ROSA: Random Orthogonal Subspace Adaptation

Marawan Gamal Abdel Hameed [1] **Guillaume Rabusseau** [1 2]

Abstract

Model training requires significantly more memory, compared with inference. Parameter efficient fine-tuning (PEFT) methods provide a means of adapting large models to downstream tasks using less memory. However, existing methods either introduce latency overhead at inference time or achieve subpar downstream performance compared with full fine-tuning. In this work we propose Random Orthogonal Subspace Adaptation (ROSA), a method that exceeds the performance of previous PEFT methods by a significant margin, while maintaining a zero latency overhead during inference time. In contrast to previous methods, ROSA is able to adapt subspaces of larger size, without consuming additional memory during runtime. As PEFT methods are especially useful in the natural language processing domain. We evaluate ROSA by finetuning GPT2 on various Natural Language Generation (NLG) tasks. Our code is publicly available at github.com/marawangamal/rosa

1. Introduction

The advent of large language models pre-trained on web-size corpus (PLMs) has led to remarkably performant models in the natural language processing domain (Brown et al., 2020; Devlin et al., 2019). As the size of such models ranges from hundreds of millions to hundreds of billions of parameters (Touvron et al., 2023), adapting them to downstream tasks is challenging (Peng et al., 2023). Compared with inference, training requires substantially more memory. For example, a GPT2 model (128M parameters) together with an input batch of 8 sequences of length 512 requires 670MB and 1139MB during inference and training respectively (Radford et al., 2019).

[1]DIRO & Mila, Université de Montréal [2]CIFAR AI chair. Correspondence to: Marawan Gamal Abdel Hameed <marawan.abdel.hameed@umontreal.ca>, Guillaume Rabusseau <grabus@iro.umontreal.ca>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

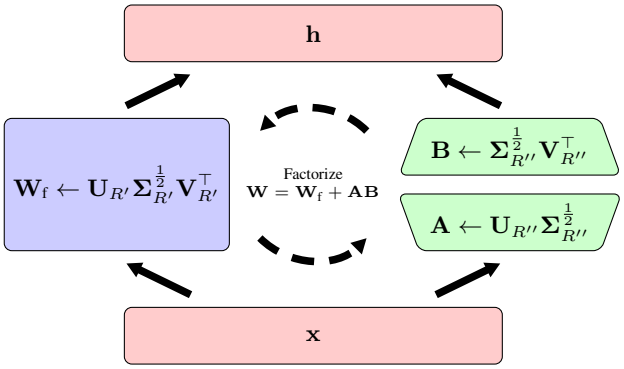


Figure 1: Illustration of ROSA. Parameter matrix $\mathbf{W}$ is factorized using SVD and split into smaller trainable matrices $(\mathbf{A}, \mathbf{B})$ and a larger fixed matrix $(\mathbf{W}_{\mathrm{f}})$. The split is then merged after a specified number of training iterations, and the process is repeated. ROSA updates an increasingly larger subspace of $\mathbf{W}$ over the course of training while remaining memory efficient

To alleviate the burdensome memory requirements of adapting PLMs to downstream tasks, various memory efficient methods have been proposed (Houlsby et al., 2019; Lin et al., 2020; Guo et al., 2021; Hu et al., 2021; Li & Liang, 2021; Lester et al., 2021; Sung et al., 2021; Liu et al., 2022). The commonality among these methods is the maintenance of fixed PLM weights while introducing a minimal quantity of trainable parameters. Although solutions like LoRA (Hu et al., 2021) and Prompt tuning (Li & Liang, 2021) are effective and do not impose any additional inference latency, they limit the expressivity of the adapted models. For instance, LoRA introduces low-rank matrices that are trainable in parallel to fixed pretrained weight matrices. Thus the weights of the adapted model are limited to subspaces whose sizes are constrained by device memory.

