Slaying the HyDRA: Parameter-Efficient Hyper Networks with Low-Displacement Rank Adaptation

Xiangyu Chen, Ye Wang, Matthew Brand, Pu (Perry) Wang, Jing Liu, Toshiaki Koike-Akino
Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA 02139, USA
{xiachen, yewang, brand, pwang, jiliu, koike}@merl.com

Abstract

Low-rank adaptation (LoRA) is a widely used parameter-efficient fine-tuning (PEFT) method to adapt large foundation models on downstream tasks. However, the weight updates are constrained to be low-rank structures, limiting their expressiveness. Alternatively, low-displacement rank (LDR)-based structured matrices are rank unrestricted, while requiring few parameters and supporting fast matrix-vector multiplication. We propose a new PEFT strategy to construct the weight updates with block-wise LDR matrices by sampling parameters from a hyper network framework. Our method, hyper low-displacement rank adaptation (HyDRA), offers high flexibility for choosing the size of a pool of trainable parameters, while not being restricted by the displacement rank. Our experiments demonstrate that the HyDRA can boost the classification accuracy by up to 3.4% and achieve two-fold improvement in parameter efficiency on an image classification benchmark compared with other PEFT variants.

1 Introduction

Through the combination of immense computational power and data, large foundation models have achieved excellent performance in a wide range of general tasks, such as computer vision [10, 16, 17] and natural language processing [11, 1, 2, 3]. Leveraging the capabilities of large foundation models has become a popular approach for downstream tasks [27, 38, 41], where data, computation, and/or memory resources may be limited. Among various methods, parameter-efficient fine-tuning (PEFT) [30], where only a small number of parameters are adapted, is attractive for yielding good performance, while not requiring excessive computational cost. An outstanding approach is low-rank adaptation (LoRA) [19] and its variants [40, 5, 43, 23, 4], which restrict the weight updates to be low-rank matrices.

The work of [34] notes low-rank adaptation has limited expressiveness, and proposes structured unrestricted-rank matrices (SURM), i.e., low-displacement rank (LDR) matrices [22], to replace LoRA. LDR matrices have two major benefits: i) the number of parameters is much smaller than the matrix size; and ii) capability of efficient matrix-vector multiplications based on fast-Fourier transform (FFT). Specifically, SURM tested the expressiveness of two classic LDR matrices, Circulant and Toeplitz, and introduced more complex matrices with simple combinations. [35] proposed to construct more complex matrices with higher ranks, whose parameters also increased linearly.

We further extend the existing PEFT by integrating hyper-networks [14] with block-wise LDR matrices to enable highly flexible PEFT, by taking parameters from a parameter pool. In this case, the number of trainable parameters is flexibly determined by the parameter pool size, while different sub-matrices can be constructed by a hyper network that samples parameters from it. Moreover, like the reservoir computing framework [13], the parameter pool can contain both trainable and fixed random parameters generated on the fly, to further improve the expressiveness in some cases.

Our contributions are summarized as follows:

NeurIPS 2024 Workshop on Adaptive Foundation Models (AFM).

Table 1: Variants of LDR matrices and displacement operator matrices

Structure Matrix $\mathbf{W}$	$\mathbf{A}$	$\mathbf{B}$	Memory	Complexity
Low-Rank	$\mathbf{I}$	$\mathbf{0}$	$(n+m)r$	$\mathcal{O}[(n+m)r]$
Circulant	$\mathbf{Z}_1$	$\mathbf{I}$	$(n+m)r$	$\mathcal{O}[r(m\log(m)+n\log(n))]$
Toeplitz-like	$\mathbf{Z}_1$	$\mathbf{Z}_{-1}$	$(n+m)r$	$\mathcal{O}[r(m\log(m)+n\log(n))]$
Hankel-like	$\mathbf{Z}_1$	$\mathbf{Z}_0^\intercal$	$(n+m)r$	$\mathcal{O}[r(m\log(m)+n\log(n))]$
Vandermonde-like	$\mathrm{diag}(\mathbf{v})$	$\mathbf{Z}_0$	$(n+m)r+m$	$\mathcal{O}[r(m\log^2(m)+n\log^2(n))]$
Cauchy-like	$\mathrm{diag}(\mathbf{v})$	$\mathrm{diag}(\mathbf{u})$	$(n+m)(r+1)$	$\mathcal{O}[r(m\log^2(m)+n\log^2(n))]$

