Supplementary Material of ConsNoTrainLoRA: Data-driven Weight Initialization of Low Rank Adapters using Constraints

Debasmit Das * Hyoungwoo Park * Munawar Hayat Seokeon Choi Sungrack Yun Fatih Porikli
Qualcomm AI Research †

1. Additional Experimental Details

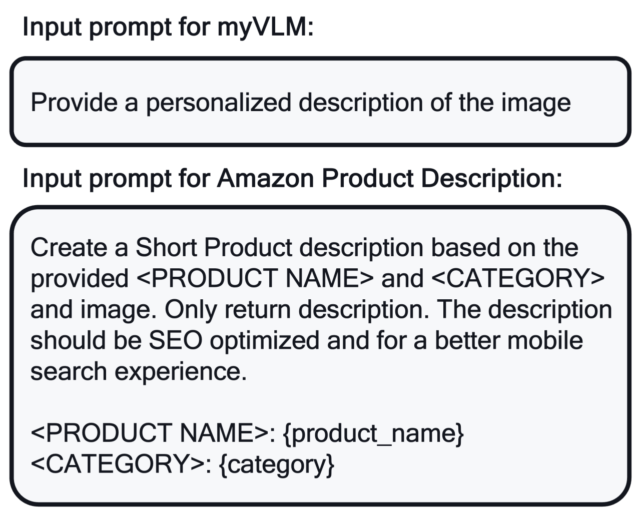
Image Generation For the image generation task, we consider the Dreambooth [6] dataset. This dataset consists of 30 image sets from 15 different categories in which each category consists of 4 to 6 images per concept. The subjects primarily fall into two types: living or non-living. Based on this distinction, 25 prompts are used for evaluation across four different seeds. For the training, we used the default learning rate of 1e-4 (AdamW optimizer), with batch size of 1, with prior preservation loss weight of 0.1. The LoRA rank is set to 128 with $\alpha = 256$ and is applied to attention layers of the text encoder, as well as to both the self-attention and cross-attention layers of the UNet of Stable Diffusion V1.5. LoRA is attached to the key, query, value, and output projection matrices in the attention layers of both the UNet and the text encoder. Training is conducted over 1000 iterations per concept.

Image Classification For image classification, we fine-tune the DINOv2-g/14 model [4] using VTAB-1K [10], which consists of 19 classification tasks spanning natural, specialized, and structured domains. In our experiments, we select 14 tasks from VTAB-1K to assess fine-tuning performance. To incorporate AdaLoRA, PiSSA, OLoRA, EVA, DoRA, and CoRDA, we adapt their implementations from the `peft` library. The classifier head is initialized with weights drawn from a normal distribution ($\sigma = 2e^{-5}$), while biases are set to zero. Throughout fine-tuning, we update the classification head, LoRA matrices, and biases. LoRA matrices are applied to most linear layers, especially query, key, and value components of attention layers, as well as the dense and fully connected layers. Input images are rescaled to 224×224 using bicubic interpolation and normalized according to ImageNet's per-channel mean and variance. Fine-tuning is performed with bfloat16 precision, using AdamW (weight decay = 0.05) for 30 epochs. The learning rate follows a cosine decay schedule, with a linear warm-up phase spanning the first three epochs. For full fine-tuning, a layer-wise learning rate decay of 0.75 is applied.

*These authors contributed equally to this work.
†Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

Image Understanding For image understanding, we consider Amazon Product Description dataset (APD) [7] for product marketing. The APD dataset is a large-scale resource designed for product marketing and e-commerce applications, encompassing both structured metadata and unstructured textual descriptions across a wide range of product categories. For fine-tuning, we use the GLM-Edge model [9], incorporating LoRA-based adaptation with a rank of 32 and $\alpha = 64$, optimizing it to generate domain-specific product descriptions. The attachment points are done at query, key and value locations of all attention layers in the vision encoder as well as the language model. Additionally, we utilize the myVLM dataset [1], which is tailored for personalized vision-language modeling, focusing on concept-based captioning where descriptions are tailored to user-defined preferences and contexts. The dataset comprises manually annotated image-text pairs, ensuring high-quality supervision for customized caption generation and multimodal retrieval tasks. We fine-tune LLAVA-v1.5-7B [2], a vision-language model, applying adapters with a rank of 128 and $\alpha = 256$ to the self-attention layers in the visual encoder and both the self-attention and feed-forward network (FFN) layers in the language model. For the attention blocks, LoRA is attached at the query, key and value locations. In both the models, training is conducted using AdamW optimizer with a learning rate of 2e-4 and batch size of 8. For the both tasks, fine-tuning is performed over 20 epochs using all available samples. Fig. 1 shows the input prompts used in the datasets.

Language Understanding For language understanding, we consider the popular GLUE Benchmark [8]. This benchmark consists of eight downstream tasks, such as natural language inference, or sentiment analyses. For the base model, we use the large version of Roberta [3]. The hyperparameters used are the same as EVA [5]. For the evaluation metrics, we report accuracy for all tasks except for CoLA and STS-B, where we report Matthew's correlation and Pearson's correlation, respectively.


Input prompt for myVLM:

Provide a personalized description of the image

Input prompt for Amazon Product Description:

Create a Short Product description based on the provided <PRODUCT NAME> and <CATEGORY> and image. Only return description. The description should be SEO optimized and for a better mobile search experience.

<PRODUCT NAME>: {product_name}
<CATEGORY>: {category}

Figure 1. Figure showing the input prompts used in the myVLM and Amazon Product Description datasets.

