization. For all the experiments in this section, we maintain a consistent configuration, except for the parameter under investigation, which is varied accordingly. We utilize the ResNet110 architecture with GN and incorporate inter-group GOR.

We report the top-1 accuracy for different values of N (number of regularization groups), G (number of normalization groups in the GN layer) and λ (regularization strength) in Tables 1 to 3 respectively.

As mentioned in the main paper, we keep the number of filters/channels in each group to be at least 4, meaning that for every layer, the following holds:

$$N_{(l)} = \min\{N, \frac{C_{out}}{4}\} \quad \text{and} \quad G_{(l)} = \min\{G, \frac{C}{4}\}.$$

Due to this limitation, the neural networks utilized in this study consist of convolutional layers with a number of channels that allows the values of N and G to reach a maximal value of 16.

Table 1 shows that optimal outcomes are achieved by aligning orthogonalization groups with the normalization group, i.e. $N = G$. This way, the orthogonality among the normalization groups increases. Table 2 supports our choice of group size. The results in Table 3 present the hyperparameter search for the optimal value of λ.

2 Inter vs. Intra GOR

Figure 3 visualizes the difference between "inter" and "intra" group partition with GN. The groups of filters are determined according to the normalization groups of the features. As

Figure 1: Qualitative comparisons on Oxford102 between baseline fine-tuned model and model fine-tuned along with GOR using the same seed. The green rectangle is zoomed in by a factor of 1.5. For each of the two rows: Top is LoRA baseline. Bottom is LoRA with our method. For the generation of the flowers themselves, the two models are comparable with similar artifacts, while our model is more successful at generating the background grass. This may be explained by the fact that we encourage orthogonality in the weights, which helps support more details.

discussed in the paper, in the inter-group setting, filters within the same group are enforced to form an orthonormal set. On the other hand, in the intra-group setting, we enforce orthonormality between filters from different groups.

3 Computational Efficency

Figure 4 include a comparison of the regularization methods across memory consumption and number of operations. In each experiment, we calculate the regularization term for a single convolution layer with a kernel of dimensions: $C_{out} \times C_{in} \times h \times w = 256 \times 256 \times 3 \times 3$. We compare non-grouped orthogonalization regularization (SO) and the two GOR variants. All experiments were performed on NVIDIA GeForce RTX 2080 Ti.

4 Diffusion Models Adapters - Experiments Details

In this section, we elaborate on the training and evaluation protocols of adapters of diffusion models presented in Section 4.2.2 of the paper.

Experiment setting. Our training protocol is built upon the example[1] published by HuggingFace [6]. For the Pokemon-BLIP dataset [9], we train with a batch size of 4 and

[1] https://github.com/huggingface/diffusers/tree/main/examples/text_to_image

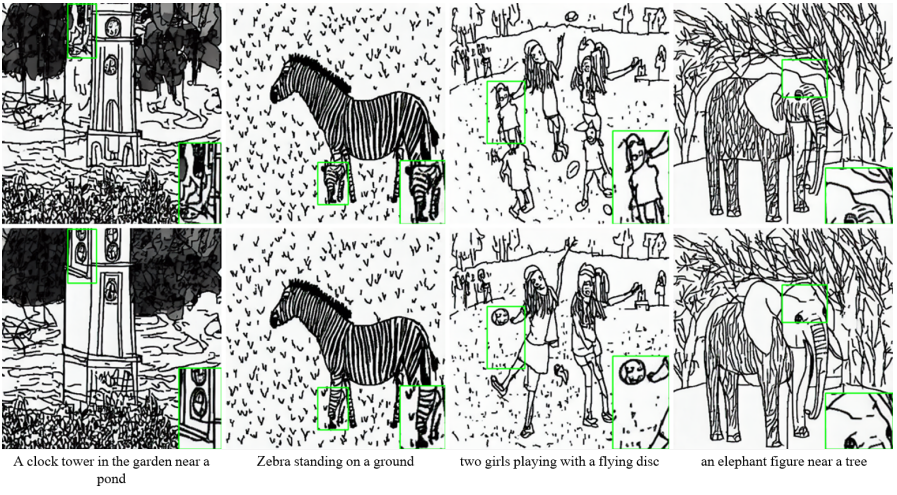

Figure 2: Qualitative comparisons on FS-COCO between baseline fine-tuned model and model fine-tuned along with GOR using the same seed. The green rectangle is zoomed in by a factor of 1.5. For each of the two rows: Top is LoRA baseline. Bottom is LoRA with our method. Our method improves the generation quality by both aligning with the text prompt more closely (second image from the right) and by removing artifacts.