- A new PEFT method called HyDRA, an integration of hyper networks and LDR adaptation, is introduced to construct large weight updates with small block-wise LDR matrices by sampling parameters from a trainable parameter pool.
- We introduce a new mechanism which adjusts the size of parameter pool, providing more flexibility to balance between model size and expressiveness, and consider both fully and partially trainable parameter pool.
- We demonstrate that our HyDRA framework offers a significant improvement in parameter efficiency and flexibility in some benchmark experiments.

2 Related Work

PEFT: PEFT is widely used for fine-tuning large foundation models on downstream tasks with restricted compute costs [25, 42]. One outstanding class of PEFT is adapter-based methods [18, 33, 21, 31, 15, 20, 29], which freeze all of the pretrained weights and insert adapters for task-specific learning. Among these, LoRA [19] stands out by restricting the searching space of weight updates to be low-rank matrices. Various variants of LoRA have emerged, including LoHA [40], LoKr [40], KronA [12], SuperLoRA [6], and LoDA [28]. SURM [35] proposed to use structured unrestricted-rank matrices like Toeplitz matrices as alternative adapters, which introduced low-displacement rank (LDR) matrices for PEFT.

Structured Matrix: LDR matrices [32, 8, 7], constructed with a few number of parameters while benefiting from superfast matrix-vector multiplication, have been explored to construct compressed neural networks [35, 37]. For PEFT, SURM [34] tried to increase the expressiveness by simply combining structured matrices, e.g., Hadamard product of two Circulant matrices. However, this combination introduced more parameters and complexity for weight updates. Different from SURM [34], we propose the concept of parameter pool and block partition, where we draw the parameters for block-wise structured matrices. Hence, the number of parameters are controllable in more flexible manner and decoupled from the complexity of the final matrix for weight update itself. Further, SURM only investigated Circulant and Toeplitz matrices, while we consider various LDR matrices.

Hyper-Network: HyperLoRA [39] uses a hyper-network framework [14] to generate LoRA weights. HyDRA uses a much simpler hyper-network by sampling parameters from the pool directly after shuffling with a function of random number generation. VB-LoRA [26] is another hyper-network method which uses vector banks, while HyDRA constructs vectors by randomly sampling from the parameter pool. Besides, HyDRA explored a class of LDR matrices beyond LoRA.

3 Method

3.1 Low-Displacement Rank (LDR) Matrices

Structured matrices $\mathbf{W} \in \mathbb{R}^{m\times n}$ can be parameterized with less than mn parameters. For instance, low-rank matrices used for LoRA can be decomposed into $\mathbf{W} = \mathbf{GH}$, where $\mathbf{G} \in \mathbb{R}^{m\times r}$, $\mathbf{H} \in \mathbb{R}^{r\times n}$ with a rank $r \ll \{m, n\}$, realizing high parameter efficiency from mn to $(m+n)r$. We focus on a more general family of structured matrices — low-displacement rank (LDR) matrices — that can be converted into low rank matrix $\mathbf{F} \in \mathbb{R}^{m\times n}$ under the Sylvester displacement operator $\nabla_{\mathbf{A},\mathbf{B}}$:

$$\nabla_{\mathbf{A},\mathbf{B}}(\mathbf{W}) = \mathbf{AW} - \mathbf{WB} = \mathbf{F} = \mathbf{GH}, \tag{1}$$

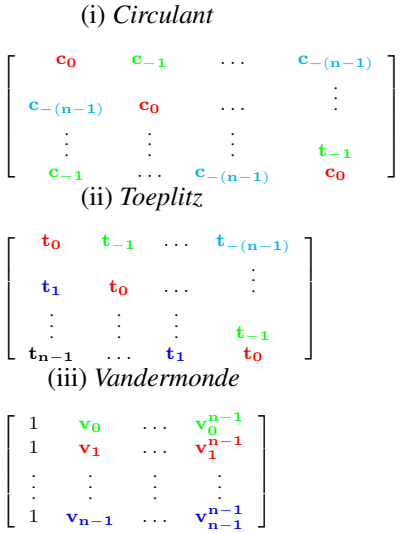

Figure 1: Illustration of different LDR matrices.