2. Gradient Analysis

Our proposed initialization method requires initialization of $\boldsymbol{A}^i$, the down matrices and $\boldsymbol{B}^i$, the up matrices separately. The gradients of $\boldsymbol{A}^i$ and $\boldsymbol{B}^i$ with respect to the task loss L is given as $\frac{\partial L}{\partial \boldsymbol{A}^i} = \boldsymbol{B}^{iT}\left(\frac{\partial L}{\partial \boldsymbol{Y}}\right)\boldsymbol{X}_{tar}^{iT}$, where $\boldsymbol{Y}$ is the output of the LoRA layer. Similarly, the gradient of the task loss L with respect to $\boldsymbol{B}^i$ is given as $\frac{\partial L}{\partial \boldsymbol{B}^i} = \left(\frac{\partial L}{\partial \boldsymbol{Y}}\right)\boldsymbol{X}_{tar}^{iT}\boldsymbol{A}^{iT}$. From the expression, it is clear that the initial gradients would depend on the initial values of $\boldsymbol{A}^i$ and $\boldsymbol{B}^i$. For random initialization, $\frac{\partial L}{\partial \boldsymbol{A}^i}$ would be zero as $\boldsymbol{B}^i$ is initialized to $\boldsymbol{0}$. This can slow down the convergence. For data-driven initialization like EVA, $\boldsymbol{A}^i$ is initialized from data and it is likely to assist $\frac{\partial L}{\partial \boldsymbol{B}^i}$ in the convergence. However, still $\boldsymbol{B}^i$ is initialized to 0 and hence the initial gradients of $\frac{\partial L}{\partial \boldsymbol{A}^i}$ would be zero and hence affecting convergence. For our proposed method, the initial values of $\boldsymbol{B}^i$ and $\boldsymbol{A}^i$ are non-zero and both depend on the principal components of the input activations $\boldsymbol{X}_{tar}^i$. Consequently, both of the gradients have a dependency on $\boldsymbol{X}_{tar}^i$ beyond first order and hence we expect faster convergence compared to EVA.

3. Competitive Low Rank Adaptors as Constraints

The first example of **Native LoRA** is as follows:

$$\boldsymbol{F}_1(\boldsymbol{X}_{src}^i, \boldsymbol{X}_{tar}^i, \boldsymbol{W}_{src}^i, \boldsymbol{W}_{tar}^i) = \boldsymbol{0} \tag{1}$$
$$\boldsymbol{W}_{tar}^i - \boldsymbol{W}_{src}^i = \boldsymbol{0} \implies \Delta\boldsymbol{W}^i = \boldsymbol{0} \tag{2}$$
$$\implies \boldsymbol{B}^i = \boldsymbol{0}, \boldsymbol{A}^i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}) \tag{3}$$

PISSA can be expressed as:

$$\boldsymbol{F}_1(\boldsymbol{X}_{src}^i, \boldsymbol{X}_{tar}^i, \boldsymbol{W}_{src}^i, \boldsymbol{W}_{tar}^i) = \boldsymbol{0} \tag{4}$$
$$\boldsymbol{W}_{tar}^i - 2\boldsymbol{W}_{src}^i = \boldsymbol{0} \implies \Delta\boldsymbol{W}^i = \boldsymbol{W}_{src}^i \tag{5}$$
$$\Delta\boldsymbol{W}^i = \boldsymbol{U}^i\boldsymbol{S}^i\boldsymbol{V}^i \implies \boldsymbol{B}^i = \boldsymbol{U}^i[::r]\boldsymbol{S}^i[:r]^{0.5}, \tag{6}$$
$$\boldsymbol{A}^i = \boldsymbol{S}^i[:r]^{0.5}\boldsymbol{V}^i[:r:] \tag{7}$$

OLORA can be expressed as:

$$\boldsymbol{F}_1(\boldsymbol{X}_{src}^i, \boldsymbol{X}_{tar}^i, \boldsymbol{W}_{src}^i, \boldsymbol{W}_{tar}^i) = \boldsymbol{0} \tag{8}$$
$$\boldsymbol{W}_{tar}^i - 2\boldsymbol{W}_{src}^i = \boldsymbol{0} \implies \Delta\boldsymbol{W}^i = \boldsymbol{W}_{src}^i \tag{9}$$
$$\Delta\boldsymbol{W}^i = \boldsymbol{Q}^i[::r]\boldsymbol{R}^i[:r:] \implies \boldsymbol{B}^i = \boldsymbol{Q}^i[::r], \tag{10}$$
$$\boldsymbol{A}^i = \boldsymbol{R}^i[:r:] \tag{11}$$

CORDA can be expressed as:

$$\boldsymbol{F}_1(\boldsymbol{X}_{src}^i, \boldsymbol{X}_{tar}^i, \boldsymbol{W}_{src}^i, \boldsymbol{W}_{tar}^i) = \boldsymbol{0} \tag{12}$$
$$(\boldsymbol{W}_{tar}^i - \boldsymbol{W}_{src}^i)\boldsymbol{X}_{tar}^i\boldsymbol{X}_{tar}^{iT} - \text{SVD}((\boldsymbol{W}_{tar}^i)\boldsymbol{X}_{tar}^i\boldsymbol{X}_{tar}^{iT}) = \boldsymbol{0} \tag{13}$$
$$\implies \Delta\boldsymbol{W}^i\boldsymbol{C}_{tar}^i = \text{SVD}(\boldsymbol{W}_{tar}^i\boldsymbol{C}_{tar}^i) \tag{14}$$
$$\implies \Delta\boldsymbol{W}^i = \text{SVD}(\boldsymbol{W}_{tar}^i\boldsymbol{C}_{tar}^i)\boldsymbol{C}_{tar}^{i\;-1} \tag{15}$$
$$\implies \boldsymbol{A}^i = (\boldsymbol{S}^i)^{0.5}[:r](\boldsymbol{V}^i\boldsymbol{C}_{tar}^i)^{-1}[:r:] \tag{16}$$
$$\implies \boldsymbol{B}^i = \boldsymbol{U}^i[::r](\boldsymbol{S}^i)^{0.5}[:r] \tag{17}$$

