Resource-Aware Parameter-Efficient Model Adaptation for Onboard High-Dimensional Data

Qiyang Zhang[1] **Xinhao Li**[2] **Lei Shi**[3] **Zheng Lin**[4]
Jinfeng Wen[1] **Ao Zhou**[1] **Shangguang Wang**[1]

[1]Beijing University of Posts and Telecommunications [2]Wuhan University
[3]Communication University of China [4]University of Luxembourg

Email:{qyzhang,jinfeng.wen,aozhou,sgwang}@bupt.edu.cn
xhl17866703382@163.com, leiky_shi@cuc.edu.cn, zhenglin@ieee.org

Abstract

Onboard satellite models often require frequent updates, but the weights adapted to earlier data distributions can quickly become outdated. However, updating large-scale model parameters in orbit presents significant challenges due to the limited uplink bandwidth of Low Earth Orbit (LEO) satellite systems, particularly for hyperspectral satellite imagery, where high-dimensional spectral–spatial inputs lead to increased model size and update costs. Existing full fine-tuning methods are thus expensive to retrain and difficult to deploy under strict communication constraints. To address this challenge, we propose NE-LoRA, a parameter-efficient adaptation framework for bandwidth-constrained onboard hyperspectral model updates. NE-LoRA combines a primary low-rank branch with a nonlinear auxiliary branch to capture both global update trends and complex spectral–spatial variations. Additionally, we introduce a differentiated training strategy for multi-matrix adapters, motivated by the asymmetric initialization and gradient dynamics of different adapter matrices. Experiments on four hyperspectral datasets and three representative backbone models demonstrate that NE-LoRA outperforms LoRA-based baselines in most evaluated settings and remains competitive with, and in several cases superior to, full fine-tuning. Across the evaluated settings, NE-LoRA updates only a small fraction of the total parameters on average while preserving low deployment overhead, offering a favorable accuracy–communication trade-off for onboard hyperspectral adaptation.

1 Introduction

Low Earth Orbit (LEO) satellites play a crucial role as key edge nodes in emerging LEO satellite networks (LSNs), offering vital sensing and communication capabilities for modern networked applications. Use cases such as machine learning (ML)-driven Earth monitoring and emergency response [5, 20, 21, 34] are expected to become increasingly important in the future satellite Internet.

However, a significant challenge to practical onboard intelligence is the absence of efficient model updating mechanisms in conventional satellite architectures. Without timely updates, onboard models struggle to adapt to evolving data distributions or shifting application demands [30, 35, 37], leading to a gradual degradation or even failure of application functionalities [14]. When updates are required, the common solution is to retrain and redistribute model weights from scratch, which is time-consuming and environmentally unsustainable. For example, [9] reports that uploading updated neural network parameters can take anywhere from several minutes to several hours, severely limiting the practicality of frequent onboard fine-tuning. This challenge is expected to worsen as model size and update frequency continue to increase [36]. In hyperspectral satellite systems, the difficulty is

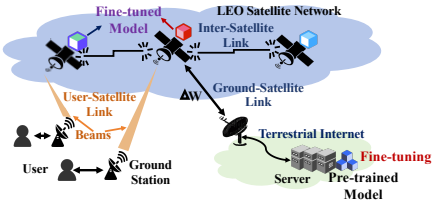
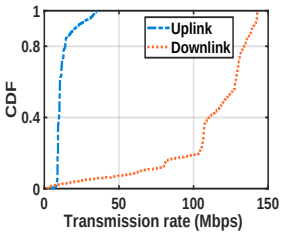


Figure 1: A high-level architecture of today's LSNs.



(a) Experimental setup of data collection.



(b) CDF of uplink/downlink transmission rates.

Figure 2: Empirical measurements on an operational satellite.

further exacerbated by the high resolution and complex spectral–spatial information, which result in high-dimensional inputs, larger models, and increased update costs. Consequently, efficient, communication-aware model updating is crucial for maintaining accuracy and uplink efficiency.

Updating hyperspectral satellite models introduces three major challenges: (i) *Diverse models and limited onboard resources.* Onboard satellite models require periodic updates to sustain performance and generalization. However, limited computation, memory, and energy make frequent or model-specific adaptation difficult, with direct onboard training often being infeasible. (ii) *Constrained bandwidth and intermittent connectivity.* Updated parameters must be transmitted over narrow bandwidth within short satellite–ground contact windows, rendering full-model transmission impractical. (iii) *High-dimensional and information-rich hyperspectral feature.* The large number of spectral bands and complex spectral–spatial structure of hyperspectral imagery significantly increase both model size and adaptation cost. Existing methods do not fully address these three challenges simultaneously. A clear gap remains in developing model update approaches that satisfy application requirements, communication constraints, and hyperspectral model characteristics. This gap motivates the design of a parameter-efficient and communication-aware adaptation framework for hyperspectral satellite model updating.

Research into onboard model updates is still in its early stage. Most existing LoRA variants rely on linear low-rank update structures, which may be insufficient for capturing the complex nonlinear spectral–spatial interactions required for hyperspectral analysis [3, 8, 16, 31, 38]. As a result, hyperspectral satellite model updating requires a framework that balances onboard accuracy, parameter efficiency, and communication cost.

To address these limitations, we propose NE-LoRA, a nonlinear-enhanced and differentiated adaptation framework for hyperspectral satellite model updating. NE-LoRA augments standard low-rank adaptation with a Gaussian Error Linear Unit (GELU)-activated auxiliary branch to improve nonlinear spectral–spatial modeling while preserving the low-rank structure. Unlike existing nonlinear LoRA variants developed for general vision tasks, our framework is specifically designed for bandwidth-constrained onboard hyperspectral adaptation, where upload cost is a critical factor, and spectral–spatial correlations differ substantially from conventional RGB imagery. In addition, while standard LoRA applies a uniform learning rate to all adapter matrices, the asymmetric initialization and gradient dynamics of different matrices can lead to inefficient or unstable optimization. This issue is particularly problematic in hyperspectral models, where large-proportion modules account for a significant portion of the overall parameter update and often exhibit highly asymmetric gradient behaviors. Without explicit control, uniform learning rates may either over-amplify noisy updates or suppress important adaptation signals. To overcome this issue, we introduce a differentiated training strategy for multi-matrix structures. Specifically, we extend the standard LoRA update by incorporating: (i) a scaling factor to amplify the output-side modules with weak gradient signals, and (ii) a reduction factor to stabilize large proportion modules by suppressing excessive updates.

As illustrated in Fig. 1, the update process begins with the satellite downlinking a small set of newly acquired samples to the ground station. The ground station then trains the dual-branch adapters using its relatively abundant computational resources. Before uplink transmission, the learned update is folded into an equivalent low-rank form so that the deployed upload cost remains of the same order as that of standard LoRA. Upon reception, the satellite integrates the transmitted update into the original

model without additional adaptation overhead, thus balancing ground-side flexibility and onboard efficiency.

Our main contributions are summarized as follows:

- We study parameter-efficient model adaptation for bandwidth-constrained onboard hyperspectral applications, a setting where spectral–spatial complexity and limited uplink capacity jointly make onboard model updates difficult.
- We propose NE-LoRA, a dual-branch adaptation framework that augments standard low-rank updates with a nonlinear auxiliary branch, enabling more expressive modeling of hyperspectral spectral–spatial variations while retaining lightweight adaptation.
- We introduce a differentiated training strategy for multi-matrix adapters, which mitigates the issue that large-proportion modules account for a significant portion of the total parameter update, and a uniform learning rate either over-amplifies noisy updates or suppresses important adaptation signals, thereby improving optimization stability.
- Extensive evaluation on four hyperspectral datasets and three backbone architectures shows that NE-LoRA outperforms LoRA-based baselines in most evaluated settings and provides a favorable accuracy–communication trade-off under deployment constraints.