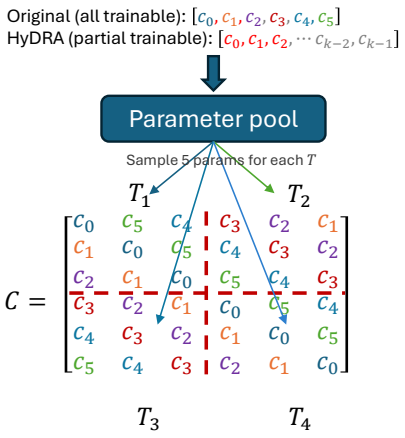

Figure 2: HyDRA. An example of constructing a matrix with 4 small Toeplitz matrices by randomly sampling parameters from the parameter pool.

where $\mathbf{A} \in \mathbb{R}^{m\times m}$ and $\mathbf{B} \in \mathbb{R}^{n\times n}$ are displacement operator matrices. Different pairs of $(\mathbf{A}, \mathbf{B})$ will result in $\mathbf{F}$ that has different properties as in Table 1. Here, $\mathbf{Z}_f$ is an f-unit-circulant matrix:

$$\mathbf{Z}_f = [\mathbf{e}_2 \quad \mathbf{e}_3 \quad \cdots \quad \mathbf{e}_n \quad f\mathbf{e}_1] = \begin{bmatrix} \mathbf{0}_{n-1}^\intercal & f \\ \mathbf{I}_{n-1} & \mathbf{0}_{n-1} \end{bmatrix}, \tag{2}$$

where $\mathbf{e}_i$ is the ith unit basis vector. LDR matrices include Circulant, Toeplitz, Hankel, Vandermonde, and Cauchy matrices as in Figure 1. Note that LoRA is a special case with $\mathbf{A} = \mathbf{I}$ and $\mathbf{B} = \mathbf{0}$. LDR matrix has two noticeable advantages:

- Parameter efficient: to construct $\mathbf{W}$ of size $m \times n$, it requires a much smaller number of parameters of the order $\mathcal{O}[r(n+m)]$.
- Computationally efficient: it enables superfast matrix-vector multiplication in sub-quadratic order of $\mathcal{O}[r(m\,\text{polylog}(m) + n\,\text{polylog}(n))]$ [32].

Table 1 shows the parameter requirement for memory and computatinal complexity.

3.2 HyDRA: Hyper-Network Low-Displacement Rank Adaptation

Block Partition: As shown in Figure 2, a Circulant matrix parameterized by its first row vector can be partitioned into multiple Toeplitz sub-matrices which are parameterized by its first row and first column. As Toeplitz matrices have more degrees of freedom, it may potentially offer more expressiveness. We introduce this block LDR partitioning to generalize a simple LDR matrix. The block-wise LDR structure still enjoys the superfast operation as discussed in Appendix A.

Hyper-Network: Can we build more complex structure like Toeplitz while keeping fewer number of parameters like Circulant? From Figure 2, a large Circulant matrix is built with small Toeplitz sub-matrices whose parameters are sampled and arranged in a unique way. Thus, to make the Circulant matrix more complex, we can simply relax the restrictions, by sampling parameters randomly from a parameter pool and arrange them to a more random LDR sub-matrices. Figure 2 shows the schematic of the parameter pool and shuffling function as a hyper-network framework used in our HyDRA.

Adjustable Parameter Pool: Noticing each LDR sub-matrices randomly sample a portion of parameters from the parameter pool, the size of parameter pool is not necessarily to be same as the degrees of freedom of LDR matrices. Instead, it can be any size so that the number of trainable parameters is flexibly adjustable and not subjected by the displacement rank and matrix size.