EVA can be expressed as:

$$\boldsymbol{F}_1(\boldsymbol{X}_{src}^i, \boldsymbol{X}_{tar}^i, \boldsymbol{W}_{src}^i, \boldsymbol{W}_{tar}^i) = \boldsymbol{0} \tag{18}$$
$$\boldsymbol{A}^i = \boldsymbol{V}^i[:r:] \quad \text{where} \tag{19}$$
$$\boldsymbol{U}^i[::r], \boldsymbol{S}^i[:r], \boldsymbol{V}^i[:r:] = \text{SVD}(\boldsymbol{X}_{tar}^i) \tag{20}$$

4. Additional Experiments

4.1. Image generation

Effect of different ranks We also study how different ranks affect quantitative performance of the model as shown in Table 1. We have originally used the default rank of 128, on which our proposed method and their variations CNTLoRA-X, CNTLoRA-S, CNTLoRA-Sh produced better DINO scores than competitive methods. The pattern holds true even for lower ranks of 64 and 32. In fact, CNTLoRA-X at rank 64 produces better DINO scores than what LoRA and EVA produces at rank of 128.

Effect of different learning rates We also study how different learning rates affect quantitative performance of the model as shown in Table 2. We have originally used the default learning rate of 1e-4, on which our proposed method and their variations CNTLoRA-X, CNTLoRA-S, CNTLoRA-Sh produce better DINO scores than competitive methods. The pattern holds true even for different learning rates of 1e-3 and 1e-5. For both these learning rates

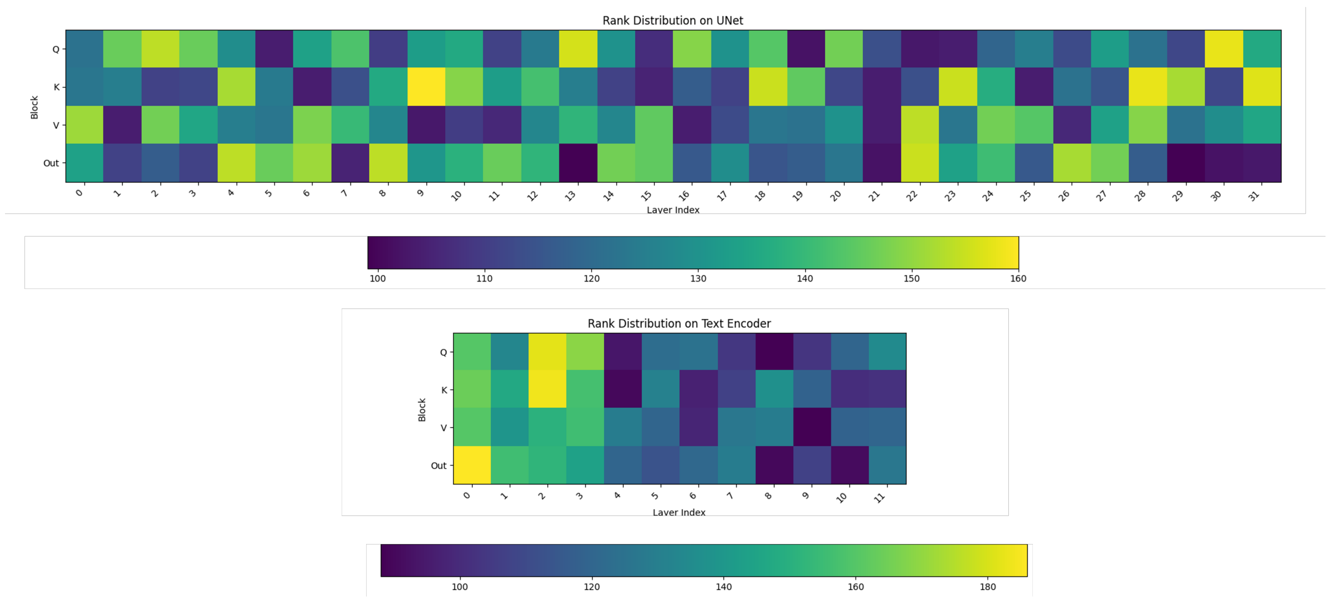


Figure 2. Plot showing how the rank distribution takes place for the dog class in the Dreambooth dataset.

Table 1. Final quantitative results (DINO) (1000 iterations) of different methods on the Dreambooth dataset using SD 1.5 for different ranks. Higher (↑) is better.

Method	**128**	**64**	**32**
LoRA	62.74	59.07	56.62
EVA	62.24	60.93	57.41
CNTLoRA-X	64.63	62.98	59.09
CNTLoRA-S	63.91	62.25	58.23
CNTLoRA-Sh	62.95	61.02	57.69

we see a general drop in performance compared to that at 1e-3. The reason for drop at 1e-5 is because of slower learning rate and not enough iterations to converge. The reason for drop in learning rate at 1e-3 is due to instability in the fine-tuning procedure.