In this work, we propose ROSA, which expands the expressivity of adapted models, while adhering to device memory constraints. Following a similar principle as in (Hu et al., 2021), ROSA satisfies memory restrictions by selectively fine-tuning low-rank matrices in parallel to fixed pretrained weight matrices. At the same time, to increase the expressivity of the adapted model, different subspaces are *resampled* throughout training. This process is depicted in Figure 1.

Under equivalent memory restrictions, ROSA surpasses preceding techniques like LoRA, as it effectively modifies larger subspaces. Just like LoRA, ROSA carries a significant advantage by introducing no additional latency overhead during inference. In addition, ROSA provides a significant training speedup compared with a full finetuning baseline.

2. Related Work

Parameter Efficient Finetuning (PEFT) defines a class of methods to alleviate memory and compute requirements during adaptation of large models to downstream tasks. Adapter layers such as in (Houlsby et al., 2019) introduce layers in each transformer block, which necessarily leads to a latency overhead.

Prompt tuning is an efficient means of adapting models via continuous optimization of prefixes added to input prompts (Li & Liang, 2021; Lester et al., 2021; Liu et al., 2022). While such approaches are memory efficient, they require reserving a portion of the available sequence length during downstream adaptation. Moreover, prompt tuning methods can be challenging to optimize as pointed out in (Hu et al., 2021).

Other notable approaches include (Ben Zaken et al., 2022) which freezes all parameters except bias terms, and FISH (Sung et al., 2021) which optimizes a sparse difference vector to be summed with the original model parameters.

Our work is most similar to LoRA (Hu et al., 2021), which has been shown to outperform the aforementioned approaches and to mitigate limitations such as increased inference latency and reduced sequence length capacity.

3. Method

In this section we describe our proposed approach, ROSA: Random Orthogonal Subspace Adaptation. Our overarching goal is to finetune large models in a memory constrained setting, while remaining competitive with full finetuning.

3.1. ROSA

In LoRA (Hu et al., 2021), low rank matrices that are trainable are added in parallel to fixed pretrained weight matrices. The rank of the matrices is typically chosen such that the training procedure satisfies device memory constraints. However, constraining the updates to a *fixed* low rank subspace limits the adapted model's expressivity. Thus, we decouple the adapted model's expressivity from device memory constraints by *sampling* new subspaces throughout the training procedure.

Crucially, to avoid losing progress throughout training, newly sampled low rank matrices are initialized using parameters from the fixed weight matrices, by decomposing weight matrices using SVD (other approaches such as QR decomposition could equivalently be used). In more detail, given a model with parameter matrices $\mathbf{W}^{(k)} \in \mathbb{R}^{m^{(k)} \times n^{(k)}}$, a *subspace sampling* step consists of factorizing weight matrices using SVD,

$$\mathbf{W}^{(k)} = \mathbf{U}^{(k)}_{R'} \mathbf{\Sigma}^{(k)}_{R'} \mathbf{V}^{(k)\top}_{R'} + \mathbf{U}^{(k)}_{R''} \mathbf{\Sigma}^{(k)}_{R''} \mathbf{V}^{(k)\top}_{R''}, \quad (1)$$

where $\text{RANK}(\mathbf{W}^{(k)}) = R' + R''$. After sampling the subspace, training is performed with the gradients computed only with respect to the R'' subspace which consists of $R''(m^{(k)} + n^{(k)})$ parameters. In contrast, full fine-tuning requires optimizing $m^{(k)} n^{(k)}$ parameters. Thus, ROSA leads to a reduction of trainable parameters ρ_{train} given by

$$\rho_{\text{train}} = \frac{m^{(k)} n^{(k)}}{R''(m^{(k)} + n^{(k)})}. \quad (2)$$

We explore two different strategies for selecting the subspace used during training: selecting the subspace spanned by (i) random selection of singular vectors and (ii) selection of singular vectors corresponding to the R'' smallest singular values.

Though each subspace sampling step is expensive, $\mathcal{O}(\max(n^{(k)}, m^{(k)})^3)$, it is only performed once every epoch. In practice the sample step adds negligible time to the training procedure as shown in Table 1.