Partially Fixed Parameter Pool: To further decrease the number of trainable parameters, we propose that some values in the parameter pool can be random but fixed, while the rest are learnable. Note that the random values do not require additional memory when random number generators create them on the fly with a pre-specified seed.

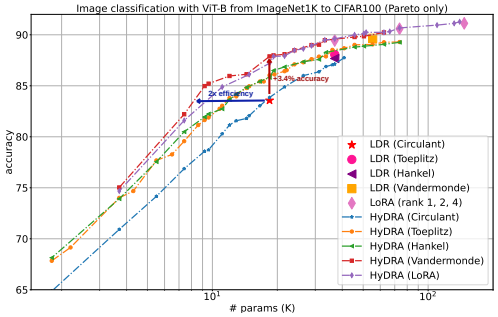

Figure 3: Comparison between HyDRA and other LDR based adapters.

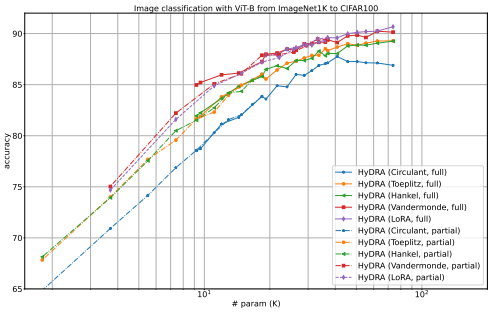

Figure 4: The effect of ratio for random fixed parameters in the parameter pool.

4 Experiments

We conduct experiments of transfer learning on image classification tasks from ImageNet1K [9] to CIFAR100 [24] for the foundation model of ViT-Base [10]. As in LoRA [19], weight updates generated by the adapters (either LDR matrices directly or our HyDRA) are added to query and value projection layers. For the classifier head, automatic label matching is used to search for the matching labels, avoiding fine-tuning classifier head and reducing the trainable parameters, similar to [6].

Comparison betweeen LDR and HyDRA: The comparison between direct LDR matrices (including LoRA) and HyDRA from different pool size can be found in Figure 3. We can see that all HyDRA variants outperform LDR (Circulant) in low-parameter regimes, where conventional LDR variants cannot reach as their flexibility is only adjustable by a rank. Specifically, HyDRA (Toeplitz) and HyDRA (Hankel) work similarly and HyDRA (Vandermonde) is much better when the amount of parameters is larger than 5k. Compared with LDR (Circulant), HyDRA (Vandermonde) improved the accuracy from 83.57% to 87.25% at the comparable amount of parameters. More importantly, HyDRA achieves roughly 2-fold parameter efficiency over conventional LDR methods.

Effect of Adjustable Parameter Pool Size: Then, we adjust the parameter pool, from 9.2k (when the parameter pool is too small that both row and column vectors for each Toeplitz matrix have same parameters) to 40k and beyond (when the pool size is large enough to make parameters for row and column vectors completely different). Results can be found from the solid lines in Figure 4. The accuracy for HyDRA increases with the increase of parameter pool. When the parameter pool is as large as when using one Toeplitz matrix as an adapter, the accuracy becomes comparable. This indicates that the parameter pool size matters, instead of how many splits it is divided into.

Effect of Using Random Values: To evaluate the effect of adding random values into the parameter pool, we select a few pool sizes, and then make the partial parameters fixed. The portion of fixed parameters varies from 90% to 0%. As shown by the dashed lines in Figure 4, accuracy decreased when the fixed portion in the parameter pool gets larger. It is also confirmed that adding random values does not degrade the performance compared to the fully trainable pool.

5 Conclusion and Future Works

In this work, we proposed HyDRA, a replacement of LoRA in the track of rank-unrestricted weight adaptation. Specifically, it constructs the $\Delta\mathbf{W}$ for weight updates by drawing parameters from a parameter pool which is partially learnable and the rest fixed for block-wise LDR sub-matrices. Experiments on the CIFAR100 dataset show that HyDRA outperforms its baseline, LDR (Circulant), by 3.4% accuracy and 2x efficiency. More experiments for diverse benchmark and foundation models are to follow.