Table 2. Final quantitative results (DINO) (1000 iterations) of different methods on the Dreambooth dataset using SD 1.5 for different learning rates.

Method	**1e-4**	**1e-3**	**1e-5**
LoRA	62.74	60.51	58.42
EVA	62.24	60.26	59.01
CNTLoRA-X	64.63	61.98	60.05
CNTLoRA-S	63.91	61.92	60.22
CNTLoRA-Sh	62.95	60.81	59.10

Rank distribution in VAS In our proposed Variable Adapter Structure (VAS) framework, we have variable ranks across different attachment points. In Fig. 2, we visualize this variable rank allocation across attachment points for both the UNet and Text Encoder when finetuned on the dog class of the Dreambooth dataset. For the UNet case, we see that the rank is distributed more or less randomly across all attachment points. For the "Out" attachment point in the attention block, we see lower rank allocation in deeper layers suggesting lesser importance of LoRA modules in that region. For the text encoder case, we see higher rank allocation in the initial layers while lower rank allocation in the deeper layers of the text encoder. This suggests lesser importance of LoRA modules in the deeper layer.

Table 3. Final quantitative results (1000 iterations) of different variations of our method on the Dreambooth dataset using SD 1.5.

Method	**DINO** (↑)	**CLIP-I** (↑)	**CLIP-T** (↑)
CNTLoRA-X-QR	65.29	78.63	27.36
CNTLoRA-X-0.75	**65.78**	**81.34**	24.91
CNTLoRA-X-0.25	63.26	79.71	27.01
CNTLoRA-X-0.1	62.41	78.23	**28.34**
CNTLoRA-Sh-Norm.	61.2	79.34	28.10
CNTLoRA-Sh-10	62.13	79.92	28.13
CNTLoRA-Sh-0.1	62.42	79.96	27.95

Additional Variations In Table 3, we consider different variations of CNTLoRA. CNTLoRA-X-QR uses QR decomposition instead of SVD for allocating the up and down matrices. CNTLoRA-X-p uses fractional allocation of p when splitting the singular matrix $\boldsymbol{S}$ and allocating it to up and down matrices. By default, we use $p = 0.5$. Furthermore, we consider different variations of CNTLoRA-Sh where the constant $\boldsymbol{C}$ is varied from the default value of identity: (a) *Norm* where $\boldsymbol{C}$ is normally distributed. (b) *10*

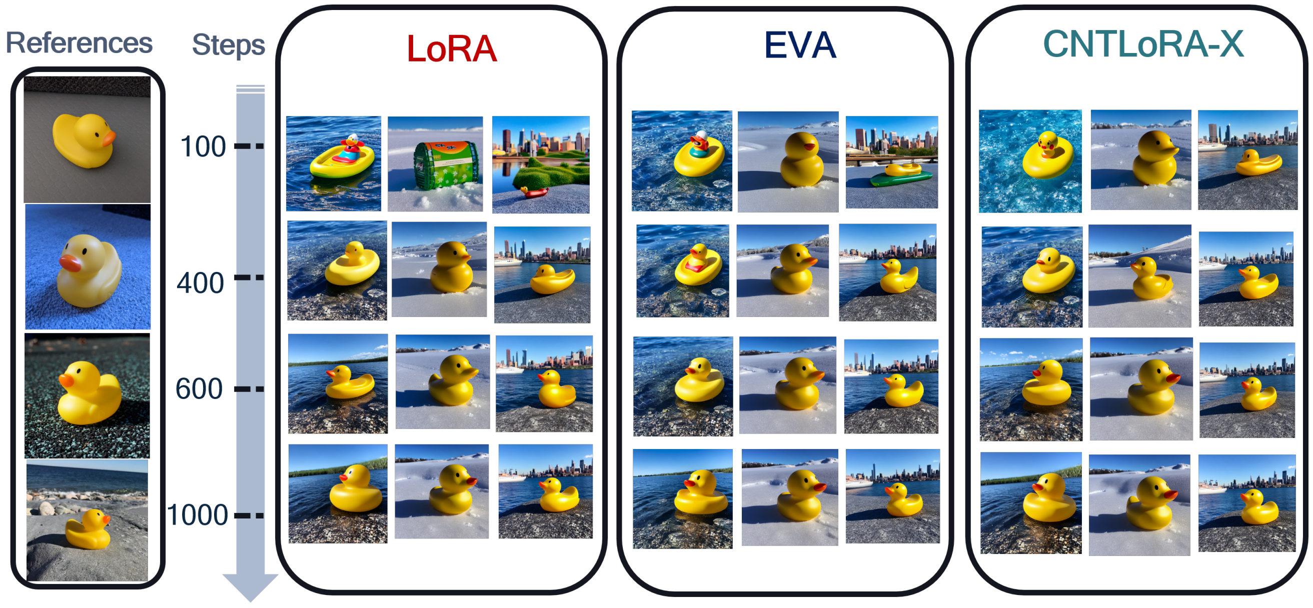


Figure 3. Plot showing how the generation evolves for different initialization methods on the Dreambooth dataset for the duck class. The prompts from left to right are (a) A S* toy floating on top of water. (b) A S* toy in the snow. (c) A S* toy with a city in the background.