2 Background & Related Work

2.1 Background

To characterize the communication constraints of onboard model updating, we consider a real-world LEO satellite computing platform and measure its downlink and uplink rates. Fig. 2(a) illustrates the experimental setup used for data collection. We employ a widely used network measurement tool (i.e., Iperf [1]) to record downlink and uplink throughput, and report the corresponding cumulative distribution functions (CDFs) in Fig. 2(b). The measurements show a clear communication asymmetry: the average downlink rate is close to 100 Mbps, whereas the average uplink rate is only about 12 Mbps. For onboard model updating, this limited uplink capacity becomes a critical bottleneck for parameter delivery from the ground station to satellites. For example, transmitting a VGG-16 model (528 MB) from the ground requires approximately 5.9 minutes, which is too long to distribute updated models to all satellites within a single satellite–ground contact window. These results highlight the stringent communication constraints faced by model updates.

Beyond uplink bottlenecks, onboard models are also constrained by limited computational resources, which restrict the complexity of feasible update algorithms. Frequent in-orbit updates, especially for large or high-dimensional models, can quickly exceed available compute and memory budget. Taken together, limited uplink bandwidth and constrained onboard computation make efficient model updating a critical challenge for satellite intelligence.

Satellite constellations, consisting of numerous broadband satellites equipped with user-satellite links (USLs) [2], extend the boundary of today's Internet and support emerging networked services at a global scale. Many of these satellites are equipped with hyperspectral sensors, which capture reflected signals across a large number of contiguous spectral bands and therefore produce high-dimensional spectral–spatial data. Unlike conventional RGB imagery with only three channels, hyperspectral imagery typically contains tens to hundreds of bands, substantially increasing the number of model input channels and the associated adaptation cost. High spectral dimensionality is a defining characteristic of hyperspectral data. For instance, on the WHU-LK dataset, there are 270 bands. This characteristic further amplifies the difficulty of efficient onboard model updating.

2.2 Related Work

Fine-tuning aims to adapt pretrained models to new tasks while controlling computational and training costs. Existing methods related to model updating can be broadly grouped into three categories.

The first category includes approaches for mitigating catastrophic forgetting, such as learning without forgetting [24] and few-shot distillation [25]. These methods aim to preserve previously acquired knowledge during model updates. For hyperspectral image classification, however, applying such methods often requires retaining feature representations, outputs, or representative samples from

previous tasks. Due to the high dimensionality of hyperspectral inputs, storing and repeatedly accessing such information can quickly exceed the storage and communication budget available in satellite systems. As a result, although these methods are effective for continual adaptation, they are not designed for onboard updating and often rely on recurrent access to historical data.

The second category includes lightweight compression approaches, such as distillation and pruning, which aim to reduce model size or training cost. For example, Meng et al. [29] proposed a distillation framework to accelerate sampling in guided diffusion models. Similarly, Cho et al. [6] introduced parameter pruning to compress neural models by removing redundant parameters. Although such methods can reduce model size, aggressive compression may weaken spectral dependencies that are important for hyperspectral classification. Therefore, these approaches do not directly address the joint challenge of communication efficiency, limited supervision, and spectral–spatial fidelity in satellite scenarios.

The third category is LoRA-based parameter-efficient fine-tuning, which has become a widely adopted paradigm for adapting pretrained models with a small number of trainable parameters. LoRA represents weight updates using low-rank matrix pairs, thereby reducing computational, memory, and transmission overhead while preserving inference efficiency. This framework has been successfully applied in natural language processing, computer vision, and multimodal learning [28]. Several variants have been proposed to further improve its effectiveness. For instance, LoRA+ uses differentiated learning rates for low-rank factors to improve optimization in wide models [11]. DoRA decomposes pretrained weights into magnitude and direction components and applies low-rank adaptation to the direction component to narrow the gap with full fine-tuning [27]. AdaLoRA adaptively allocates rank budgets across layers according to their importance, thereby improving parameter usage efficiency [40]. Despite these advances, most existing LoRA variants still rely on linear low-rank update structures. In hyperspectral applications, where inputs contain many spectral bands and exhibit complex spectral–spatial interactions, purely linear adapters may have limited capacity to capture the nonlinear patterns required for accurate adaptation.

Building on this analysis, we propose NE-LoRA, a dual-branch low-rank adaptation framework with differentiated training for multi-matrix adapters. The proposed method introduces a nonlinear auxiliary branch to improve representational capacity and a differentiated training strategy to balance optimization across matrices, thereby achieving a better trade-off among adaptation accuracy, parameter efficiency, and transmission cost in hyperspectral satellite model updating.

3 Design

3.1 Overview

Fig. 3 illustrates the overview design of NE-LoRA, which augments standard LoRA with nonlinear enhancement and differentiated training. The framework adopts a dual branch design: a main low-rank branch that captures the dominant linear update with minimal trainable parameters, and an auxiliary branch activated by GELU to model complex spectral–spatial characteristics. In addition, we introduce a differentiated training strategy for multi-matrix adapters, motivated by the asymmetric initialization and gradient dynamics of different adapter matrices in hyperspectral satellite models. Together, these designs enable communication-efficient and stable adaptation, making the framework well suited for bandwidth-constrained onboard hyperspectral applications. During deployment, the auxiliary branch is folded into an equivalent low-rank update form used for transmission, so that the deployed upload cost remains of the same order as that of standard LoRA. After ground-side folding, only the two effective low-rank factors for each adapted layer are transmitted to the satellite. The branch merging is therefore a deployment-time reparameterization rather than transmission of a dense full-size update.

3.2 Nonlinear-Enhanced Low-Rank Adaptation

Parameter-efficient adaptation is a fundamental challenge for hyperspectral satellite models operating under strict uplink constraints. Hyperspectral models jointly process spectral and spatial information [22]. However, as the number of input channels increases, the parameter count grows rapidly. Each layer must handle high-dimensional spectral features, often resulting in models with millions of

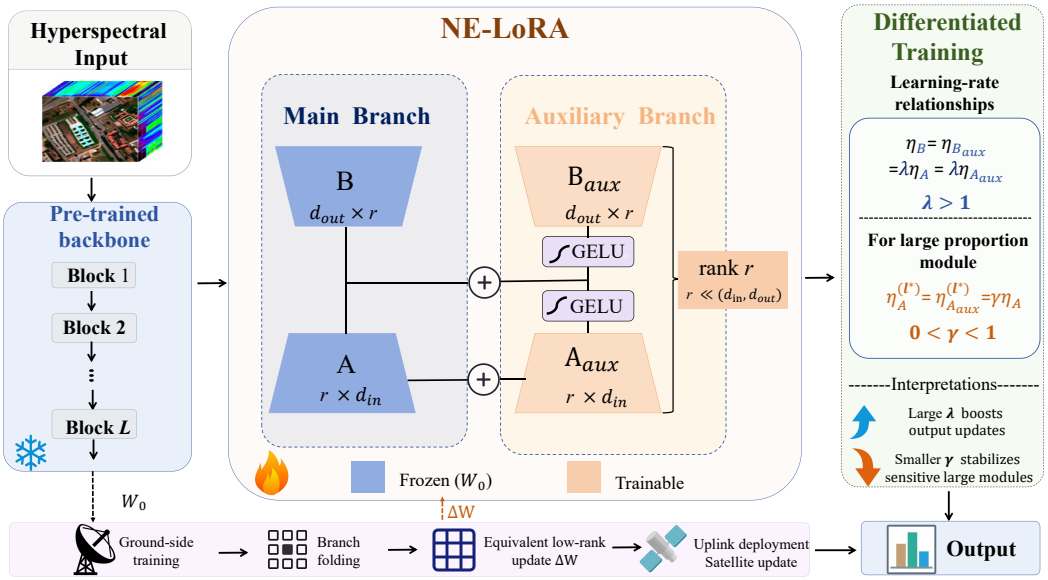


Figure 3: NE-LoRA overview.

parameters [26]. This not only increases the computational cost of fine-tuning, but also substantially raises the transmission cost of model updates, thereby limiting practical deployment in orbit.