Table 1: CIFAR10 Top-1 accuracy for a varying number of groups, N. ResNet110 GN model is used. We keep the number of normalization groups to be $G = \min\{32, \#\text{channels} / 4\}$. Mean and std across 3 seeds are reported.

N	**1**	**2**	**4**	**8**	**16**
G	**16**	**16**	**16**	**16**	**16**
	92.33 ± 0.03	92.55 ± 0.11	92.16 ± 0.03	92.31 ± 0.08	$\mathbf{92.73 \pm 0.03}$

512×512 resolution. As for the Oxford102 [3] and the FS-COCO [1] datasets, we use a batch size of 64 and 256×256 resolution. We set the base learning rate to 10^{-4} and apply a cosine scheduler. Data is pre-processed using central crop and normalization. Random flip is employed as data augmentation.

FID calculation. The procedure consists of two stages: first, producing samples from the model; second, computing the discrepancy between the InceptionV3 [5] statistics of the model-generated images and the original ones. For both steps, we build upon the code published for [2]. Following common practice, before being passed to the Inception model for statistics calculation, the images (both generated and non-generated) are undergone the same pre-processing (normalization and central crop) as mentioned above.

5 Qualitative Examples

We present more qualitative comparisons between our method and the baseline in Figures 6 to 11. The text prompt used to condition the generative model is presented at the bottom of

Table 2: CIFAR10 Top-1 accuracy for different values of G. ResNet110 GN model is used. We keep $N = G$. Mean and std across 3 seeds are reported.

N	**1**	**2**	**4**	**8**	**16**
G	**1**	**2**	**4**	**8**	**16**
	92.19 ± 0.17	92.45 ± 0.1	92.59 ± 0.17	92.45 ± 0.15	$\mathbf{92.73 \pm 0.03}$

Table 3: CIFAR10 Top-1 accuracy for different values of λ. ResNet110 GN model is used. We report mean and std across 3 seeds.

λ	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
	92.44 ± 0.17	$\mathbf{92.73 \pm 0.03}$	92.22 ± 0.21	92.53 ± 0.1	92.42 ± 0.4

each pair. Note that the presented results are randomly generated with no cherry-picking.

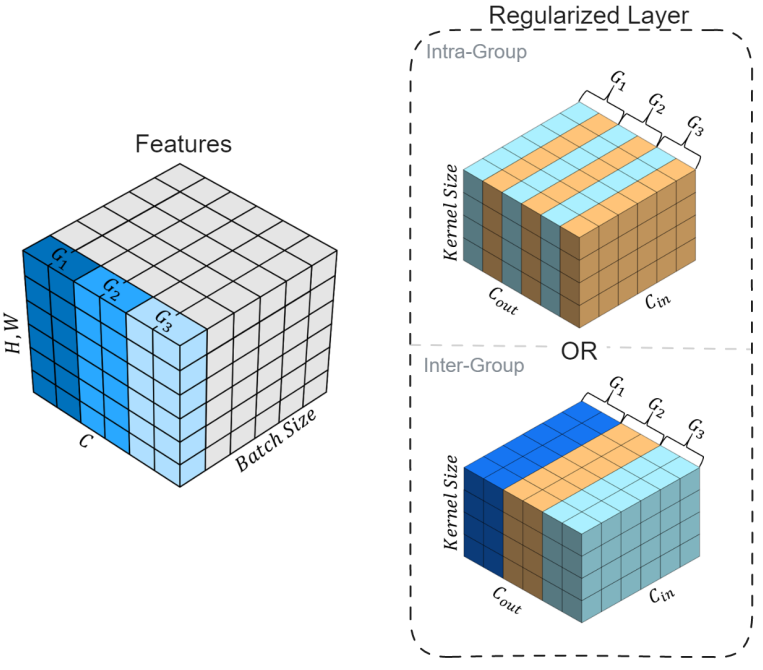

Figure 3: Partition of filters for GOR according to Inter-Group and Intra-Group for $N = 3$. Input features (left) are colored according to GN normalization with $G = 3$. Filters (right) are colored according to the sets orthogonality is enforced on. Best viewed in color.