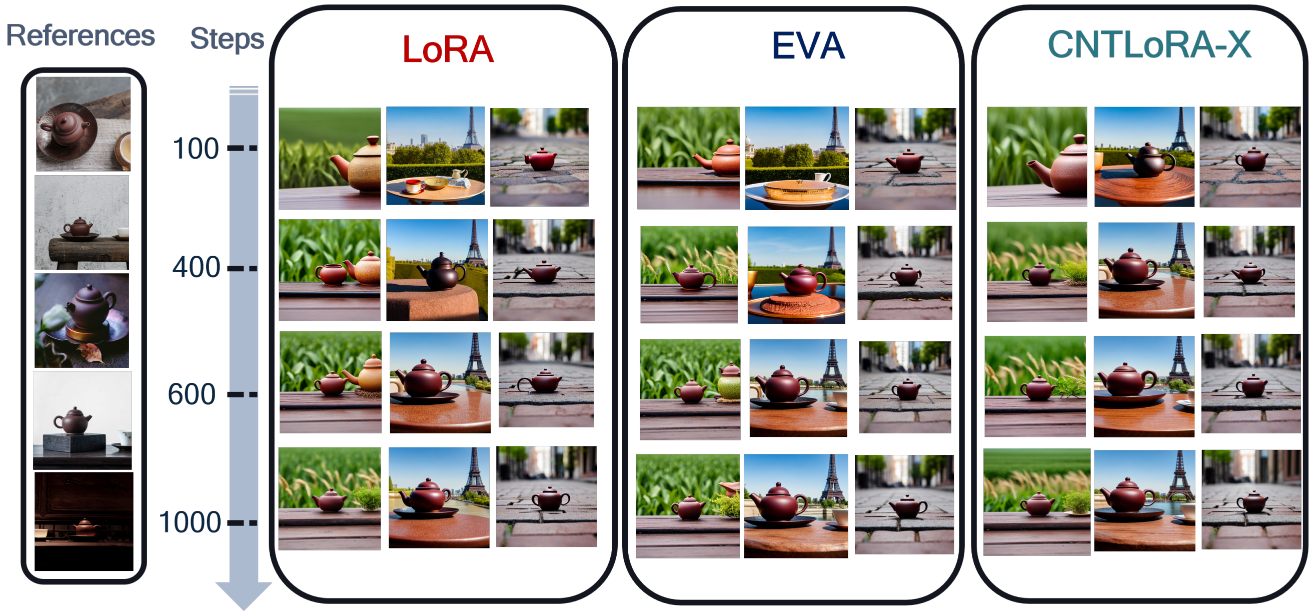


Figure 4. Plot showing how the generation evolves for different initialization methods on the Dreambooth dataset for the teapot class. The prompts from left to right are (a) A S* teapot with a wheat field in the background. (b) A S* teapot with the Eiffel tower in the background. (c) A S* teapot on a cobblestone street.

where $\boldsymbol{C}$ is 10 times identity (c) *0.1* where $\boldsymbol{C}$ is 0.1 times identity. Overall, we see that the DINO score and CLIP-T score of CNTLoRA-X-QR improves over CNTLoRA-X. This might be due to the fact that for QR decomposition, there is no fractional allocation of singular matrix between up and down matrices and hence there is balance between image and text fidelity quite well. In fact, the fractional value p can allow us to balance image and text fidelity well. As seen for higher fraction value of 0.75, DINO score increases while CLIP-T score decreases. If we decrease the fraction value to 0.25, DINO score decrease to 63.26 while the CLIP-T score increases to 27.01. When the fraction value is decreased further to 0.1, DINO score decreases to 62.41 while CLIP-T increases to 28.04. Hence,