To address the challenges, LoRA provides a natural starting point, as it reduces both computation and transmission overhead by representing weight updates with low-rank factors. Instead of updating the full parameter matrix, LoRA optimizes only a small fraction of trainable parameters, making it particularly attractive for resource-constrained hyperspectral applications. Formally, for a model with pretrained weight $W_0 \in \mathbb{R}^{d\times k}$, where d and k denote the output and input dimensions, respectively, the adapted weight is written as:

$$W = W_0 + \Delta W, \tag{1}$$

where the weight increment $\Delta W = \alpha BA$, $B \in \mathbb{R}^{d\times r}$ and $A \in \mathbb{R}^{r\times k}$ with $r \ll \min\{d, k\}$, and α is a scaling factor that controls the update magnitude. Following the standard low-rank adaptation setup, one factor is randomly initialized while the other is initialized to zero so that the initial update is zero. Despite its efficiency, the strictly linear form of LoRA limits its ability to capture nonlinear spectral–spatial patterns, which are common in hyperspectral sensing [4].

To improve representational capacity while preserving parameter efficiency, we propose NE-LoRA, a nonlinear-enhanced low-rank adaptation mechanism. The key idea is to augment the standard low-rank update with an auxiliary nonlinear branch. Specifically, the main branch (A, B) models the dominant linear update, while the auxiliary branch $(A_{\text{aux}}, B_{\text{aux}})$ captures additional nonlinear variations through GELU activation. Here, $A_{\text{aux}} \in \mathbb{R}^{r\times k}$ and $B_{\text{aux}} \in \mathbb{R}^{d\times r}$ have the same dimensionality as the corresponding matrices in the main branch. Accordingly, the weight increment is defined as:

$$\Delta W = \alpha\Big(\big(B + \text{GELU}(B_{\text{aux}})\big)\big(A + \text{GELU}(A_{\text{aux}})\big)\Big). \tag{2}$$

Only A, B, A_{aux}, and B_{aux} are trainable, while W_0 remains frozen. The main and auxiliary factors are jointly optimized through the same effective low-rank update. Their gradients are coupled through the effective factors, allowing the auxiliary parameterization to influence the optimization trajectory. This design allows NE-LoRA to jointly model global linear trends and localized nonlinear corrections, both of which are important in heterogeneous and dynamic satellite environments. GELU acts on the adapter parameters rather than on the input features, introducing a nonlinear reparameterization that modifies the optimization dynamics. This formulation enables exact folding into the standard rank-r LoRA form without increasing the deployment parameter budget.

The use of GELU is motivated by its smooth nonlinear behavior and favorable optimization properties [33]. In particular, GELU is approximately linear around small inputs while providing richer nonlinear responses for larger activations, which improves expressiveness without discarding the low-rank

structure. The GELU function is defined as:

$$\mathrm{GELU}(x) = \frac{x}{2}\left(1 + \mathrm{erf}\left(\frac{x}{\sqrt{2}}\right)\right), \tag{3}$$

where $\mathrm{erf}(\cdot)$ denotes the error function. Because the auxiliary matrices share the same dimensions as the main low-rank factors, their outputs remain naturally aligned with the main branch, which facilitates joint modeling of linear and nonlinear patterns while avoiding unnecessary parameter redundancy.

NE-LoRA also adopts a stability-aware initialization strategy. Specifically, the auxiliary matrices A_{aux} and B_{aux}, together with the main output-side matrix B, are initialized to zero, while the main input-side matrix A is initialized with Kaiming initialization [12]. At the beginning of training, since $A_{\mathrm{aux}} = B_{\mathrm{aux}} = B = \mathbf{0}$, the update is initially close to the standard LoRA form and avoids introducing large perturbations. Notably, this asymmetric initialization leads to different gradient dynamics across the adapter factors. This imbalance motivates the differentiated training strategy introduced in Section 3.3. As training proceeds, the soft activation behavior of GELU gradually enables the auxiliary branch, yielding a smooth transition from purely linear adaptation to hybrid linear–nonlinear adaptation. This progressive activation reduces optimization instability and improves training robustness.

3.3 Differentiated Training for Accuracy Compensation

Beyond representational capacity, adapting hyperspectral satellite models involves a second challenge: different adapter matrices contribute unevenly to optimization and exhibit asymmetric gradient dynamics. Here, the input-side matrices project features into a low-rank subspace, whereas the output-side matrices map the adapted representation back to the original feature space. This mismatch becomes even more pronounced in NE-LoRA, where multiple matrices are jointly optimized.

To address this issue, we introduce a differentiated training strategy tailored to NE-LoRA. The strategy is motivated by two observations. First, the adapter matrices in NE-LoRA are not optimization-wise equivalent. The input-side and output-side matrices play different roles in the low-rank adaptation path, and their asymmetric initialization further leads to different gradient scales during training. Consequently, applying a uniform learning rate to all matrices can disrupt the update balance, causing some matrices to accumulate unnecessary perturbations while others remain under-updated. Second, this problem becomes more pronounced in large-proportion modules. Since such modules contribute a substantial fraction of the overall parameter update, their optimization dynamics exert a disproportionately strong influence on the whole adaptation process. If their learning rates are not properly controlled, the resulting updates can dominate the training trajectory and degrade training stability. Accordingly, our strategy includes two components: (i) learning-rate differentiation across adapter matrices to restore update balance, and (ii) learning-rate reduction for the identified large-proportion module to improve training stability.

We first address the imbalance caused by the asymmetric initialization and gradient dynamics of the multi-matrix structure. In particular, the input-side matrices (A, A_{aux}) and output-side matrices (B, B_{aux}) play different roles during adaptation and should therefore not share the same learning rate. The matrix A is initialized with Kaiming initialization to prevent excessive feature scaling, while A_{aux}, B, and B_{aux} are initialized to zero to ensure stable adaptation from pretrained weights. As model width increases, this initialization asymmetry leads to different gradient scales: the randomly initialized input-side matrices can introduce small perturbations from the beginning of training, whereas the zero-initialized output-side matrices tend to receive weaker effective updates in the early stage. Using a uniform learning rate therefore either over-amplifies perturbations in the input-side matrices or under-updates the output-side matrices, resulting in suboptimal adaptation.

To mitigate this issue, we assign a larger learning rate to the output-side matrices. Specifically, we set the learning rates of B and B_{aux} to λ times those of A and A_{aux}, where $\lambda > 1$: $\eta_B = \eta_{B_{\mathrm{aux}}} = \lambda\eta_A = \lambda\eta_{A_{\mathrm{aux}}}$. This design improves optimization in three ways: First, a larger learning rate compensates for the weak early-stage gradients of B and B_{aux} caused by zero initialization. Second, a smaller learning rate for A and A_{aux} suppresses the accumulation of unnecessary perturbations and stabilizes training. Third, the fixed ratio λ maintains a balanced update relationship between input-side and output-side matrices as model width varies. Therefore, $\lambda > 1$ serves as a principled training modulation factor rather than an arbitrary amplification coefficient.

After addressing the imbalance across adapter matrices, we further consider instability caused by large-proportion modules. Such modules account for a substantial fraction of the parameter update and are particularly sensitive to learning-rate choices, as illustrated in the Appendix. Prior research overlooked the low-rank adaptation characteristics, despite their large parameter scales and strong influence on overall performance [39]. Without tailored control, they may introduce excessively large or insufficient updates, thereby destabilizing the overall adaptation process. To identify the large-proportion module, we denote by P_l^{adapter} the number of trainable adapter parameters in the main and auxiliary low-rank factors of the l-th adapted layer. We select the layer with the largest adapter parameter count as

$$l^\star = \arg\max_l P_l^{\text{adapter}}, \tag{4}$$

where P_l^{adapter} counts the parameters of both the main and auxiliary low-rank factors. This parameter-count-based criterion provides a simple and reproducible way to identify the module for additional learning-rate control. The selected module remains fixed throughout training.