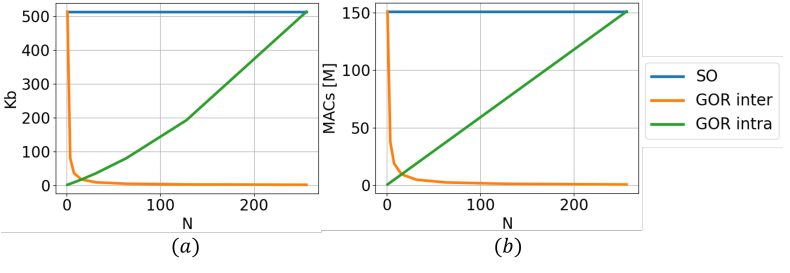

Figure 4: For different N (group size) values, we report (a) runtime, (b) multiply-accumulate (MAC). GOR improves over SO in terms of MACs and memory while getting accuracy improvement.

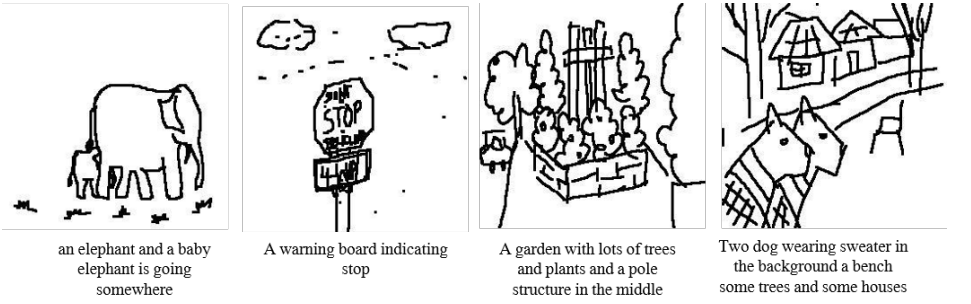

Figure 5: Examples of image-text pairs from the FS-COCO dataset

Figure 6: Qualitative comparisons on Pokemon-BLIP between baseline fine-tuned model and model fine-tuned along with GOR using the same seed. For each of the two rows: Top is LoRA baseline. Bottom is LoRA with our method.