Acknowledgment

We would like to express our sincere gratitude to Mr. Key Chron for selfless support.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[3] Rohan Anil, Andrew M. Dai, and Orhan Firat et al. PaLM 2 technical report, 2023.

[4] Klaudia Bałazy, Mohammadreza Banaei, Karl Aberer, and Jacek Tabor. LoRA-XS: Low-rank adaptation with extremely small number of parameters. *arXiv preprint arXiv:2405.17604*, 2024.

[5] Eric L Buehler and Markus J Buehler. X-loRA: Mixture of low-rank adapter experts, a flexible framework for large language models with applications in protein mechanics and molecular design. *APL Machine Learning*, 2(2), 2024.

[6] Xiangyu Chen, Jing Liu, Ye Wang, Pu Wang, Matthew Brand, Guanghui Wang, and Toshiaki Koike-Akino. SuperLoRA: Parameter-efficient unified adaptation for large vision models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 8050–8055, June 2024.

[7] Krzysztof Choromanski, Arijit Sehanobish, Somnath Basu Roy Chowdhury, Han Lin, Avinava Dubey, Tamas Sarlos, and Snigdha Chaturvedi. Fast tree-field integrators: From low displacement rank to topological transformers. *arXiv preprint arXiv:2406.15881*, 2024.

[8] Christopher De Sa, Albert Cu, Rohan Puttagunta, Christopher Ré, and Atri Rudra. A two-pronged progress in structured dense matrix vector multiplication. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1060–1079. SIAM, 2018.

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

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

[11] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[12] Ali Edalati, Marzieh Tahaei, Ivan Kobyzev, Vahid Partovi Nia, James J Clark, and Mehdi Rezagholizadeh. KronA: Parameter efficient tuning with Kronecker adapter. In *NeurIPS'23 Workshop on on Efficient Natural Language and Speech Processing*, 2023.

[13] Daniel J Gauthier, Erik Bollt, Aaron Griffith, and Wendson AS Barbosa. Next generation reservoir computing. *Nature communications*, 12(1):1–8, 2021.

[14] David Ha, Andrew M Dai, and Quoc V Le. Hypernetworks. In *International Conference on Learning Representations*, 2022.

[15] Xuehai He, Chunyuan Li, Pengchuan Zhang, Jianwei Yang, and Xin Eric Wang. Parameter-efficient model adaptation for vision transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 817–825, 2023.

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[17] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

[18] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *International conference on machine learning*, pages 2790–2799. PMLR, 2019.

[19] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[20] Shibo Jie and Zhi-Hong Deng. FacT: Factor-tuning for lightweight adaptation on vision transformer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 1060–1068, 2023.

[21] Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in Neural Information Processing Systems*, 34:1022–1035, 2021.

[22] Plamen Koev. Matrices with displacement structure–a survey. 1999.

[23] Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki M Asano. VeRA: Vector-based random matrix adaptation. In *The Twelfth International Conference on Learning Representations*, 2024.

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

[25] Jaejun Lee, Raphael Tang, and Jimmy Lin. What would elsa do? freezing layers during transformer fine-tuning. *arXiv preprint arXiv:1911.03090*, 2019.

[26] Yang Li, Shaobo Han, and Shihao Ji. VB-LoRA: Extreme parameter efficient fine-tuning with vector banks. *arXiv preprint arXiv:2405.15179*, 2024.

[27] Chenyu Lian, Hong-Yu Zhou, Yizhou Yu, and Liansheng Wang. Less could be better: Parameter-efficient fine-tuning advances medical vision foundation models. In *Medical Imaging with Deep Learning*, 2024.

[28] Jing Liu, Toshiaki Koike-Akino, Pu Wang, Matthew Brand, Ye Wang, and Kieran Parsons. LoDA: Low-dimensional adaptation of large language models. In *NeurIPS'23 Workshop on on Efficient Natural Language and Speech Processing*, 2023.