For this module, we introduce a reduction factor γ ($0 < \gamma < 1$) to further modulate the learning rates of its input-side matrices:

$$\eta_A^{(l^\star)} = \eta_{A_{\text{aux}}}^{(l^\star)} = \gamma\eta, \qquad \eta_B^{(l^\star)} = \eta_{B_{\text{aux}}}^{(l^\star)} = \eta. \tag{5}$$

Here, η denotes the reference learning rate for the selected module, applied to its output-side factors. This design extends the logic of differentiated training by applying additional control to the selected large-proportion module. Reducing the learning rate of its input-side matrices alleviates the risk of overly aggressive updates caused by large dimensionality, while preserving sufficient adaptation capacity on the output side.

4 Implementation and Methodology

Implementation Details. Our method is implemented in Python 3.8 with PyTorch 2.2 and CUDA 12.1; all experiments are conducted on NVIDIA RTX 4090 GPUs. For hyperparameter exploration, we vary the rank r over $\{1, 3, 5, 7, 9, 11, 13, 15\}$ to examine the effect of different low-rank constraints. In particular, we also evaluate the extreme low-rank case $r = 1$ to assess whether NE-LoRA can still capture high-level hyperspectral semantics under a highly constrained adaptation budget. We further vary the learning-rate ratio λ over $\{0.0001, 0.0005, 0.001, 0.005, 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ to study its effect on differentiated optimization. In practice, we observe that $\lambda \in [2, 8]$ and $\gamma \in [0.2, 0.8]$ provide stable and effective performance across datasets.

Models. We evaluate three representative backbone models widely used in hyperspectral image analysis: CNN3D [23], which directly captures joint spectral–spatial features through 3D convolutions; M3DDCNN [13], a multi-scale 3D convolutional neural network with hierarchical feature fusion for complex spectral variations; and HybridSN [32], a hybrid network combining 3D convolutions and 2D convolutions for hyperspectral image classification. For CNN3D, we adapt all 3D convolutional layers; for M3DDCNN, we adapt the convolutional layers in the first two blocks (conv1 and conv2); and for HybridSN, we adapt all 3D convolutional layers and the first fully connected layer. Although LoRA is more commonly studied in large foundation models, our choice of lightweight backbones is motivated by the computational constraints of on-orbit platforms. Even for compact models, parameter-efficient adaptation remains important because full-parameter transmission still incurs prohibitive uplink cost in orbit.

Datasets. We evaluate NE-LoRA on hyperspectral classification, a representative remote-sensing task relevant to satellite applications, to assess adaptation accuracy and parameter transmission efficiency. We conduct experiments on four hyperspectral image datasets: (i) Salinas Valley (SA) [7]: with 204 bands and 16 classes; (ii) Pavia University (PU) [18]: with 103 bands and 9 classes; (iii) HyRANK-Loukia (HR-L) [17]: with 176 bands, spatial resolution 249×945, and 14 classes; (iv) WHU-LongKou (WHU-LK) [41]: with 270 bands and 9 classes. For data partitioning, we use 1% of samples for initial training and 1% for LoRA updates on SA and PU, 0.05% + 0.05% on WHU-LK, and 3% + 3% on HR-L. Each hyperspectral sample is supervised by the label of its center pixel, and the labeled center pixels assigned to the training, validation, and test sets are mutually disjoint. These settings are consistent with common practice in hyperspectral classification, where limited labeled samples are often sufficient for effective adaptation.

Table 1: Performance comparison of different methods on CNN3D. Values in parentheses for OA, AA, and Kappa indicate the improvement of NE-LoRA over the average of the other baselines on each dataset, while those in the Upload column indicate the reduction factor in uplink parameters relative to FT.

Dataset	Method	OA (%)	AA (%)	Kappa ($\times 100$)	Total	Train	Upload
SA	NFT	91.98	94.22	91.07	161.7k	0	0
	FT	93.22	95.67	92.45	161.7k	161.7k	161.7k
	LoRA	93.13	95.71	92.35	189.4k	27.7k	27.7k
	LoRA+	93.15	96.12	92.39	189.4k	27.7k	27.7k
	VERA	90.55	93.79	89.47	190.0k	0.7k	0.7k
	DoRA	93.03	95.46	92.24	189.6k	27.9k	27.9k
	AdaLoRA	93.20	95.70	92.44	217.1k	55.4k	50.3k
	NE-LoRA	**94.37 (+1.76)**	**96.78 (+1.54)**	**93.74 (+1.97)**	217.1k	55.4k	27.7k (5.8×)
PU	NFT	93.50	90.96	91.28	70.3k	0	0
	FT	96.90	95.44	95.90	70.3k	70.3k	70.3k
	LoRA	96.48	94.55	95.34	86.6k	16.3k	16.3k
	LoRA+	96.75	95.16	95.70	86.6k	16.3k	16.3k
	VERA	90.19	82.74	86.78	87.3k	0.7k	0.7k
	DoRA	96.33	93.87	95.13	86.8k	16.5k	16.5k
	AdaLoRA	96.37	93.94	95.18	103.0k	32.7k	29.6k
	NE-LoRA	**97.42 (+2.20)**	**96.30 (+3.92)**	**96.59 (+2.97)**	103.0k	32.7k	16.3k (4.3×)
HR-L	NFT	78.95	71.01	74.92	132.1k	0	0
	FT	81.89	77.91	78.49	132.1k	132.1k	132.1k
	LoRA	84.93	79.19	81.96	156.9k	24.8k	24.8k
	LoRA+	84.20	78.01	80.96	156.9k	24.8k	24.8k
	VERA	77.53	65.44	72.89	157.6k	0.7k	0.7k
	DoRA	81.49	74.82	77.96	157.1k	25.0k	25.0k
	AdaLoRA	81.74	74.43	78.15	181.8k	49.7k	46.5k
	NE-LoRA	**87.43 (+5.90)**	**84.23 (+9.83)**	**85.03 (+7.13)**	181.8k	49.7k	24.8k (5.3×)
WHU-LK	NFT	84.19	50.88	78.87	127.0k	0	0
	FT	87.78	55.88	83.63	127.0k	127.0k	127.0k
	LoRA	88.56	58.61	84.75	162.2k	35.2k	35.2k
	LoRA+	89.95	65.99	86.71	162.2k	35.2k	35.2k
	VERA	80.94	47.38	74.45	162.9k	0.7k	0.7k
	DoRA	90.24	67.04	87.07	162.4k	35.4k	35.4k
	AdaLoRA	**90.60**	65.29	**87.52**	197.5k	70.5k	67.5k
	NE-LoRA	90.29 (+2.82)	**67.63 (+8.91)**	87.15 (+3.86)	197.5k	70.5k	35.2k (3.6×)

Metrics. We adopt three classification metrics to evaluate model performance [10]: (i) *Overall Accuracy* (OA), which measures the proportion of correctly classified pixels; (ii) *Kappa coefficient*, which quantifies agreement beyond chance and complements OA in assessing prediction reliability; and (iii) *Average Accuracy* (AA), which averages class-wise accuracies and reflects performance across different categories. To evaluate parameter efficiency, we additionally report three parameter-related metrics: Total parameters, the total number of model parameters; Trainable parameters, the number of parameters optimized during ground-side training; Upload parameters, the number of parameters that must be transmitted to the satellite during deployment.

Stability-Oriented Configuration. Unless otherwise specified, we employ consistent settings across all methods to ensure a fair comparison. We evaluate the framework's sensitivity to the rank r, the learning rate ratio λ for output-side matrices, and the reduction factor for large modules. Rather than relying on exhaustive hyper-parameter tuning, our objective is to demonstrate that NE-LoRA provides stable and effective performance across non-extreme ranges, which is critical for autonomous in-orbit operation. The evaluated ranges for these factors are detailed in Appendix A.4.