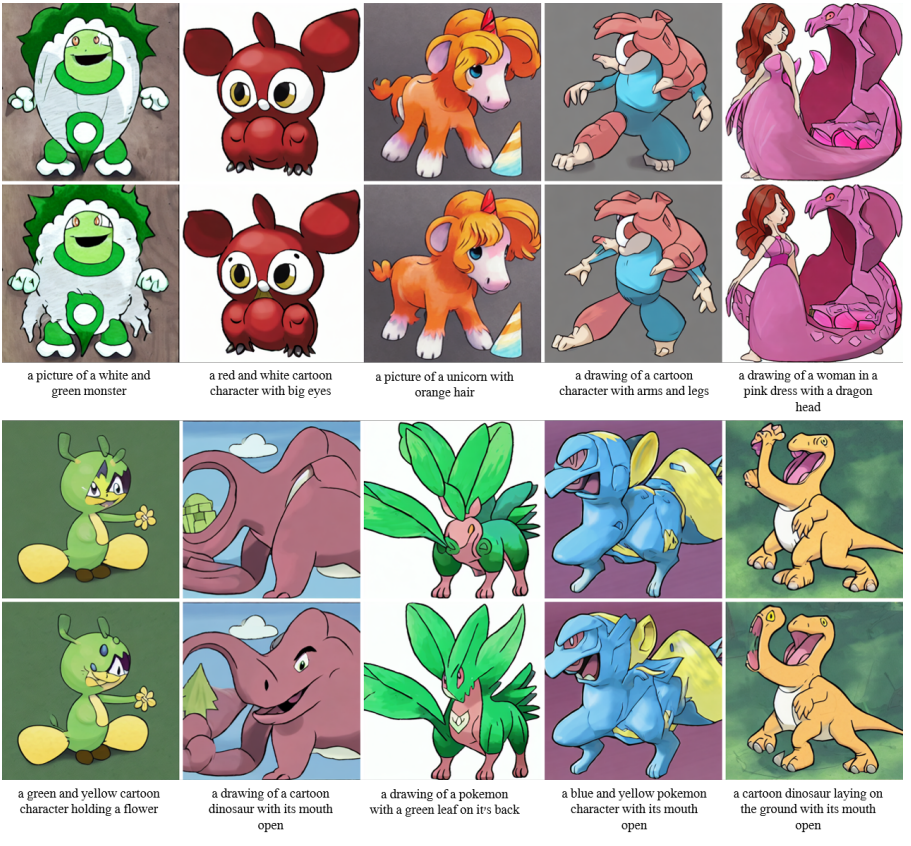

Figure 7: Qualitative comparisons on Pokemon-BLIP between baseline fine-tuned model and model fine-tuned along with GOR using same seed. For each of the two rows: Top is LoRA baseline. Bottom is LoRA with our method.

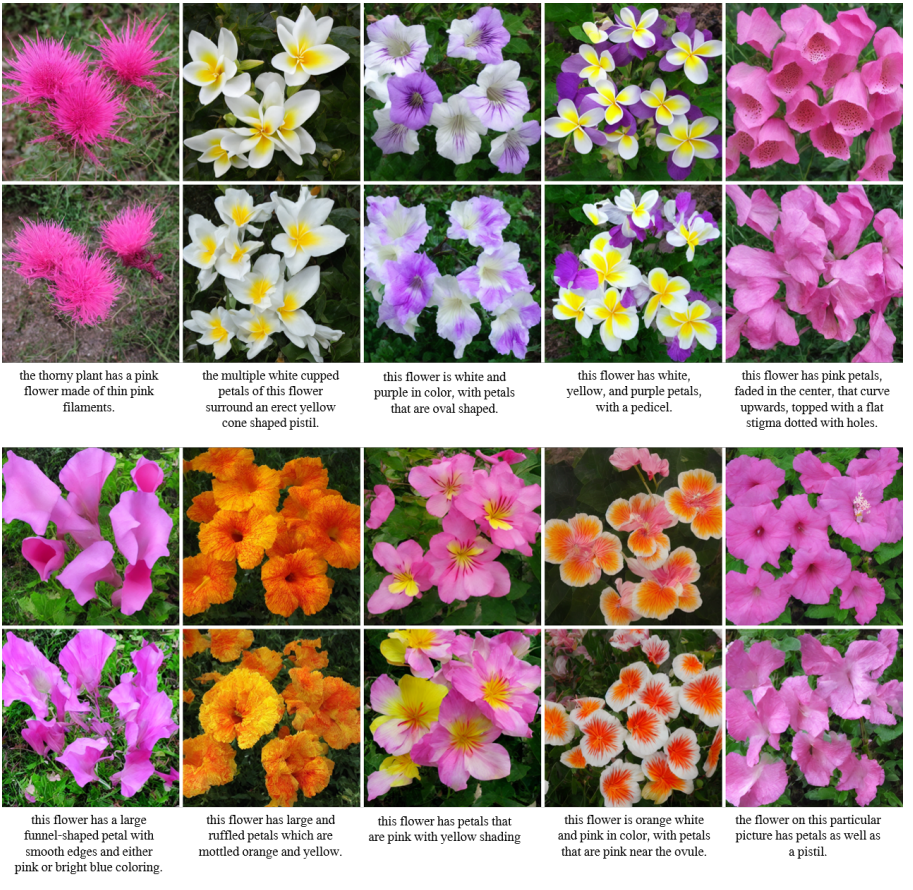

Figure 8: Qualitative comparisons on Oxford102 between baseline fine-tuned model and model fine-tuned along with GOR using the same seed. For each of the two rows: Top is LoRA baseline. Bottom is LoRA with our method.

