# BASIS-TO-LORA: LEARNING TO GENERATE LORA ADAPTERS FOR TASK ADAPTATION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Parameter-efficient fine-tuning methods such as Low-Rank Adapterss (LoRAs) have become a standard approach for adapting large language models to downstream tasks. Nevertheless, training a separate LoRA adapter for every new task remains computationally expensive. Weight-space learning offers an alternative by training a hypernetwork to generate adapter weights directly. However, predicting full weight updates is challenging due to their high dimensionality, while predicting LoRA factors introduces non-identifiability from invariance to a group of linear transformations. In this work, we show that LoRA updates trained on related datasets approximately share a low-dimensional subspace, and leverage this structure to improve weight-space learning. Our method learns a shared basis from a collection of trained LoRA checkpoints and constrains the hypernetwork to predict only the coordinates of a new adapter within this basis. This formulation provides an identifiable representation while substantially reducing the hypernetwork's output dimension. We evaluate our approach on commonsense reasoning and mathematical reasoning benchmarks, where it consistently outperforms weight-space learning baselines. On Qwen2.5-1.5B, our method achieves 33.7% accuracy on MATH and 69.3% on GSM8K, improving over Drag-and-Drop by 9.8 and 3.0 percentage points, respectively.

## 1 INTRODUCTION

Large Language Models (LLMs) trained on massive text corpora have demonstrated remarkable capabilities across a wide range of downstream tasks. Although LLMs show strong zero-shot performance on many downstream datasets, Supervised Fine-Tuning (SFT) is commonly used to further adapt an LLM to a specific task. SFT updates all model parameters using gradients computed on task-specific data. While effective, fine-tuning is computationally expensive and requires significant memory and storage for each adapted model. Rather than updating the entire model, Parameter-Efficient Fine-Tuning (PEFT) learns a small set of additional parameters while keeping the pre-trained weights frozen. Low-Rank Adaptation (LoRA) (Hu et al., 2022) has emerged as one of the most widely adopted approaches for PEFT by learning task-specific low-rank weight updates. In practice, however, LoRA adapters must be optimized using task-specific gradients so training remains a computationally expensive procedure.

Although training individual Low-Rank Adapters (LoRA) adapters is expensive, the cost is modest relative to fine-tuning. Consequently, thousands of task-specific LoRA adapters have been publicly released. The cost of training a new LoRA adapter and the availability of many existing adapters create a new opportunity to amortize the process of building adapters. Weight-Space Learning (WSL) is a promising direction to amortize training models and generate new model weights on the fly for downstream tasks. Weight Space Learning (Horwitz et al., 2025) uses a hypernetwork (Han et al., 2026) to map from datasets onto model weights; for example, hypernetworks conditioned on task descriptions, few-shot examples, or unlabeled task prompts (Phang et al., 2023; Charakorn et al., 2025; Liang et al., 2025). These methods directly predict LoRA parameters — for example, Text-to-LoRA generates the low-rank factors, while Drag-and-Drop (DnD) generates the full set of LoRA matrices (Charakorn et al., 2025; Liang et al., 2025). Consequently, the hypernetwork must solve a high-dimensional weight-prediction problem. This increases the complexity of the optimization problem, making generalization to unseen tasks increasingly difficult.



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

*Can we further simplify the process of generating LoRA adapters?* Recent work has provided empirical evidence that LoRA adapters trained across diverse tasks and datasets share a universal low-dimensional weight subspace (Kaushik et al., 2025; Gabrielsson et al., 2025). Motivated by this observation, we ask a natural question: can we simplify weight-space learning by predicting only the task-specific coordinates within a shared subspace. Instead of training the hypernetwork to predict the weights directly, we can simplify the learning problem by only learning to predict the low-rank coordinate within the shared sub-space, obtained from a zoo of LoRA adapters.

In this work, we propose **Basis-to-LoRA (B2L)**, a novel method for generating LoRA weights by learning a mapping from dataset prompts to LoRA weights. We extract a shared basis from a zoo of trained LoRA weights. A hypernetwork is trained to learn the mapping from input prompts to low-rank coordinates in this shared basis. This formulation offers two key advantages. First, it substantially simplifies the learning problem by reducing the hypernetwork's output space from high-dimensional weight matrices to low-rank coordinates. Second, it reframes LoRA generation as a subspace localization problem: given a downstream task, the objective is to identify an appropriate region of the shared weight subspace and predict its coordinates, rather than synthesizing the adapter weights from scratch.