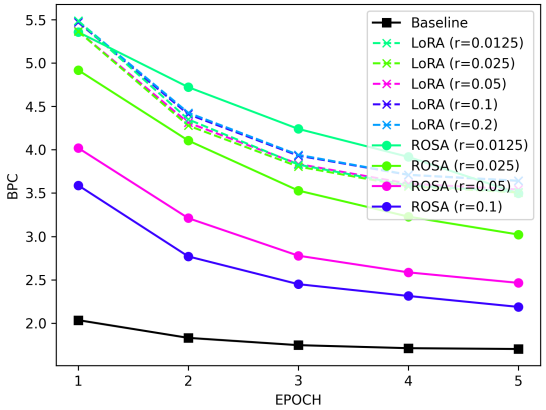
Table 1: Runtime of one training epoch of finetuning of GPT2-S (128M parameters), using ROSA and LoRA on a single GPU (Quadro RTX 8000) with an input batch of 8 sequences of length 512.

Model	**Latency (s)**
Baseline	657
LoRA	333
ROSA	332

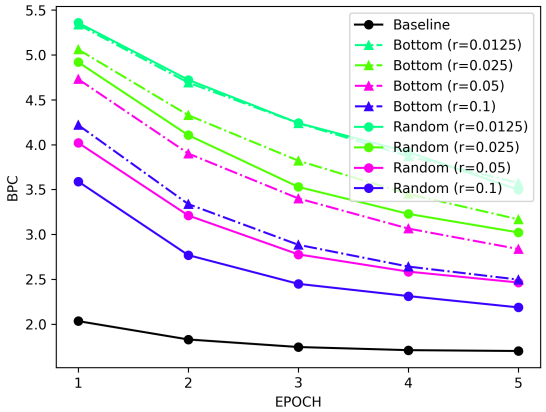
3.2. Memory efficiency

In contrast to inference, training necessitates considerably more memory usage. During inference, each parameter requires 4 bytes of storage (assuming single precision arithmetic). Meanwhile, additional memory is consumed during training for storing gradients (4 bytes per parameter), internal states of momentum-based optimizers like ADAM (8 bytes per parameter) and intermediate activations used in back-propagation.

Thus, by optimizing only a smaller subspace of parameters



(a) Validation set bits per character (BPC) training curves of ROSA and LoRA. Different ranks for trainable parameter matrices are used, and are conveyed as ratios of the rank used compared with the baseline (full rank during full finetuning)



(b) Validation set bits per character (BPC) training curves of ROSA, using different sampling strategies.

Figure 2: Finetuning of GPT2-S on the E2E dataset

we reduce the training memory burden of parameters by:

$$\frac{4\rho_{\text{train}}}{1+\rho_{\text{train}}} \tag{3}$$

Equation (3) approaches 4, as ρ_{train} increases, giving us an upper bound of $4\times$ memory reduction. In our analysis, typically $\rho_{\text{train}} \in [5, 50]$.

3.3. LoRA as a special case of ROSA

Low Rank Adaption (LoRA) introduces trainable parameters in the form of low-rank matrices added in parallel to original matrices, during the fine-tuning stage (Hu et al., 2021). That is, given a model with parameter matrices $\mathbf{W}^{(k)}$, LoRA replaces these matrices with

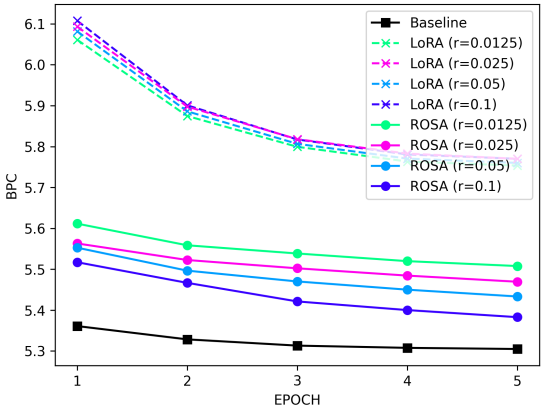


Figure 3: Validation set bits per character (BPC) training curves of ROSA and LoRA finetuning of GPT2-S on the ELI5 dataset. Different ranks for trainable parameter matrices are used, and are conveyed as ratios of the rank used compared with the baseline (full rank during full finetuning)