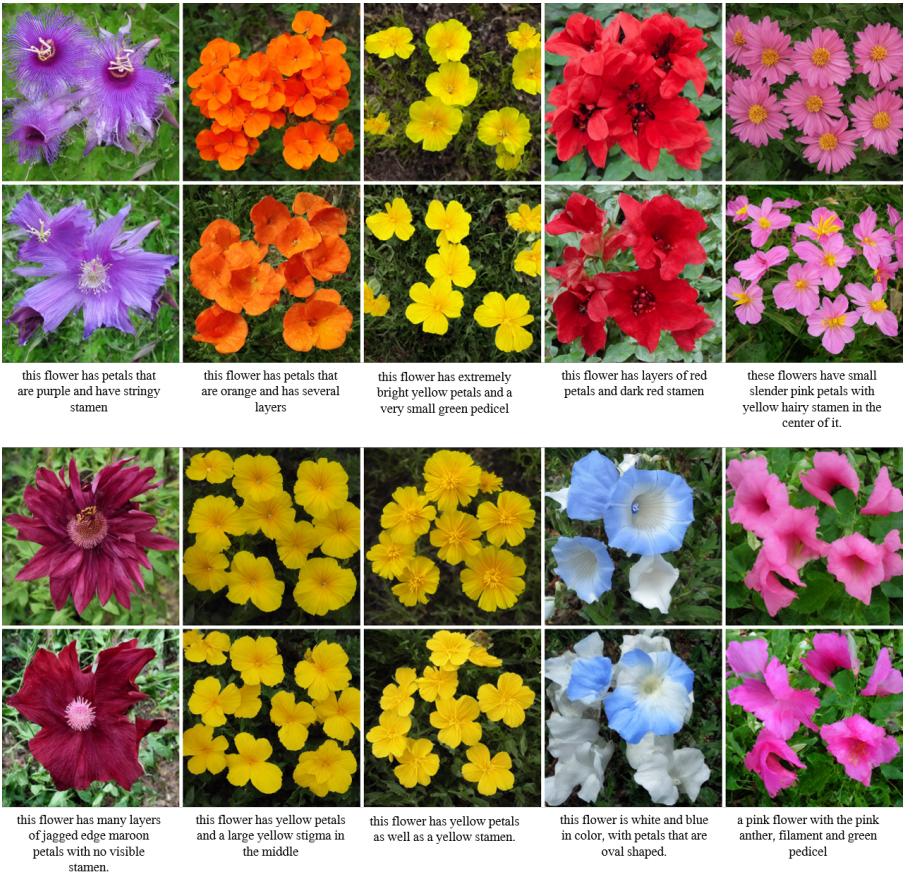

Figure 9: Qualitative comparisons on Oxford102 between baseline fine-tuned model and model fine-tuned along with GOR using the same seed. For each of the two rows: Top is LoRA baseline. Bottom is LoRA with our method.