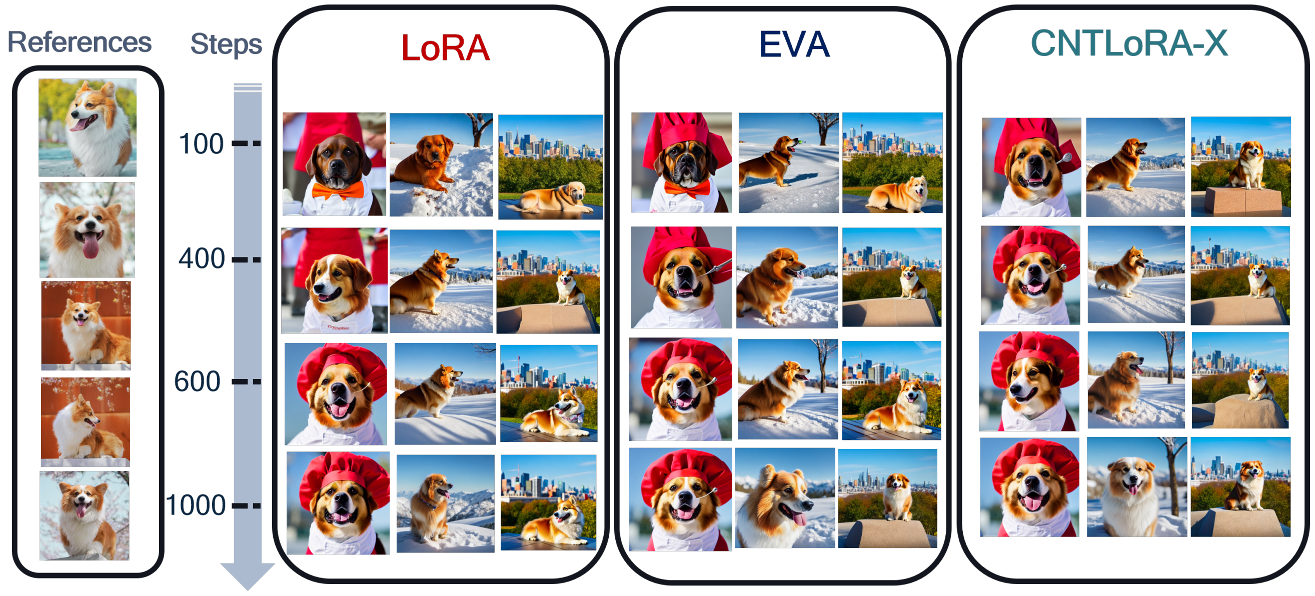


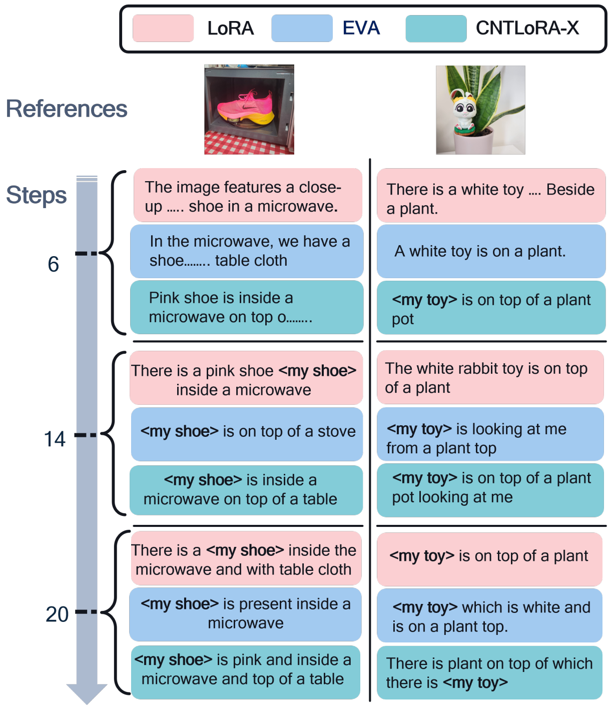
Figure 5. Plot showing how the generation evolves for different initialization methods on the Dreambooth dataset for the dog class. The prompts from left to right are (a) A S* dog in a chef outfit. (b) A S* dog in the snow. (c) A S* dog with a city in the background.

our proposed fractional allocation can maintain a balance between image and text fidelity. As for different variants of CNTLoRA-Sh, the newer hyperparameters of C seems to produce poorer image fidelty performance. However, they lead to better prompt fidelity.

Qualitative Results We also show qualitative results on additional prompts for the duck, teapot and dog classes in Figs. 3, 4 and 5 respectively. In Fig. 3, we observe that at 100 iterations of training, LoRA (i.e. random initialization) produces poor performance in terms of image fidelity. However, for both EVA and CNTLoRA-X, we observe better image fidelity due to data-driven initialization. Infact, for the prompt "A S* toy floating on top of water", CNTLoRA-X produces better image fidelity where the face of the duck atleast appears compared to that of EVA.

In Fig. 4, we observe that at 100 iterations of training, LoRA (i.e. random initialization) and EVA produces poor performance in terms of image fidelity for the prompt "A S* teapot with the Eiffel tower in the background.". However, for CNTLoRA-X, we observe better image fidelity. The pattern is repeated even for the prompt "A S* teapot with a wheat field in the background."

In Fig. 5, we observe that at 100 iterations of training, LoRA (i.e. random initialization) and EVA produces poor image fidelity performance for the prompts "A S* dog in a chef outfit." and "A S* dog with a city in the background.". However, for CNTLoRA-X, we observe better image fidelity. Even at 400 iterations of training, CNTLoRA-X produces better image fidelity for the prompt "A S* dog in the snow".



Steps			
6	LoRA	The image features a close-up ..... shoe in a microwave.	There is a white toy .... Beside a plant.
6	EVA	In the microwave, we have a shoe........ table cloth	A white toy is on a plant.
6	CNTLoRA-X	Pink shoe is inside a microwave on top o........	**<my toy>** is on top of a plant pot
14	LoRA	There is a pink shoe **<my shoe>** inside a microwave	The white rabbit toy is on top of a plant
14	EVA	**<my shoe>** is on top of a stove	**<my toy>** is looking at me from a plant top
14	CNTLoRA-X	**<my shoe>** is inside a microwave on top of a table	**<my toy>** is on top of a plant pot looking at me
20	LoRA	There is a **<my shoe>** inside the microwave and with table cloth	**<my toy>** is on top of a plant
20	EVA	**<my shoe>** is present inside a microwave	**<my toy>** which is white and is on a plant top.
20	CNTLoRA-X	**<my shoe>** is pink and inside a microwave and top of a table	There is plant on top of which there is **<my toy>**

Figure 6. Figure showing how image captions evolve with increasing number of steps for different conditioning input images. The samples are from myVLM dataset. Presence of <*> in the generated outputs suggest that the concept has been identified. Furthermore, generation of concise prompts suggest that the model has been well fine-tuned.