Our contributions are as follows:

1. We formalize LoRA weight-space learning as prediction over a set of weight subspaces.
2. We build a hypernetwork to learn this subspace directly from a large-scale zoo of pre-trained adapters using joint diagonalization.
3. We empirically demonstrate the effectiveness of the shared-basis parameterization for LoRA weight-space learning. We validate our hypothesis using the DnD baseline (Liang et al., 2025); On Qwen2.5-1.5B, our method achieves 33.7% on MATH and 69.5% on GSM8K, outperforming the Drag-and-Drop baseline by 9.8 and 3.2 percentage points, respectively. Our method is simple to implement and can be integrated into existing WSL methods as a plug-in reparameterization.

## 2 Background

**Parameter-Efficient Fine-Tuning (PEFT).** Fine-tuning all parameters of a large pretrained model is computationally expensive and requires storing a separate set of model weights for each downstream task. PEFT methods were proposed to reduce the computational cost and time of fine-tuning Large Language Models (LLMs) PEFT methods overcome this limitation by only updating a subset of model-weights or learning lower-dimensional updates. Aghajanyan et al. (2021) showed that effective language-model adaptation can be achieved by optimizing within a low-dimensional subspace of the full parameter space. Motivated by the observation that weight updates during adaptation have a low intrinsic rank, Hu et al. (2022) introduced Low-Rank Adaptation (LoRA), which freezes the pretrained model weights and parameterizes each task-specific update as the product of two low-rank matrices. LoRA parameterizes the update to a pretrained weight matrix $\mathbf{W}_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ as

$$\mathbf{W} = \mathbf{W}_0 + \Delta\mathbf{W}, \qquad \Delta\mathbf{W} = \mathbf{B}\mathbf{A}, \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{r \times d_{\text{in}}}$, $\mathbf{B} \in \mathbb{R}^{d_{\text{out}} \times r}$, and $r \ll \min(d_{\text{in}}, d_{\text{out}})$. Only $\mathbf{A}$ and $\mathbf{B}$ are optimized, substantially reducing the number of trainable and task-specific parameters. By using low-rank adapters, LoRA reduces the number of trainable parameters for each weight matrix from $d_{\text{out}}d_{\text{in}}$ to $r(d_{\text{out}} + d_{\text{in}})$. Subsequent LoRA-based methods have further improved parameter allocation, memory efficiency, and optimization dynamics (Dettmers et al., 2023). Nevertheless, these methods generally train each adapter independently, requiring a separate gradient-based optimization procedure whenever a new task is introduced (Phang et al., 2023).

**Weight-Space Learning (WSL).** WSL treats trained neural-network parameters as structured data and aims to learn representations or distributions directly over them, which can later be sampled or predicted (Schürholt et al., 2022; Han et al., 2026; Knyazev et al., 2021). Given a collection of trained models or adapters, commonly referred to as a *model zoo*, WSL methods can predict model properties, learn compact representations of checkpoints, or generate parameters for previously unseen configurations. Generative approaches use hypernetworks, autoencoders, diffusion models, or

other conditional generative models to map task or dataset representations to model parameters (Ha et al., 2017; Schürholt et al., 2022; Yang et al., 2026; Wang et al., 2025). More generally, WSL amortizes task-specific optimization by learning a mapping $h_\phi$ from a task or dataset representation ($z_\tau$) to model parameters ($\theta_\tau$) adapted to that task:

$$h_\phi(\mathbf{z}_\mathcal{T}) \longmapsto \boldsymbol{\theta}_\mathcal{T}. \tag{2}$$

For example, Hypter generates lightweight adapters from natural-language task descriptions (Ye & Ren, 2021), while Text-to-LoRA generates task-specific LoRA parameters using a single hypernetwork forward pass (Charakorn et al., 2025). Charakorn et al. (2026) proposed Doc-to-LoRA, which trains a hypernetwork to generate a LoRA adapter from a document, allowing a model to answer subsequent queries without receiving the document again. Such methods enable rapid adaptation and can potentially generalize to tasks not observed during hypernetwork training. However, existing approaches commonly generate the complete set of adapter parameters directly. Although smaller than the underlying model, the collection of LoRA matrices across all layers remains a high-dimensional and structured output space, making the hypernetwork difficult to train from a limited adapter zoo. To address this, Liang et al. (2025) proposed DnD, which uses a pre-trained tokenizer to reduce the dimensionality of the weights.