$$\widehat{\mathbf{W}}^{(k)} = \mathbf{W}^{(k)} + \mathbf{A}^{(k)}\mathbf{B}^{(k)}. \tag{4}$$

where $\mathbf{A}^{(k)} \in \mathbb{R}^{m^{(k)}\times R}$ and $\mathbf{A}^{(k)} \in \mathbb{R}^{R\times n^{(k)}}$. Our formulation in (1) bears close resemblance to (4). ROSA can be seen as a generalization of LoRA with sampling only conducted once. Different from LoRA, ROSA splits weight matrices into two orthogonal subspaces, one of which is used to initialize the weights of the trainable matrices. In contrast, LoRA updates a subspace that is initialized randomly and spanned by columns of the fixed parameter matrix.

3.4. Limitations of ROSA

While ROSA achieves better performance than previous state-of-the-art adaptation methods such as LoRA, it bears one main limitation compared with other methods. Namely, it requires storage of the whole model after it is adapted for a downstream task.

Other adapter methods try to simultaneously address two challenges (1) reducing memory usage during training, to ease the hardware barrier when adapting large models to a *single* downstream task and (2) reducing disk space usage when adapting a base model to *many* downstream tasks.

In contrast, ROSA focuses only on the former problem. Therefore, ROSA is better suited for use cases targetting a single downstream task, while other PEFT methods might be more suitable when targeting many downstream tasks.

Table 2: GPT2 (128M parameters) finetuned using LoRA and ROSA on the E2E NLG dataset. ROSA outperforms LoRA in perplexity at various levels of compressed training

Model	# Trainable Parameters	PPL ↓
GPT2-S (FT)	124M	1.70
GPT2-S (LoRA)	1.3M	3.51
GPT2-S (LoRA)	2.8M	3.51
GPT2-S (LoRA)	5.6M	3.54
GPT2-S (LoRA)	11.2M	3.64
GPT2-S (ROSA)	1.4M	3.50
GPT2-S (ROSA)	**2.9M**	**3.02**
GPT2-S (ROSA)	**5.7M**	**2.46**
GPT2-S (ROSA)	**11.3M**	**2.19**
GPT2-M (FT)	354M	1.52
GPT2-M (LoRA)	**4.7M**	**2.63**
GPT2-M (LoRA)	9.8M	2.68
GPT2-M (LoRA)	20.1M	2.68
GPT2-M (LoRA)	40.1M	2.87
GPT2-M (ROSA)	4.9M	2.91
GPT2-M (ROSA)	**10.1M**	**2.51**
GPT2-M (ROSA)	**20.3M**	**2.18**
GPT2-M (ROSA)	**40.3M**	**1.93**
GPT2-L (FT)	774M	1.34
GPT2-L (LoRA)	**11.8M**	**1.83**
GPT2-L (LoRA)	**23.6M**	**1.85**
GPT2-L (LoRA)	47.2M	1.86
GPT2-L (LoRA)	94.4M	1.86
GPT2-L (ROSA)	12.2M	2.28
GPT2-L (ROSA)	24.0M	2.01
GPT2-L (ROSA)	**47.6M**	**1.79**
GPT2-L (ROSA)	**94.8M**	**1.60**

Table 3: GPT2 (128M parameters) finetuned using LoRA and ROSA on the ELI5 NLG dataset. ROSA outperforms LoRA in perplexity at various levels of compressed training