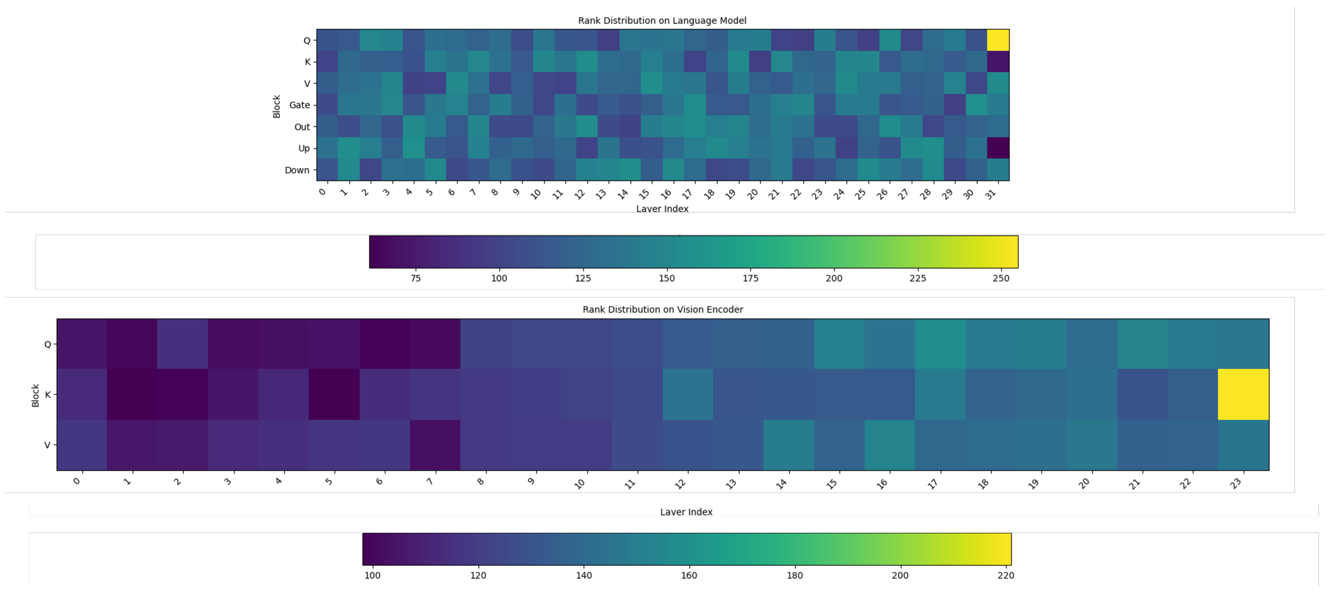


Figure 7. Plot showing how the rank distribution takes place when fine-tuned on the myVLM dataset.

4.2. Image Understanding

Qualitative results In Fig. 6, we observe how the captions evolve for different training steps i.e. 6, 14, 20 for the <my shoe> and <my toy> object. Our proposed framework CNTLoRA-X can recognize the personalized toy even at 6 steps. Even at 20 steps, it produces a more descriptive caption compared to LoRA and EVA. For the personalized shoe, our proposed method can produce more descriptive and accurate captions at Step 14 and Step 20. At step 6, even though the personalized object is not identified, our method CNTLoRA-X produces more accurate descriptions.

Rank Distribution in VAS In our proposed Variable Adapter Structure (VAS) framework, we have variable ranks across different attachment points. In Fig. 7, we visualize this variable rank allocation across attachment points for both the language model and vision encoder when fine-tuned on the myVLM dataset. For the language model case, we see that the rank is distributed more or less randomly across all attachment points. For the vision encoder case, we see higher rank allocation in the later layers while lower rank allocation in the shallower layers of the text encoder. This suggests that the fine-tuning image dataset has not very different distribution from pre-training dataset. Rather, the language style is changed and adapters need to be learned mainly for the language model.

4.3. Language Understanding

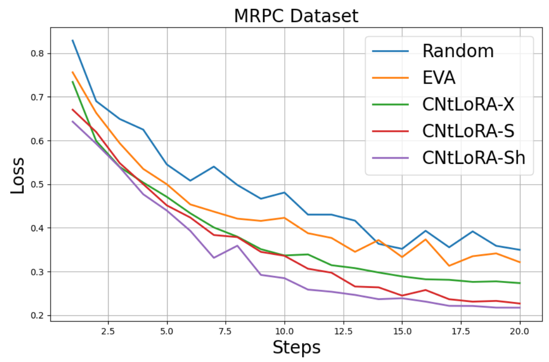
Quantitative results We compare our methods against existing parameter-efficient fine-tuning approaches on the GLUE benchmark, which includes diverse language understanding tasks. Also, we compare our method combined with QR decomposition simultaneously. As shown in Table. 4, most of CNTLoRA variants consistently achieve high performance. Especially, CNTLoRA-X-SVD consistently achieves the best or near-best performance in various datasets. These results highlight that CNTLoRA achieves strong performance not only when using SVD but also when initialized with QR decomposition, demonstrating the effectiveness of both approaches in enhancing fine-tuning.

Training curve We present the training curves for MRPC dataset and RTE dataset of the GLUE Benchmark in Figures 8a and 8b, respectively showing that our method achieves faster convergence compared to LoRA and EVA, particularly in the early epochs. We observe that our initialization better preserves pre-trained knowledge, allowing for a more stable adaptation to downstream tasks. We also find that our model maintains consistently lower training loss curves across multiple tasks, demonstrating improved fine-tuning stability.

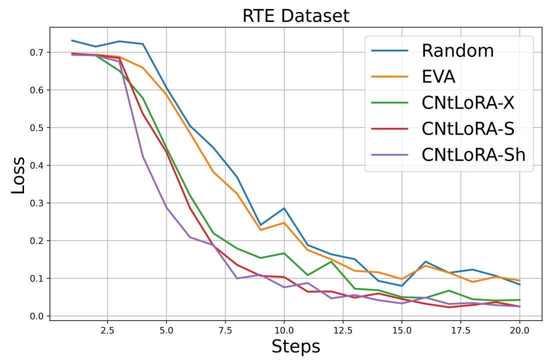
Performance convergence We present the performance convergence results for MRPC and RTE of the GLUE Benchmark in Figures 9a and 9b, showing that our method achieves more stable performance compared to LoRA and EVA, with reduced fluctuations during fine-tuning. We observe that our initialization accelerates convergence, allowing the model to reach optimal performance with fewer fine-tuning steps. We also find that our approach consistently achieves higher final accuracy, demonstrating its effectiveness in enhancing fine-tuning efficiency and stability.