**Shared and universal weight subspaces.** Multiple works have studied low-dimensional structure in both pretrained and fine-tuned model weights. Aghajanyan et al. (2021) showed that effective fine-tuning can be performed within a low-dimensional subspace of the full parameter space. More recent work has directly investigated whether weights or updates from different models can be represented using shared subspaces. Gabrielsson et al. (2025) jointly compressed collections of LoRA adapters using shared left and right bases together with adapter-specific coordinate matrices. Similarly, Kaushik et al. (2025) extracted principal subspaces from collections of pretrained adapters and adapted to new tasks by learning only lightweight coefficients within those subspaces. At a broader scale, the Universal Weight Subspace Hypothesis proposes that models trained across different initializations, tasks, and domains exhibit similar dominant spectral subspaces (Kaushik et al., 2025). These findings motivate a more structured formulation of weight-space learning.

Rather than requiring a hypernetwork to generate every entry of the LoRA matrices, we first extract a collection of shared bases from a zoo of pretrained adapters and represent each task-specific update through its coordinates within these bases. In particular, an update can be approximated as

$$\Delta\mathbf{W}_\mathcal{T} \approx \mathbf{U}_{k(\mathcal{T})}\mathbf{C}_\mathcal{T}\mathbf{V}_{k(\mathcal{T})}^\top, \tag{3}$$

where $\mathbf{U}_k$ and $\mathbf{V}_k$ define a shared subspace, $k(\mathcal{T})$ selects the subspace associated with task $\mathcal{T}$, and $\mathbf{C}_\mathcal{T}$ contains the task-specific coordinates. Weight generation is therefore reduced to predicting an appropriate subspace and its low-dimensional coordinates. This formulation incorporates the geometry observed across the adapter zoo as an inductive bias, reduces the dimensionality of the hypernetwork output, and reframes LoRA generation from unrestricted weight prediction as a structured subspace-selection and coordinate-prediction problem.

## 3 Basis-to-LoRA: Learning to generate LoRA Adapters

The growing collection of publicly available LoRA adapters trained on diverse datasets provides an opportunity to learn directly from their weights. However, directly learning the manifold of LoRA adapters is difficult for two reasons: (1) although LoRA is a low-rank PEFT method, its adapters still contain a large number of parameters for WSL methods to model directly; and (2) the LoRA factorization $\Delta\mathbf{W} = \mathbf{BA}$ is non-identifiable under the action of a linear transformation $\mathbf{G} \in \mathrm{GL}(r)$ from the group of invertible $r \times r$ matrices $\mathrm{GL}(r)$, since $(\mathbf{BG})(\mathbf{G}^{-1}\mathbf{A})$ represents the same model update. Consequently, directly learning from the factors $\mathbf{A}$ and $\mathbf{B}$ without accounting for this symmetry can make equivalent adapters appear different in weight space (Putterman et al., 2025; da Silva et al., 2026). Learning from the invariant update $\Delta\mathbf{W}$ avoids this ambiguity, but requires operating in the much higher-dimensional ambient weight space. Motivated by this, our method, **B2L**, aims to learn a representation of trained LoRA adapters that is both symmetry-invariant and low-dimensional. In section 3.1, we first discuss the non-identifiability of the LoRA factorization and explain how learning a shared basis directly from the weight updates yields a representation invariant to the $\mathrm{GL}(r)$ symmetry. This produces low-dimensional coordinates that do not depend on