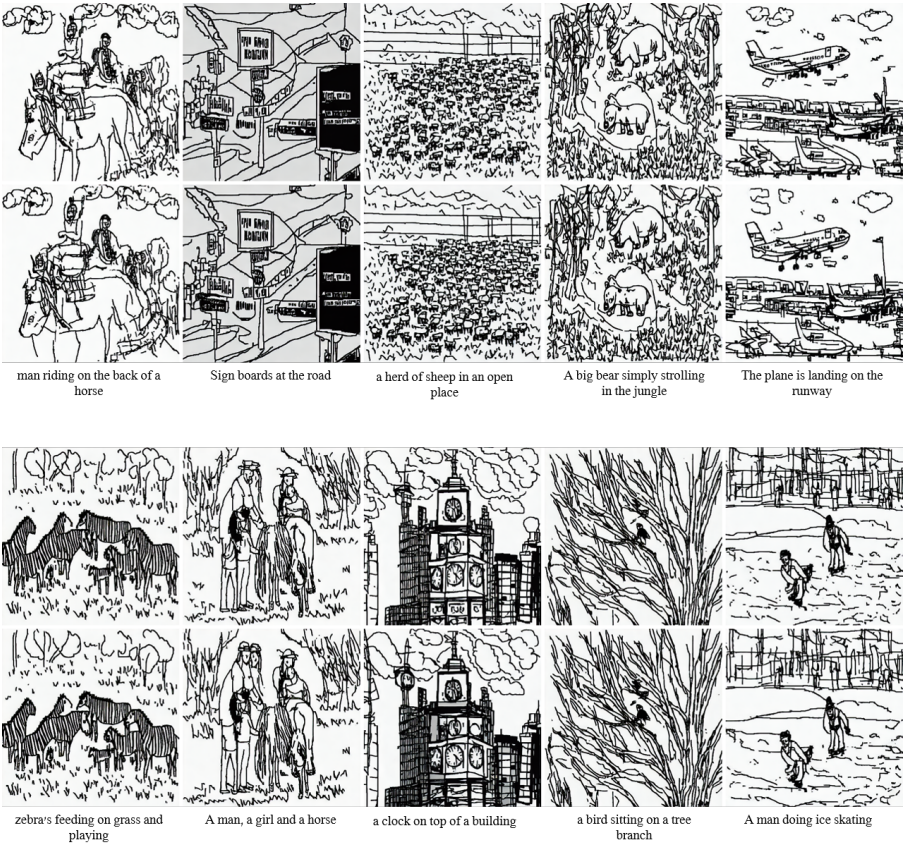

Figure 10: Qualitative comparisons on FS-COCO between baseline fine-tuned model and model fine-tuned along with GOR using the same seed. For each of the two rows: Top is LoRA baseline. Bottom is LoRA with our method.

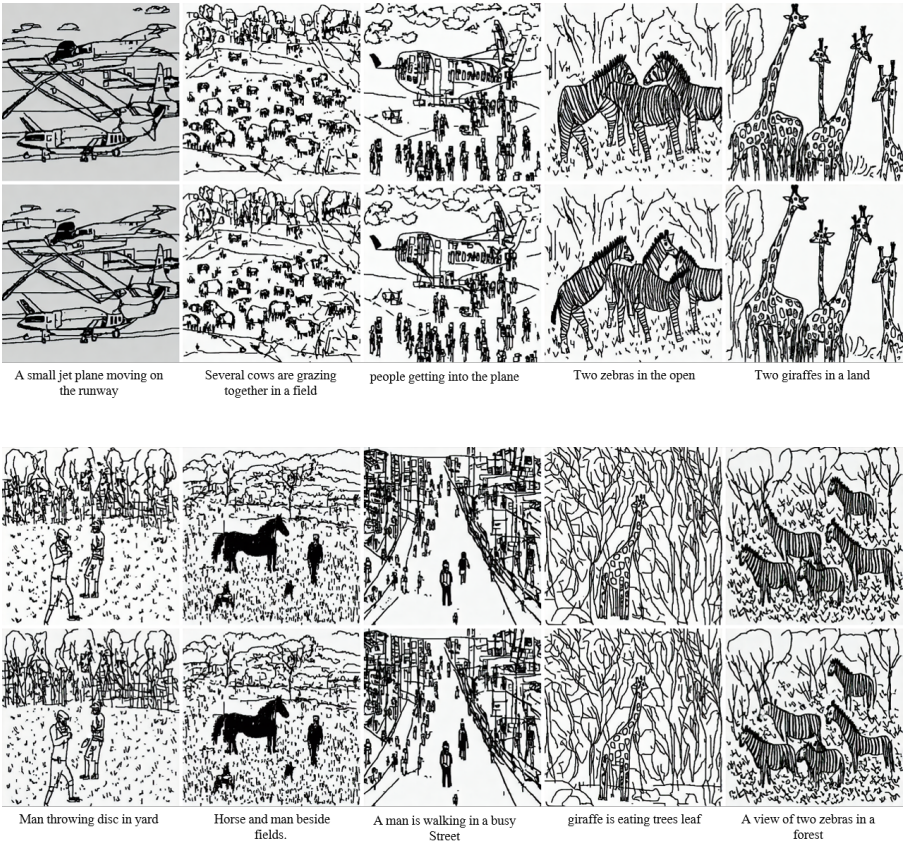

Figure 11: Qualitative comparisons FS-COCO between baseline fine-tuned model and model fine-tuned along with GOR using the same seed. For each of the two rows: Top is LoRA baseline. Bottom is LoRA with our method.

References

[1] Pinaki Nath Chowdhury, Aneeshan Sain, Ayan Kumar Bhunia, Tao Xiang, Yulia Gryaditskaya, and Yi-Zhe Song. Fs-coco: Towards understanding of freehand sketches of common objects in context. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part VIII*, pages 253–270, 2022.

[2] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *arXiv preprint arXiv:2206.00364*, 2022.

[3] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008.

[4] Justin N. M. Pinkney. Pokemon blip captions. https://huggingface.co/datasets/lambdalabs/pokemon-blip-captions/, 2022.

[5] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[6] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers, 2022.