Baselines. We compare NE-LoRA against the following baselines: (i) No Fine-tuning (NFT), which performs no parameter updates; (ii) Full Fine-tuning (FT), which updates all model parameters; (iii) LoRA [15], which freezes pretrained weights and introduces trainable low-rank adapters for downstream adaptation; (iv) LoRA+ [11], which assigns different learning rates to LoRA's adapter to improve optimization in wide models; (v) VERA [19], which shares a single pair of low-rank matrices across layers and learns lightweight scaling vectors to reduce the number of trainable parameters; (vi) DoRA [27], which decomposes pretrained weights into magnitude and direction for low-rank adaptation; and (vii) AdaLoRA [40], which adaptively allocates rank budgets across layers.

Table 1 summarizes the classification performance and parameter cost of all methods. NE-LoRA consistently achieves a decisive "low-cost, high-performance" advantage, yielding an average OA gain of 2.43 percentage points over FT. As visually demonstrated by the convergence of all NE-LoRA data points in the top-left corner of Fig. 4, our framework provides a highly scalable solution for high-fidelity onboard model updates under severe bandwidth constraints. These results suggest that integrating nonlinear refinement with a differentiated training strategy substantially improves the adaptation capability of low-rank updates for complex hyperspectral data.

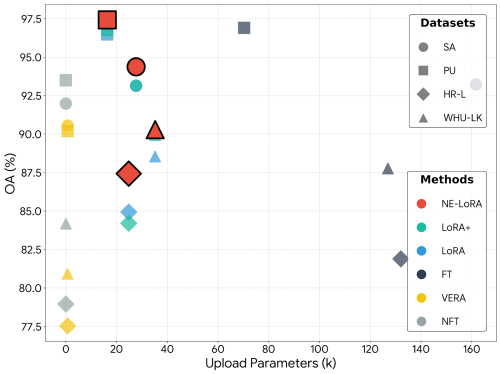


Figure 4: Accuracy vs. Communication.

5 Evaluation

5.1 Performance Analysis

Classification Accuracy. Classification accuracy is the primary metric for assessing adaptation quality in onboard hyperspectral applications. Compared with NFT, most adaptation methods improve classification accuracy, although the gains vary across methods and datasets. NE-LoRA consistently outperforms LoRA and LoRA+ on CNN3D across all four datasets. For instance, on HR-L, NE-LoRA achieves an OA of 87.43%, outperforming standard LoRA by 2.50 percentage points. In several settings, NE-LoRA also matches or exceeds full fine-tuning, suggesting that parameter-efficient adaptation can achieve competitive accuracy without updating all pretrained parameters. On PU, NE-LoRA reaches 97.42% OA, surpassing the 96.90% achieved by FT. Furthermore, NE-LoRA achieves higher OA than both DoRA and AdaLoRA on SA, PU, and HR-L, as well as higher AA across all four datasets. These results further demonstrate the effectiveness of NE-LoRA in achieving competitive adaptation performance under limited supervision while maintaining parameter efficiency.

Parameter Efficiency. Communication efficiency is a critical requirement for satellite deployment due to stringent uplink bandwidth constraints. NE-LoRA substantially reduces the number of parameters transmitted to satellites while achieving competitive or even superior accuracy compared with full fine-tuning (FT). On WHU-LK, NE-LoRA reduces the number of upload parameters from 127.0k (FT) to 35.2k on CNN3D, corresponding to an approximately 72% reduction in parameter transmission volume while achieving higher classification accuracy. By leveraging a branch-folding deployment strategy, NE-LoRA maintains the same upload parameter count as standard LoRA while improving adaptation performance. Specifically, the auxiliary branch is folded into two effective low-rank factors after ground-side training, avoiding additional parameters during deployment. For b-bit parameter storage, the raw payload of the folded factors is $(b/8)\sum_l r_l(d_l + k_l)$ bytes, excluding serialization metadata, tensor descriptors, and protocol headers. Assuming a 12 Mbps uplink rate, the estimated raw parameter transmission time on SA decreases from 431.2 ms for FT to 73.9 ms for NE-LoRA, corresponding to a 5.8$\times$ reduction. These results demonstrate that NE-LoRA can substantially reduce uplink communication overhead, facilitating parameter-efficient model updates within limited satellite–ground contact windows.

5.2 Ablation Studies

Table 2 presents the ablation results and highlights the complementary roles of the three key components in NE-LoRA. The GELU branch improves representational capacity, λ restores update balance across adapter matrices, and γ enhances training stability in large-proportion modules. Introducing the nonlinear GELU branch consistently improves classification accuracy over GELU-free variants. This result suggests that the auxiliary nonlinear branch effectively compensates for the limited expressiveness of purely linear low-rank updates in complex spectral–spatial nonlinearities of hyperspectral data. Incorporating λ further improves performance by compensating for the asymmetry between input matrices and output matrices, ensuring balanced feature updates. Adding γ improves OA in most evaluated settings, although the gains vary across models and datasets.

Table 2: Ablation analysis of different methods for each model.

Model	Method	SA (%)	PU (%)	HR-L (%)	WHU-LK (%)
CNN3D	GELU + λ + γ	**94.37**	**97.42**	**87.43**	90.29
	GELU + λ	94.06	96.97	85.97	**90.71**
	$\lambda + \gamma$	93.73	97.01	84.98	89.23
	GELU	92.37	96.68	84.16	88.78
M3DDCNN	GELU + λ + γ	**94.35**	**97.06**	**84.57**	**91.85**
	GELU + λ	93.75	96.68	83.06	91.00
	$\lambda + \gamma$	93.90	96.00	83.73	91.47
	GELU	92.85	96.02	82.87	91.71
HybridSN	GELU + λ + γ	90.84	**97.69**	**84.76**	**89.98**
	GELU + λ	**91.82**	97.07	84.68	88.22
	$\lambda + \gamma$	89.57	96.98	84.24	84.81
	GELU	88.76	96.99	84.67	89.16

6 Conclusion and Discussion

This work proposes NE-LoRA, a nonlinear-enhanced and differentiated adaptation framework for hyperspectral satellite models. NE-LoRA combines a GELU-activated auxiliary branch with customized learning-rate strategies designed to address asymmetric optimization dynamics across adapter matrices and instability in large-proportion modules. NE-LoRA delivers a decisive "low-cost, high-performance" advantage by achieving state-of-the-art classification accuracy while updating only a small fraction of total parameters, proving that high-fidelity hyperspectral model evolution is feasible even under the most stringent satellite uplink bandwidth constraints.

Our work also has several limitations. The current evaluation is centered on hyperspectral classification models and does not yet cover larger or more general pretrained architectures, which limits the present evidence for broader cross-domain generalization. Moreover, we focus on a practically motivated ground-side training and onboard deployment pipeline, but do not explicitly study long-term continual updating under time-varying orbital connectivity and resource dynamics. Our evaluation does not include direct measurements of onboard memory use, update integration latency, inference latency, or energy consumption. Finally, while NE-LoRA reduces the parameter transmission burden, a tighter integration with system-level factors such as contact-window scheduling, update latency, and multi-satellite coordination remains an important direction for future work.

Acknowledgments

Qiyang Zhang and Xinhao Li contributed equally to this work and share first authorship. Corresponding author: Jinfeng Wen. This work was supported in part by the National Natural Science Foundation of China under Grants 62425203, 62602023, 62602068.

References

[1] iperf: The tcp/udp bandwidth measurement tool, 1999. URL `https://www.telesat.com/blog/the-right-way-to-introduce-leo-services/`.

[2] Spacex successfully tests inter-satellite starlink connectivity via lasers, 2020. URL `https://wccftech.com/spacex-starlink-satellite-laser-test/`.

[3] Afia Anjum, Maksim E. Eren, Ismael Boureima, Boian Alexandrov, and Manish Bhattarai. Tensor train low-rank approximation (tt-lora): Democratizing ai with accelerated llms. In *Proceedings of International Conference on Machine Learning and Applications*, pages 583–590, 2024.

[4] Jose M. Bioucas-Dias, Antonio Plaza, Gustavo Camps-Valls, Paul Scheunders, Nasser Nasrabadi, and Jocelyn Chanussot. Hyperspectral remote sensing data analysis and future challenges. *IEEE Geoscience and Remote Sensing Magazine*, 1(2):6–36, 2013.