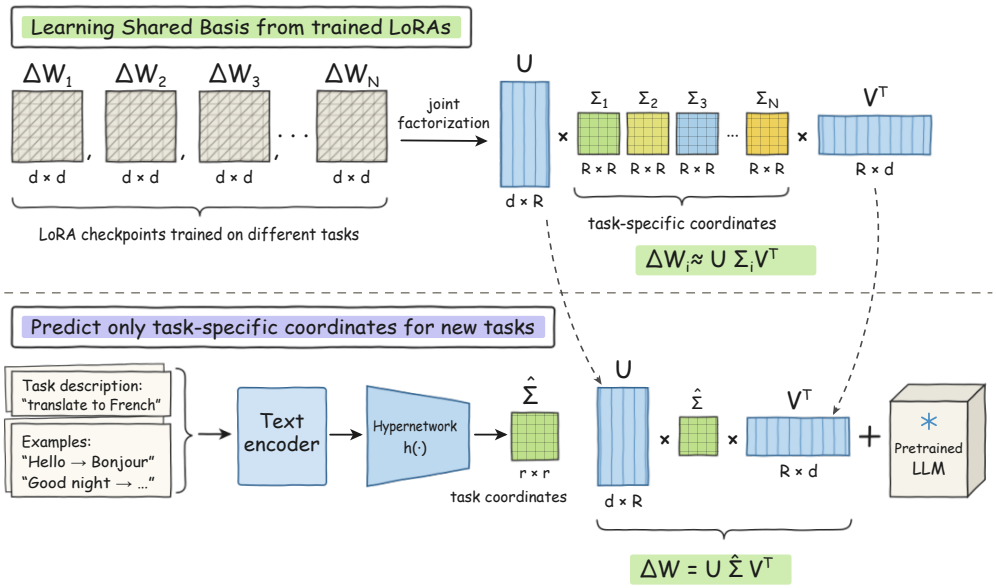



Figure 1: **Schematic view of B2L**. We extract a shared basis (matrices $\mathbf{U}, \mathbf{V}$) from a collection of LoRA checkpoints trained on different tasks, and use the basis to project the original LoRA checkpoint into task-specific low-rank coordinates. This reduces the output dimension of the hypernetwork, which is trained to predict the task-specific coordinate instead of the original weight matrices.

the arbitrary choice of LoRA factorization. In section 3.2, we explain how to learn this shared basis from a collection of trained LoRA adapters.

### 3.1 LoRA $\mathrm{GL}(r)$ Symmetry

LoRA represents the weight update as a product of two low-rank factors, $\Delta\mathbf{W} = \mathbf{BA}$. This factorization, however, is not unique: for any invertible matrix $\mathbf{G} \in \mathrm{GL}(r)$, the factors $\mathbf{BG}$ and $\mathbf{G}^{-1}\mathbf{A}$ produce the same update $\Delta\mathbf{W}$, Therefore, for any invertible matrix $\mathbf{G} \in \mathrm{GL}(r)$, the transformation

$$(\mathbf{B}, \mathbf{A}) \mapsto (\mathbf{BG}, \mathbf{G}^{-1}\mathbf{A}) \tag{4}$$

leaves the resulting update unchanged, since

$$(\mathbf{BG})(\mathbf{G}^{-1}\mathbf{A}) = \mathbf{BA} = \Delta\mathbf{W}. \tag{5}$$

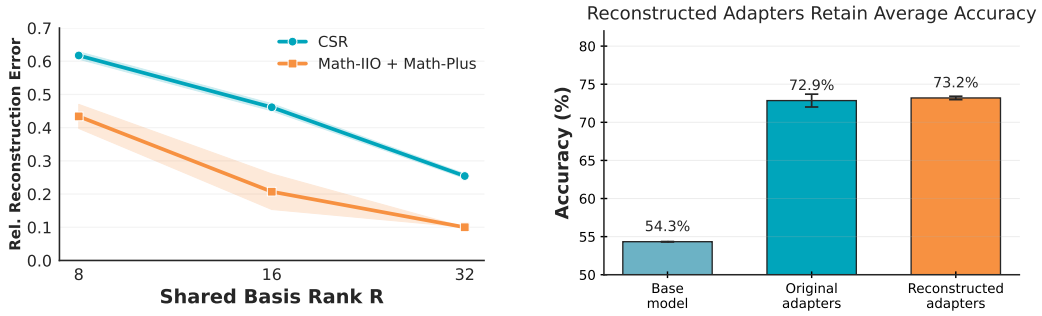
This symmetry creates an ill-posed learning target for a hypernetwork that predicts $\mathbf{A}$ and $\mathbf{B}$ directly. Similar LoRA adapters that may be functionally similar or even equivalent can be very far apart in the weight space. This makes it difficult to learn a mapping from a dataset representation to weight representations as training may penalize a predicted pair even when it represents the correct update under a different choice of $G$. Operating on the product $\Delta\mathbf{W} = \mathbf{BA}$ removes this ambiguity because the product is invariant under the transformation in eq. (4). However, directly predicting $\Delta\mathbf{W}$ requires operating in the high-dimensional ambient space $\mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$. Our method, **B2L**, addresses this challenge by mapping each LoRA weight update $\Delta\mathbf{W}_i$ to a low-dimensional coordinate matrix $\mathbf{\Sigma}_i$ in a pair of shared bases. Let $\mathbf{U}$ and $\mathbf{V}$ denote the shared bases for the output and input spaces, respectively, and define