Model	# Trainable Parameters	PPL ↓
GPT2-S (FT)	124.4M	5.26
GPT2-S (LoRA)	1.3M	5.50
GPT2-S (LoRA)	2.8M	5.59
GPT2-S (LoRA)	5.6M	5.50
GPT2-S (LoRA)	11.2M	5.73
GPT2-S (ROSA)	**1.4M**	**5.42**
GPT2-S (ROSA)	**2.9M**	**5.38**
GPT2-S (ROSA)	**5.7M**	**5.35**
GPT2-S (ROSA)	**11.3M**	**5.31**

4. Experiments

We evaluate the finetuning performance of ROSA on GPT2 variants (124 - 774 M parameters), for two Natural Language Generation (NLG) tasks. Namely, generating text using the E2E NLG (Novikova et al., 2017) and ELI5 (Fan et al., 2019) datasets. In our experiments, we sample new ROSA subspaces once per epoch. Our implementation uses the huggingface transformers library (Wolf et al., 2020). We finetune all models for 5 epochs using stochastic gradient descent with an initial learning rate of 2e-4, a linear learning rate schedule, 500 warmup steps and a batch size of 8.

4.1. Language Modelling with the E2E Dataset

We conducted a comparative analysis between ROSA and LoRA in terms of their finetuning performance, as shown in Table 2. Notably, ROSA consistently outperforms LoRA with a significant margin in terms of perplexity. For example, ROSA achieves a $+1.4$ improvement in perplexity over LoRA for the finetuning of GPT2-S with 11M trainable parameters. Additionally, the training curves depicted in Figure 2a demonstrate that increasing the rank of the trainable matrix benefits ROSA, whereas LoRA does not exhibit the same advantage. This observation aligns with the findings reported by the authors of LoRA (Hu et al., 2021).

4.2. Language modelling with the ELI5 NLG dataset

We further compare the finetuning performance between ROSA and LoRA on the ELI5 dataset. Looking at Table 3, we observe once more that (i) ROSA consistently outperforms LoRA and (ii) ROSA benefits from increasing rank, whereas the performance of LoRA remains relatively constant.

4.3. Empirical evaluation of subspace selection strategies

We now compare the two approaches proposed in Section 3.1 for subspace selection. Namely, we compare selection of spanning singular vectors at random compared with selecting singular vectors with lowest corresponding singular values. Figure 2b illustrates that random selection consistently outperforms the explicit selection scheme. For this reason, all other experiments use a random sampling strategy.

5. Conclusion & Future Work

In this work we explored ROSA, a memory efficient finetuning method that does not limit expressiveness of adapted models. We demonstrated that ROSA consistently outperforms LoRA and also attains better performance with increasing rank (a property that is absent in LoRA). In future work we aim to explore properties of the effective subspaces updated by ROSA. Additionally, we plan to perform more comprehensive experiments by using larger models and assessing ROSA on Natural Language Understanding (NLU) benchmarks.

References

Ben Zaken, E., Goldberg, Y., and Ravfogel, S. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 2022.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33. Curran Associates, Inc., 2020.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

Fan, A., Jernite, Y., Perez, E., Grangier, D., Weston, J., and Auli, M. ELI5: long form question answering. In Korhonen, A., Traum, D. R., and Màrquez, L. (eds.), *Proceedings of the 57th Conference of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics, 2019.

Guo, D., Rush, A., and Kim, Y. Parameter-efficient transfer learning with diff pruning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, 2021.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019.

Hu, E., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models, 2021.

Lester, B., Al-Rfou, R., and Constant, N. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.

Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, 2021.

Lin, Z., Madotto, A., and Fung, P. Exploring versatile generative language model via parameter-efficient transfer learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020.

Liu, X., Ji, K., Fu, Y., Tam, W., Du, Z., Yang, Z., and Tang, J. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 2022.

Novikova, J., Dušek, O., and Rieser, V. The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 2017.

Peng, B., Li, C., He, P., Galley, M., and Gao, J. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*, 2023.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

Sung, Y.-L., Nair, V., and Raffel, C. A. Training neural networks with fixed sparse masks. In *Advances in Neural Information Processing Systems*, 2021.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, 2020.