[5] Kejie Chen, Jean-Philippe Avouac, Saif Aati, Chris Milliner, Fu Zheng, and Chuang Shi. Cascading and pulse-like ruptures during the 2019 ridgecrest earthquakes in the eastern california shear zone. *Nature Communications*, 11(1):22, 2020.

[6] Minsik Cho, Saurabh Adya, and Devang Naik. Pdp: Parameter-free differentiable pruning is all you need. *Advances in Neural Information Processing Systems*, 36:45833–45855, 2023.

[7] Xin Deng, Scott Wagner, Dan Wang, Yuzhou Luo, and Kean S Goh. Pesticide detections, benchmarkexceedances, and temporal trends insurface water of california's imperial, salinas, and santa maria valleys. In *Proceedings of Pesticides in Surface Water: Monitoring, Modeling, Risk Assessment, and Management*, pages 119–142. 2019.

[8] Haonan Dong, Wenhao Zhu, Guojie Song, and Liang Wang. Aurora: Breaking low-rank bottleneck of lora with nonlinear mapping. *arXiv preprint arXiv:2505.18738*, 2025.

[9] Jonah Ekelund, Ricardo Vinuesa, Yuri Khotyaintsev, Pierre Henri, Gian Luca Delzanno, and Stefano Markidis. Ai in space for scientific missions: Strategies for minimizing neural-network model upload. In *Proceedings of IEEE 20th International Conference on e-Science*, pages 1–10, 2024.

[10] Margherita Grandini, Enrico Bagli, and Giorgio Visani. Metrics for multi-class classification: an overview. *arXiv preprint arXiv:2008.05756*, 2020.

[11] Soufiane Hayou, Nikhil Ghosh, and Bin Yu. Lora+: Efficient low rank adaptation of large models. *arXiv preprint arXiv:2402.12354*, 2024.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.

[13] Mingyi He, Bo Li, and Huahui Chen. Multi-scale 3d deep convolutional neural network for hyperspectral image classification. In *Proceedings of IEEE International Conference on Image Processing*, pages 3904–3908, 2017.

[14] Max Hort, Maria Kechagia, Federica Sarro, and Mark Harman. A survey of performance optimization for mobile applications. *IEEE Transactions on Software Engineering*, 48(8): 2879–2904, 2021.

[15] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *Proceedings of the International Conference on Learning Representations*, volume 1, page 3, 2022.

[16] Yeonjoon Jung, Daehyun Ahn, Hyungjun Kim, Taesu Kim, and Eunhyeok Park. Gralora: Granular low-rank adaptation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2505.20355*, 2025.

[17] K Karantzalos, Christina Karakizi, Zacharias Kandylakis, and Georgia Antoniou. Hyrank hyperspectral satellite dataset i (version v001). *Zenodo, Apr*, 2018.

[18] K Kavitha, S Arivazhagan, and B Suriya. Classification of pavia university hyperspectral image using gabor and svm classifier. *Int. J. New Trends Electron. Commun*, 2(3):9–14, 2014.

[19] Dawid J Kopiczko, Tijmen Blankevoort, and Yuki M Asano. Vera: Vector-based random matrix adaptation. *arXiv preprint arXiv:2310.11454*, 2023.

[20] Nico Lang, Walter Jetz, Konrad Schindler, and Jan Dirk Wegner. A high-resolution canopy height model of the earth. *Nature Ecology & Evolution*, 7(11):1778–1789, 2023.

[21] Jihao Li, Hewu Li, Zeqi Lai, Qian Wu, Yijie Liu, Qi Zhang, Yuanjie Li, and Jun Liu. Satguard: Concealing endless and bursty packet losses in leo satellite networks for delay-sensitive web applications. In *Proceedings of the ACM Web Conference*, pages 3053–3063, 2024.

[22] Shutao Li, Weiwei Song, Leyuan Fang, Yushi Chen, Pedram Ghamisi, and Jon Atli Benediktsson. Deep learning for hyperspectral image classification: An overview. *IEEE Transactions on Geoscience and Remote Sensing*, 57(9):6690–6709, 2019.

[23] Ying Li, Haokui Zhang, Xizhe Xue, Yenan Jiang, and Qiang Shen. Deep learning for remote sensing image classification: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(6):e1264, 2018.

[24] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017.

[25] Guoliang Lin, Yongheng Xu, Hanjiang Lai, and Jian Yin. Revisiting few-shot learning from a causal perspective. *IEEE Transactions on Knowledge and Data Engineering*, 36(11):6908–6919, 2024.

[26] Rong Liu, Zhilin Li, Jiaqi Yang, Jian Sun, and Quanwei Liu. Hyper-lkcnet: Exploring the utilization of large kernel convolution for hyperspectral image classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 18:13950–13966, 2025.

[27] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. In *Proceedings of 41st International Conference on Machine Learning*, 2024.

[28] Yulong Mao, Kaiyu Huang, Changhao Guan, Ganglin Bao, Fengran Mo, and Jinan Xu. Dora: Enhancing parameter-efficient fine-tuning with dynamic rank distribution. *arXiv preprint arXiv:2405.17357*, 2024.

[29] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14297–14306, 2023.

[30] Joel R Norris, Robert J Allen, Amato T Evan, Mark D Zelinka, Christopher W O'Dell, and Stephen A Klein. Evidence for climate change in the satellite cloud record. *Nature*, 536(7614): 72–75, 2016.

[31] Rushi Qiang, Ruiyi Zhang, and Pengtao Xie. Bilora: A bi-level optimization framework for overfitting-resilient low-rank adaptation of large pre-trained models. *arXiv preprint arXiv:2403.13037*, 2024.

[32] Swalpa Kumar Roy, Gopal Krishna, Shiv Ram Dubey, and Bidyut B. Chaudhuri. Hybridsn: Exploring 3-d–2-d cnn feature hierarchy for hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*, 17(2):277–281, February 2020. ISSN 1558-0571. doi: 10.1109/lgrs.2019.2918719. URL `http://dx.doi.org/10.1109/LGRS.2019.2918719`.

[33] Qinliang Su, Lawrence Carin, et al. A probabilistic framework for nonlinearities in stochastic neural networks. *Advances in Neural Information Processing Systems*, 30, 2017.

[34] Aysim Toker, Lukas Kondmann, Mark Weber, Marvin Eisenberger, Andrés Camero, Jingliang Hu, Ariadna Pregel Hoderlein, Çağlar Şenaras, Timothy Davis, Daniel Cremers, et al. Dynamicearthnet: Daily multi-spectral satellite dataset for semantic change segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21158–21167, 2022.

[35] Yanxin Xi, Tong Li, Huandong Wang, Yong Li, Sasu Tarkoma, and Pan Hui. Beyond the first law of geography: Learning representations of satellite imagery by leveraging point-of-interests. In *Proceedings of the ACM Web Conference*, pages 3308–3316, 2022.

[36] Mengwei Xu, Dongqi Cai, Yaozong Wu, Xiang Li, and Shangguang Wang. {FwdLLM}: Efficient federated finetuning of large language models with perturbed inferences. In *Proceedings of USENIX Annual Technical Conference*, pages 579–596, 2024.

[37] Jun Yang, Peng Gong, Rong Fu, Minghua Zhang, Jingming Chen, Shunlin Liang, Bing Xu, Jiancheng Shi, and Robert Dickinson. The role of satellite remote sensing in climate change studies. *Nature Climate Change*, 3(10):875–883, 2013.

[38] Menglin Yang, Jialin Chen, Yifei Zhang, Jiahong Liu, Jiasheng Zhang, Qiyao Ma, Harshit Verma, Qianru Zhang, Min Zhou, Irwin King, et al. Low-rank adaptation for foundation models: A comprehensive review. *arXiv preprint arXiv:2501.00365*, 2024.