$$\mathbf{\Sigma}_i = \mathbf{U}^\top \Delta\mathbf{W}_i \mathbf{V}.$$

Since $\Delta\mathbf{W}_i = \mathbf{B}_i\mathbf{A}_i$, the coordinate matrix after the transformation in eq. (4) is given by:

$$\mathbf{\Sigma}_i' = \mathbf{U}^\top(\mathbf{B}_i\mathbf{G})(\mathbf{G}^{-1}\mathbf{A}_i)\mathbf{V} = \mathbf{U}^\top\Delta\mathbf{W}_i\mathbf{V} = \mathbf{\Sigma}_i. \tag{6}$$

Thus, for fixed shared bases learned from the weight updates, $\mathbf{\Sigma}_i$ is invariant to the choice of LoRA factors. We now explain how to learn the shared bases from trained LoRA adapters.




(a) Reconstruction error decreases with shared-basis rank.

(b) Reconstructed adapters preserve downstream accuracy.

Figure 2: Shared-basis reconstruction on common-sense reasoning adapters from ARC-c, ARC-e, BoolQ, HellaSwag, OBQA, PIQA, and WinoGrande. Error is measured over 711 adapter checkpoints; accuracy is evaluated on 5 sampled checkpoints per dataset.

### 3.2 Learning Shared Basis from LoRA Adapters

Recent work has shown that LoRA adapters trained on similar tasks often share a low-dimensional subspace. We aim to find shared bases $\mathbf{U}$ and $\mathbf{V}$ that map these updates into a low-dimensional subspace with low reconstruction error. To learn the shared bases from $\{\Delta\mathbf{W}_i\}$, we use the Joint-Diagonalization approximation method (Gabrielsson et al., 2025). We want to obtain shared bases from a collection of $N$ trained LoRA updates, where each update is given by

$$\Delta\mathbf{W}_i = \mathbf{B}_i\mathbf{A}_i, \qquad i = 1, \ldots, N. \tag{7}$$

$$\Delta\mathbf{W}_i \approx \mathbf{U}\mathbf{\Sigma}_i\mathbf{V}^\top, \tag{8}$$

where $\mathbf{U} \in \mathbb{R}^{d_{\text{out}} \times r}$ and $\mathbf{V} \in \mathbb{R}^{d_{\text{in}} \times r}$ are shared across all checkpoints, while $\mathbf{\Sigma}_i \in \mathbb{R}^{r \times r}$ contains the task-specific coordinates. The Joint Diagonalization method (Gabrielsson et al., 2025) obtains the shared bases by solving:

$$\min_{\substack{\mathbf{U}^\top\mathbf{U}=I_r,\ \mathbf{V}^\top\mathbf{V}=I_r \\ \{\mathbf{\Sigma}_i\}_{i=1}^N}} \sum_{i=1}^{N} \left\|\mathbf{B}_i\mathbf{A}_i - \mathbf{U}\mathbf{\Sigma}_i\mathbf{V}^\top\right\|_F^2. \tag{9}$$

For fixed shared bases $\mathbf{U}$ and $\mathbf{V}$, the optimal coordinate matrix is given by:

$$\mathbf{\Sigma}_i = \mathbf{U}^\top\mathbf{B}_i\mathbf{A}_i\mathbf{V}. \tag{10}$$

The bases are optimized by alternating eigendecomposition updates:

$$\mathbf{U} \leftarrow \text{Eig}_r\left(\sum_{i=1}^{N} \mathbf{B}_i\mathbf{A}_i\mathbf{V}\mathbf{V}^\top\mathbf{A}_i^\top\mathbf{B}_i^\top\right), \tag{11}$$

$$\mathbf{V} \leftarrow \text{Eig}_r\left(\sum_{i=1}^{N} \mathbf{A}_i^\top\mathbf{B}_i^\top\mathbf{U}\mathbf{U}^\top\mathbf{B}_i\mathbf{A}_i\right), \tag{12}$$

where $\text{Eig}_r(\cdot)$ returns the eigenvectors associated with the $r$ largest eigenvalues. $\mathbf{\Sigma}_i$ and the shared bases ($\mathbf{U}$ and $\mathbf{V}$) are optimized alternatively until convergence is achieved. This is repeated for each layer independently. The resulting $\mathbf{U}$ and $\mathbf{V}$ form the fixed shared bases extracted from the checkpoint collection, while $\mathbf{\Sigma}_i$ provides the low-dimensional representation of checkpoint $\Delta\mathbf{W}_i$.