[29] Gen Luo, Minglang Huang, Yiyi Zhou, Xiaoshuai Sun, Guangnan Jiang, Zhiyu Wang, and Rongrong Ji. Towards efficient visual adaption via structural re-parameterization. *arXiv preprint arXiv:2302.08106*, 2023.

[30] Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. PEFT: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`, 2022.

[31] Yuning Mao, Lambert Mathias, Rui Hou, Amjad Almahairi, Hao Ma, Jiawei Han, Scott Yih, and Madian Khabsa. UniPELT: A unified framework for parameter-efficient language model tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6253–6264, 2022.

[32] Vadim Olshevsky and Amin Shokrollahi. Matrix-vector product for confluent Cauchy-like matrices with application to confluent rational interpolation. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 573–581, 2000.

[33] Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, 2020.

[34] Arijit Sehanobish, Avinava Dubey, Krzysztof Choromanski, Somnath Basu Roy Chowdhury, Deepali Jain, Vikas Sindhwani, and Snigdha Chaturvedi. Structured unrestricted-rank matrices for parameter efficient fine-tuning. *arXiv preprint arXiv:2406.17740*, 2024.

[35] Vikas Sindhwani, Tara Sainath, and Sanjiv Kumar. Structured transforms for small-footprint deep learning. *Advances in Neural Information Processing Systems*, 28, 2015.

[36] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, pages 369–386. SPIE, 2019.

[37] Anna Thomas, Albert Gu, Tri Dao, Atri Rudra, and Christopher Ré. Learning compressed transforms with low displacement rank. *Advances in neural information processing systems*, 31, 2018.

[38] Jifeng Wang, Kaouther Messaoud, Yuejiang Liu, Juergen Gall, and Alexandre Alahi. Forecast-PEFT: Parameter-efficient fine-tuning for pre-trained motion forecasting models. *arXiv preprint arXiv:2407.19564*, 2024.

[39] Zedian Xiao, William Held, Yanchen Liu, and Diyi Yang. Task-agnostic low-rank adapters for unseen English dialects. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7857–7870, 2023.

[40] Shih-Ying Yeh, Yu-Guan Hsieh, Zhidong Gao, Bernard BW Yang, Giyeong Oh, and Yanmin Gong. Navigating text-to-image customization: From LyCORIS fine-tuning to model evaluation. In *The Twelfth International Conference on Learning Representations*, 2024.

[41] M Zahweh, H Nasrallah, M Shukor, G Faour, and A Ghandour. Empirical study of PEFT techniques for winter wheat segmentation. *Environ. Sci. Proc*, 1(0), 2023.

[42] Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 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 (Volume 2: Short Papers)*, pages 1–9, 2022.

[43] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*, 2023.

A Block-Partition with LDR Sub-Matrices

HyDRA uses block-wise LDR matrix. Supposed that $\Delta\mathbf{W} \in \mathbb{R}^{m\times n}$ is partitioned into k^2 LDR sub-matrices (i.e., k-split per axis), a feature vector $\mathbf{x}$ should be split into k chunks to compute $\Delta\mathbf{W}\mathbf{x}$. For example, when the number of splits along each dimension is $k = 2$, then both $\Delta\mathbf{W}$ and $\mathbf{x}$ are divided as follows:

$$\Delta\mathbf{W} = \begin{bmatrix} \Delta\mathbf{W}_0 & \Delta\mathbf{W}_1 \\ \Delta\mathbf{W}_2 & \Delta\mathbf{W}_3 \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \end{bmatrix}, \tag{3}$$

where $\Delta\mathbf{W}_i \in \mathbb{R}^{\frac{m}{k}\times\frac{n}{k}}$ is the ith LDR matrix for $i \in \{0, 1, 2, 3\}$, and $\mathbf{x}_i \in \mathbb{R}^{\frac{n}{k}\times 1}$. We then have

$$\Delta\mathbf{W}\mathbf{x} = \begin{bmatrix} \Delta\mathbf{W}_0 & \Delta\mathbf{W}_1 \\ \Delta\mathbf{W}_2 & \Delta\mathbf{W}_3 \end{bmatrix} \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \end{bmatrix} = \begin{bmatrix} \Delta\mathbf{W}_0\mathbf{x}_0 + \Delta\mathbf{W}_1\mathbf{x}_1 \\ \Delta\mathbf{W}_2\mathbf{x}_0 + \Delta\mathbf{W}_3\mathbf{x}_1 \end{bmatrix} \tag{4}$$