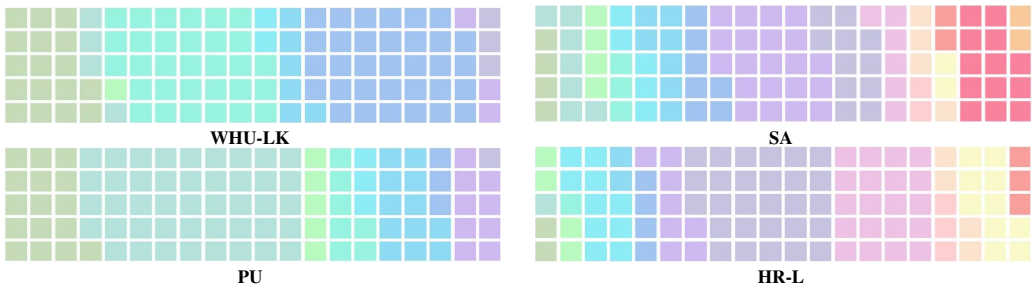


Figure 5: Hyperspectral dataset overview. Different colors in the upper cells denote various ground object categories within each dataset, where color proportion indicates the relative prevalence of each class.

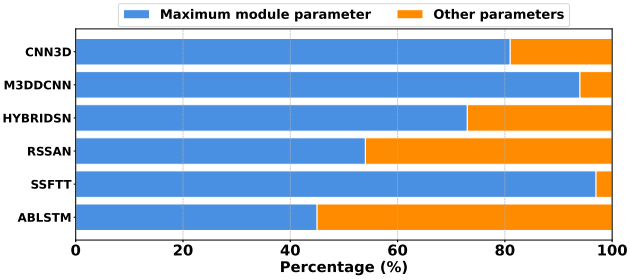


Figure 6: Distribution of parameter proportions for maximum-volume modules across different models.

[39] Yuchen Zeng and Kangwook Lee. The expressive power of low-rank adaptation. *arXiv preprint arXiv:2310.17513*, 2023.

[40] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*, 2023.

[41] Yanfei Zhong, Xin Hu, Chang Luo, Xinyu Wang, Ji Zhao, and Liangpei Zhang. Whu-hi: Uav-borne hyperspectral with high spatial resolution (h2) benchmark datasets and classifier for precise crop identification based on deep convolutional neural network with crf. *Remote Sensing of Environment*, 250:112012, 2020.

A Appendix

A.1 Hyperspectral Dataset Overview

Fig. 5 illustrates the category distributions of the four hyperspectral datasets. The panels visualize the class composition of each dataset, highlighting their diversity in both category coverage and class proportion. This diversity indicates that the evaluated datasets span heterogeneous scene characteristics, making them suitable for assessing the robustness and generality of the proposed method.

A.2 Distribution of Parameter Proportions

Fig. 6 shows the parameter proportion of the largest module in each backbone model. The results indicate that several representative hyperspectral backbones contain one dominant module that accounts for a substantial fraction of the total parameters. This observation motivates the large-proportion module analysis in our differentiated training strategy. Because such modules contribute disproportionately to the overall update, they are more sensitive to learning-rate choices and can strongly influence optimization stability.

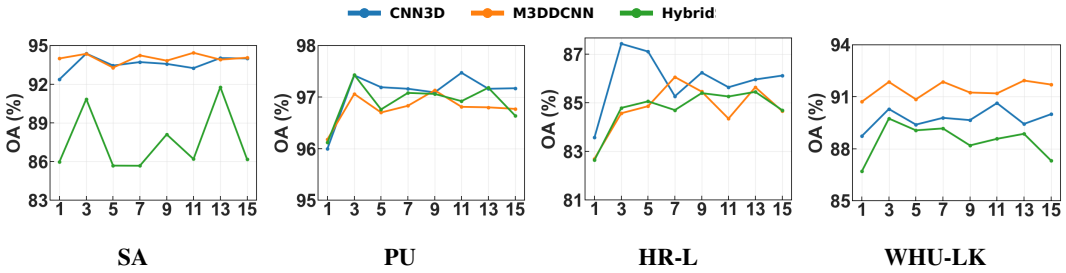


Figure 7: Accuracy variation of models across datasets with varying rank.

A.3 Diverse Datasets

On M3DDCNN, LoRA and LoRA+ perform similarly on PU. NE-LoRA maintains clear advantages, particularly on SA and HR-L, outperforming LoRA, LoRA+, and FT. These results demonstrate the effectiveness of NE-LoRA across diverse hyperspectral scenes.

On HybridSN, FT again provides a strong baseline, whereas NFT and VERA perform relatively poorly. LoRA and LoRA+ yield only modest gains, with LoRA+ narrowing but not closing the gap with FT. By contrast, NE-LoRA achieves the best or near-best performance on several datasets: it surpasses FT on PU (OA: 97.43%) and WHU-LK (OA: 89.74%), achieves an OA of 90.84% on SA, and remains competitive on HR-L. These results indicate that combining nonlinear enhancement with differentiated training can effectively compensate for the limitations of purely linear low-rank adaptation. Although the absolute OA gains of NE-LoRA are sometimes modest, typically around 1–5%, such improvements are meaningful for hyperspectral image classification, where spectral–spatial distinctions are often subtle and difficult to model. In particular, HR-L yields the lowest accuracies across most methods because of its heterogeneous land-cover patterns and spectral ambiguity. Even in this challenging setting, NE-LoRA still demonstrates robust fine-tuning capability, suggesting that the proposed method generalizes well to complex hyperspectral scenes.

Across the four datasets, these upload parameters average 25.9% on M3DDCNN, and just 2.5% on HybridSN. Crucially, NE-LoRA maintains the same upload parameter count as LoRA through its branch-folding deployment strategy. The auxiliary refinement branch improves training but is folded into the deployed update afterward, so only lightweight low-rank parameters need to be transmitted. By contrast, although VERA updates very few parameters, its post-update accuracy remains unsatisfactory. This result suggests that hyperspectral data, characterized by complex spectral overlap and spatial heterogeneity, cannot always be adapted effectively with extremely small update budgets.

A.4 Stability and Robustness Analysis

Robustness to Low-Rank Constraints. We first investigate the stability of NE-LoRA under varying low-rank constraints r. As illustrated in Fig. 7, the classification accuracy saturates at a very small rank (typically $r = 3$) and remains consistently high thereafter. This trend reinforces the intrinsic low-rank property of hyperspectral features, suggesting that essential spectral–spatial patterns can be encapsulated within a minimal parameter budget. Notably, even in the extreme case of $r = 1$, NE-LoRA maintains high-level semantic capture and competitive accuracy. This serves as a strong validation that our nonlinear auxiliary branch (GELU) effectively compensates for the limited representational capacity of purely linear adapters, allowing for "High Performance" even under the most stringent "Low Cost" constraints.

Robustness to Optimization Hyper-parameters. Next, we perform a robustness validation of the learning rate modulation factor γ for large-proportion modules. Fig. 8 presents the accuracy distribution across a wide range of γ settings, with the red dashed line representing the baseline without modulation. The results show that performance remains at or above the baseline across nearly all evaluated γ values. This parameter insensitivity is vital for LEO satellite nodes, where limited onboard energy and intermittent connectivity make frequent, model-specific fine-tuning infeasible. By moderating input-side updates, γ prevents aggressive optimization and unstable gradient dynamics in high-dimensional modules while preserving sufficient adaptation capacity.

Table 3: Performance comparison of different methods on M3DDCNN. Values in parentheses for OA, AA, and Kappa indicate the improvement of NE-LoRA over the average of the other baselines on each dataset, while those in the Upload column indicate the reduction factor in uplink parameters relative to FT.