**Reconstruction Error.** We reconstruct the original LoRA adapters using the learned shared bases and measure the reconstruction error to quantify how effectively the shared bases represent adapters from different tasks. For adapter $i$, we compute the relative reconstruction error as

$$\mathcal{E}_i = \frac{\left\|\mathbf{B}_i\mathbf{A}_i - \mathbf{U}\mathbf{\Sigma}_i\mathbf{V}^\top\right\|_F^2}{\left\|\mathbf{B}_i\mathbf{A}_i\right\|_F^2}. \tag{13}$$

We report the average reconstruction error across all adapters and adapted layers in Figure 2. As shown in Figure 2a, the reconstruction error decreases as we increase the rank of the shared bases. For the common-sense reasoning (CSR) and MATH datasets, a relatively low rank of 32–64 achieves low reconstruction error. The adapters reconstructed using the shared bases retain performance similar to that of the original LoRA adapters as shown in Figure 2b, empirically supporting the existence of a shared low-dimensional subspace. We further show analytically that adapters trained on similar tasks can be represented in a shared low-dimensional subspace in section 4.

## 4 Mathematical Analysis under a Linear Model

In this section, we characterize the exact minimizer of equation 9 in closed form under a linear-feature model. for the data. We extend the setup from Tripuraneni et al. (2021) to vector-valued responses by writing the input-output pairs $\{(\mathbf{x}_{t,k}, \mathbf{y}_{t,k})\}_{k=1}^{n_t}$ for task $t = 1, \ldots, T$ as arising from the model:

$$\mathbf{y}_{t,k} = (\mathbf{S}\mathbf{R}_t)^\top \mathbf{x}_{t,k} + \epsilon_{t,k} \tag{14}$$

where $\mathbf{x}_{t,k} \in \mathbb{R}^{d_{\text{in}}}$, $\mathbf{y}_{t,k} \in \mathbb{R}^{d_{\text{out}}}$, and $\epsilon_{t,k}$ is independent Gaussian noise. The primary assumption is that the matrix $\mathbf{S} \in \mathbb{R}^{d_{\text{in}} \times r}$ of orthonormal columns is shared across tasks, while $\mathbf{R}_t \in \mathbb{R}^{r \times d_{\text{out}}}$ is a task-specific loading matrix.

By construction, we take the LoRA updates in equation 7 to be defined as $\Delta\mathbf{W}_i = \mathbf{S}\mathbf{R}_{t(i)}$ where $t(i)$ denotes the task associated with sample $i$, i.e., the updates are shared across samples within each task. As in equation 8 we can factorize the updates for each task as $\Delta\mathbf{W}_t = \mathbf{U}_t\mathbf{\Sigma}_t\mathbf{V}_t$ and define the task-averaged diagonal matrix of the update coefficients $\mathbf{M} := \sum_{t=1}^{T} \mathbf{\Sigma}_t\mathbf{\Sigma}_t^\top \in \mathbb{R}^{r \times r}$.

The following theorem shows that the closed-form solutions of equation 9 for the optimal shared bases are exactly rank-$r$. Moreover, the closed-form eigenspace solutions recover the bases defined across tasks. This holds provided that the matrix $M$ is full-rank.