For arbitrary k, the above equations should be modified accordingly. Then each $\Delta\mathbf{W}_i\mathbf{x}_j$ can be computed with a superfast matrix-vector multiplication algorithm [32]. Correspondingly, the overall computational complexity is still superfast even for the partitioned LDR structure.

For instance, when $\Delta\mathbf{W}_i$ is a Toeplitz-like matrix having $\mathbf{G}_i = [\mathbf{g}_{i,1}, \ldots, \mathbf{g}_{i,r}]$ and $\mathbf{H}_i^\intercal = [\mathbf{h}_{i,1}, \ldots, \mathbf{h}_{i,r}]$, we can express as follows:

$$\Delta\mathbf{W}_i\mathbf{x}_j = \sum_{l=1}^{r} \mathsf{Krylov}(\mathbf{Z}_1, \mathbf{g}_{i,l}) \cdot \mathsf{Krylov}(\mathbf{Z}_{-1}, \mathbf{h}_{i,l})\mathbf{x}_j \tag{5}$$

where Kryolov operator is computed by FFT/IFFT as follows:

$$\mathsf{Krylov}(\mathbf{Z}_1, \mathbf{v})\mathbf{u} = \mathbf{ifft}(\mathbf{fft}(\mathbf{v}) \circ \mathbf{fft}(\mathbf{u})), \tag{6}$$

$$\mathsf{Krylov}(\mathbf{Z}_{-1}, \mathbf{v})\mathbf{u} = \bar{\eta} \circ \mathbf{ifft}(\mathbf{fft}(\bar{\eta} \circ \mathbf{v}) \circ \mathbf{fft}(\bar{\eta} \circ \mathbf{u})), \tag{7}$$

where $\bar{\eta} = [1, \eta, \eta^2, \ldots, \eta^{n-1}]^\intercal$, and $\eta = \exp(i\frac{\pi}{n})$ which can be further simplified with diagonalization as in [35]. Refer to [37] for fast matrix-vector multiplication of other LDR matrices and corresponding implementation[1].

B Experimental Settings

In the transfer learning experiments from ImageNet1K to CIFAR100, we use AdamW optimizer with OneCycle [36] learning rate scheduler. Experiment is run for $5{,}000$ iterations in total. The maximum learning rate is 0.004 for LoRA, 0.002 for Circulant, Toeplitz and Hankel, and 0.01 for Vandermonde. Note that dimension for query/value projection layers is 768 for ViT-B model. The pretrained ViT-B model we used is from HuggingFace[2]. Other hyper-parameters settings are as below:

- Number of splits k: 1 for HyDRA (LoRA), 4 for all other HyDRA variants tested.
- Size of fully trainable parameter pool (denoted as the pool size for each $\Delta\mathbf{W}$:
  - HyDRA (Circulant): 384, 400, 500, 600, 768, 800, 900, 1000, ... 1500, 1536 (768×2), 1700, 1900, 2100, 768×3, 2600
  - HyDRA (Vandermonde): HyDRA (Circulant) and 768×4
  - HyDRA (Toeplitz): HyDRA (Circulant) and 768×p where $p = 4, 5, 6, 7, 8$
- Parameter pool size with fixed ratio (0, 0.1, 0.2, ... 0.9):
  - HyDRA (Circulant/Vandermonde/Hankel): 768
  - HyDRA (Toeplitz): 768, 900, 1536
  - HyDRA (LoRA): 1536 (rank 1), 3072 (rank 2), 6144 (rank 4)

[1] `https://github.com/HazyResearch/structured-nets`

[2] `https://huggingface.co/google/vit-base-patch16-224`