Dataset	Method	OA (%)	AA (%)	Kappa ($\times 100$)	Total	Train	Upload
SA	NFT	92.12	94.79	91.24	273.1k	0	0
	FT	92.85	95.18	92.04	273.1k	273.1k	273.1k
	LoRA	93.33	95.78	92.57	324.7k	51.5k	51.5k
	LoRA+	93.75	96.18	93.04	324.7k	51.5k	51.5k
	VERA	88.03	86.51	86.63	325.0k	0.4k	0.4k
	DoRA	93.80	96.02	93.09	274.8k	1.7k	1.7k
	AdaLoRA	93.55	95.73	92.81	276.4k	3.3k	1.8k
	NE-LoRA	**94.35 (+1.86)**	**96.53 (+2.22)**	**93.71 (+2.08)**	376.2k	103.1k	51.5k ($5.3\times$)
PU	NFT	93.90	91.09	91.89	81.9k	0	0
	FT	95.49	95.49	94.05	81.9k	81.9k	81.9k
	LoRA	96.13	95.11	94.87	107.3k	25.4k	25.4k
	LoRA+	95.92	95.34	94.61	107.3k	25.4k	25.4k
	VERA	86.51	82.93	81.54	107.7k	0.4k	0.4k
	DoRA	96.94	95.76	95.93	83.6k	1.7k	1.7k
	AdaLoRA	96.37	95.36	95.18	85.2k	3.3k	1.9k
	NE-LoRA	**97.06 (+2.59)**	**96.42 (+3.41)**	**96.11 (+3.53)**	132.7k	50.8k	25.4k ($3.2\times$)
HR-L	NFT	77.26	66.38	72.54	208.6k	0	0
	FT	82.04	73.07	78.46	208.6k	208.6k	208.6k
	LoRA	82.08	73.49	78.57	253.2k	44.6k	44.6k
	LoRA+	82.63	74.79	79.39	253.2k	44.6k	44.6k
	VERA	71.82	55.82	65.96	253.6k	0.4k	0.4k
	DoRA	81.77	74.09	78.27	210.3k	1.7k	1.7k
	AdaLoRA	81.35	72.60	77.70	211.9k	3.3k	1.8k
	NE-LoRA	**84.57 (+4.72)**	**77.39 (+7.36)**	**81.63 (+5.79)**	297.8k	89.2k	44.6k ($4.7\times$)
WHU-LK	NFT	89.83	63.41	86.50	210.9k	0	0
	FT	91.30	63.80	88.39	210.9k	210.9k	210.9k
	LoRA	91.21	64.69	88.28	279.3k	68.4k	68.4k
	LoRA+	91.19	64.90	88.26	279.3k	68.4k	68.4k
	VERA	89.61	59.45	86.18	279.7k	0.4k	0.4k
	DoRA	91.16	**70.17**	88.34	212.6k	1.7k	1.7k
	AdaLoRA	90.16	67.82	87.06	214.2k	3.3k	2.2k
	NE-LoRA	**91.85 (+1.21)**	66.87 (+1.98)	**89.11 (+1.54)**	347.7k	136.8k	68.4k ($3.1\times$)

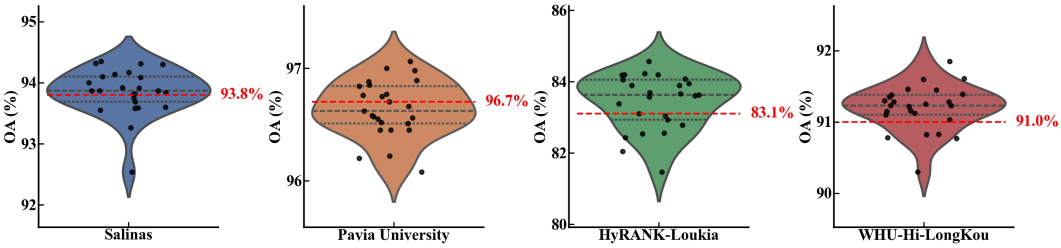


Figure 8: Accuracy distribution across datasets with fixed λ and variable γ for large matrices.

Overall, these findings confirm that NE-LoRA is highly robust to hyper-parameter selection, ensuring reliable model evolution in dynamic orbital environments.

Table 4: Performance comparison of different methods on HybridSN. Values in parentheses for OA, AA, and Kappa indicate the improvement of NE-LoRA over the average of the other baselines on each dataset, while those in the Upload column indicate the reduction factor in uplink parameters relative to FT.

Dataset	Method	OA (%)	AA (%)	Kappa ($\times 100$)	Total	Train	Upload
SA	NFT	85.53	67.33	83.83	452.7k	0	0
	FT	**92.79**	**82.92**	**91.96**	452.7k	452.7k	452.7k
	LoRA	89.98	74.02	88.82	463.9k	11.2k	11.2k
	LoRA+	89.21	73.56	87.96	463.9k	11.2k	11.2k
	VERA	86.78	74.45	85.26	465.2k	1.3k	1.3k
	DoRA	90.97	79.74	89.93	464.4k	11.7k	11.7k
	AdaLoRA	86.14	67.67	84.52	475.2k	22.4k	14.5k
	NE-LoRA	90.84 (+2.07)	80.12 (+5.88)	89.80 (+2.33)	475.1k	22.4k	11.2k (40.4×)
PU	NFT	95.64	92.04	94.21	451.8k	0	0
	FT	97.34	94.68	96.47	451.8k	451.8k	451.8k
	LoRA	94.31	85.37	92.43	463.0k	11.2k	11.2k
	LoRA+	97.05	94.71	96.08	463.0k	11.2k	11.2k
	VERA	87.09	75.55	82.78	464.3k	1.3k	1.3k
	DoRA	95.41	89.92	93.91	456.0k	4.2k	4.2k
	AdaLoRA	95.21	88.48	93.65	459.6k	7.7k	3.6k
	NE-LoRA	**97.43 (+2.85)**	**95.07 (+6.39)**	**96.59 (+3.80)**	474.2k	22.3k	11.2k (40.3×)
HR-L	NFT	78.96	62.38	74.86	452.5k	0	0
	FT	**86.83**	74.33	**84.30**	452.5k	452.5k	452.5k
	LoRA	84.07	70.10	81.05	463.7k	11.2k	11.2k
	LoRA+	85.69	**77.11**	82.96	463.7k	11.2k	11.2k
	VERA	72.03	50.88	66.34	465.0k	1.3k	1.3k
	DoRA	79.05	62.65	75.04	456.6k	4.2k	4.2k
	AdaLoRA	78.54	62.48	74.47	460.2k	7.7k	3.9k
	NE-LoRA	84.79 (+4.05)	68.39 (+2.69)	81.87 (+4.87)	474.9k	22.4k	11.2k (40.4×)
WHU-LK	NFT	80.97	54.99	74.45	451.8k	0	0
	FT	88.83	68.10	85.23	451.8k	451.8k	451.8k
	LoRA	83.45	61.38	78.08	463.0k	11.2k	11.2k
	LoRA+	85.17	61.60	80.23	463.0k	11.2k	11.2k
	VERA	84.96	51.66	79.63	464.3k	1.3k	1.3k
	DoRA	**89.99**	**69.03**	**86.73**	456.0k	4.2k	4.2k
	AdaLoRA	89.37	67.77	85.89	459.6k	7.7k	3.3k
	NE-LoRA	89.74 (+3.63)	68.13 (+6.05)	86.45 (+4.99)	474.2k	22.3k	11.2k (40.3×)

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we

acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our contributions are summarized in a list in §1.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We thoroughly discuss the limitations of our work in §6.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: This paper does not include theoretical results.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide detailed instructions on how to reproduce the main experimental results in §5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: We do not publicly release code, model checkpoints, or datasets with this paper, as we plan to build on this work in follow-up research.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide detailed instructions on how to reproduce the main experimental results in §4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Error bars are not reported because of the time limit. We will attempt to add them in the camera-ready version.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide detailed hardware information in §4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and believe that our research conforms to it.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

Justification: We have discussed and provided real-world examples of both positive and negative societal impacts in §1 and §2.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: This paper is intended for onboard model adaptation and does not involve high-risk data or models.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We have properly cited the original data and models in §4.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: No new code, model checkpoints, or datasets are released with this paper, as we plan to build on this work in follow-up research.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not part of the core methodology of this work. Any LLM usage was limited to writing or editing assistance and did not affect the scientific content, experiments, or originality of the research.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.