Table 4. Comparison of all methods for RoBERTa$_{\text{Large}}$ [3] on GLUE tasks. We report Matthew's correlation for CoLA, Pearson correlation for STS-B, matched accuracy for MNLI, and accuracy for remaining tasks.

Method	MNLI	QNLI	QQP	SST2	CoLA	MRPC	RTE	STS-B	Avg
FFT	90.2	94.7	92.2	96.4	68.0	90.9	86.6	92.4	88.93
LoRA	90.7	94.8	92.0	96.2	69.1	91.1	88.1	92.3	89.29
AdaLoRA	90.5	94.8	90.6	96.1	68.2	90.7	84.4	91.8	88.39
PiSSA	90.1	94.7	92.2	96.1	68.7	90.4	87.6	92.5	88.89
OLoRA	90.9	95.0	92.0	96.3	69.0	91.0	87.9	92.4	89.32
EVA	90.8	95.0	92.1	96.2	69.5	91.4	88.8	92.6	89.55
DoRA	89.5	94.6	89.9	96.1	69.3	91.0	88.4	92.4	88.90
CNTLoRA-X-SVD	**91.1**	96.1	**93.1**	96.4	69.7	91.6	89.1	**93.2**	**90.03**
CNTLoRA-X-QR	90.9	**96.2**	92.9	95.1	69.5	91.2	88.9	92.1	89.6
CNTLoRA-S-SVD	90.2	95.9	92.2	96.1	**70.1**	91.8	88.2	92.8	89.6
CNTLoRA-S-QR	90.8	95.7	92.8	**97.1**	69.6	91.6	88.8	92.6	89.8
CNTLoRA-Sh-SVD	90.9	95.3	92.3	96.4	69.8	91.3	**89.5**	92.4	89.7
CNTLoRA-Sh-QR	90.6	95.0	92.1	96.3	69.5	**92.4**	89.1	92.5	89.7

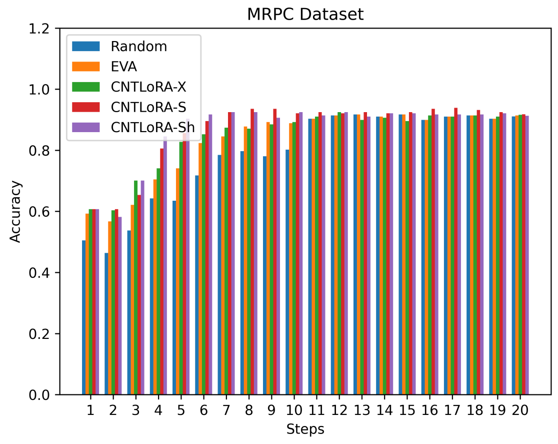


(a) Training loss for different initialization methods on the MRPC dataset.

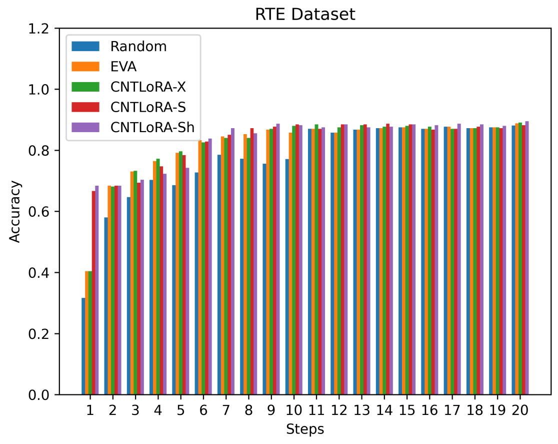


(b) Training loss for different initialization methods on the RTE dataset.

Figure 8. Plots showing how the training loss varies with different epochs for different initialization methods on the MRPC and RTE datasets.



(a) Accuracy for different initialization methods on the MRPC dataset.



(b) Accuracy for different initialization methods on the RTE dataset.

Figure 9. Plots showing how the accuracy varies with different epochs for different initialization methods on the MRPC and RTE datasets.

References

[1] Yotam Alaluf, Elad Richardson, Sergey Tulyakov, Kfir Aberman, and Daniel Cohen-Or. Myvlm: Personalizing vlms for user-specific queries. In *European Conference on Computer Vision (ECCV)*, pages 73–91. Springer, 2024. 1

[2] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023. 1

[3] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. 1, 7

[4] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 1

[5] Fabian Paischer, Lukas Hauzenberger, Thomas Schmied, Benedikt Alkin, Marc Peter Deisenroth, and Sepp Hochreiter. One initialization to rule them all: Fine-tuning via explained variance adaptation. *arXiv preprint arXiv:2410.07170*, 2024. 1

[6] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22500–22510, 2023. 1

[7] Philipp Schmid. Amazon product descriptions vlm, 2024. Accessed: 2025-02-14. 1

[8] A Wang, A Singh, J Michael, F Hill, O Levy, and SR Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. arxiv preprint arxiv: 180407461, 2018. 1

[9] Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, et al. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *CoRR*, 2024. 1

[10] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019. 1