**Theorem 1** (Closed-form optimal basis under the multivariate model is exactly low rank)**.** *For the data model in eq. (14), the Gram matrix*

$$\mathbf{G}_U := \sum_{t=1}^{T} \Delta\mathbf{W}_t \Delta\mathbf{W}_t^\top \in \mathbb{R}^{d_{\text{in}} \times d_{\text{in}}} \tag{15}$$

*satisfies* $\mathbf{G}_U = \mathbf{U}\mathbf{M}\mathbf{U}^\top$*, so that* $\operatorname{rank}(\mathbf{G}_U) \le r$ *in general, with equality* $\operatorname{rank}(\mathbf{G}_U) = r$ *when* $\mathbf{M} \succ 0$*. Moreover, the closed-form global optimum of eq. (9) is given by*

$$\operatorname{Eig}_r(\mathbf{G}_U) \;=\; \operatorname{range}(\mathbf{S}). \tag{16}$$

*Proof.* Letting $\Delta\mathbf{W}_t = \mathbf{S}\mathbf{R}_t = \mathbf{U}(\mathbf{\Sigma}_t\mathbf{V}_t)$ with $\mathbf{S} = \mathbf{U}$ and $\mathbf{R}_t = \mathbf{\Sigma}_t\mathbf{V}_t$ for every $t$ and substituting,

$$\mathbf{G}_U = \sum_{t=1}^{T} \mathbf{U}\mathbf{\Sigma}_t\mathbf{V}_t^\top\mathbf{V}_t\mathbf{\Sigma}_t^\top\mathbf{U}^\top = \mathbf{U}\Big(\sum_{t=1}^{T} \mathbf{\Sigma}_t\mathbf{\Sigma}_t^\top\Big)\mathbf{U}^\top = \mathbf{U}\mathbf{M}\mathbf{U}^\top, \tag{17}$$

using $\mathbf{V}_t^\top\mathbf{V}_t = I_r$ for each $t$. Since $U$ has orthonormal columns, $\mathbf{G}_U$ has the same nonzero eigenvalues as $M$, with eigenvectors $\mathbf{U}q_k$ for $q_k$ an eigenvector of $\mathbf{M}$. Hence $\operatorname{rank}(\mathbf{G}_U) = \operatorname{rank}(\mathbf{M}) \le r$, with equality when $\mathbf{M} \succ 0$. When $\mathbf{M} \succ 0$, $\mathbf{G}_U$ has exactly $r$ nonzero eigenvalues and $\operatorname{range}(\mathbf{G}_U) = \operatorname{range}(\mathbf{U})$, so $\operatorname{Eig}_r(\mathbf{G}_U) = \operatorname{range}(\mathbf{U})$. Noticing $\mathbf{U} = \mathbf{S}$ yields the final result. □

**Remark 1.** *Unlike* $\mathbf{G}_U$*, whose rank is bounded by* $r$ *regardless of the number of tasks* $T$*, the matrix* $\mathbf{G}_V = \sum_{t=1}^{T} \mathbf{R}_t^\top\mathbf{R}_t$ *has* $\operatorname{rank}(\mathbf{G}_V) \le \min\big(\sum_{t=1}^{T} \operatorname{rank}(\mathbf{R}_t),\, d_{\text{out}}\big)$*, which can grow with* $T$*. This asymmetry reflects that only the input-side basis* $\mathbf{S}$ *is assumed shared across tasks in equation 14.*

## 5 Methodology and Results

Once the shared bases are learned from trained LoRA adapters, we can generate new LoRA adapters within this subspace for new downstream tasks. In this representation, a task-specific adapter is approximated as

$$\mathbf{B}_i\mathbf{A}_i \approx \mathbf{U}\mathbf{\Sigma}_i\mathbf{V}^\top, \tag{18}$$

Table 1: **Results on the common-sense reasoning benchmark.** We compare our method, B2L, with trained LoRA adapters and the DnD baseline. For each held-out dataset, the LoRA adapters, DnD, and B2L are trained on the remaining datasets. B2L outperforms the baselines in terms of accuracy (%) across all unseen datasets.

| Method | ARC-c | ARC-e | BoolQ | HellaSwag | OBQA | PIQA | WinoGrande | Avg |
|---|---|---|---|---|---|---|---|---|
| Training LoRAs | 39.5 | 37.5 | 13.5 | 22.4 | 30.2 | 40.5 | 38.8 | 31.7 |
| DnD | 51.6 | 68.6 | 44.9 | 25.9 | 40.8 | 47.0 | 50.0 | 47.0 |
| **B2L (ours)** | **52.7±0.4** | **69.7±0.1** | **54.5±1.1** | **27.8±0.7** | **58.7±0.6** | **58.4±0.7** | **51.7±0.4** | **53.4** |

where $\mathbf{U} \in \mathbb{R}^{d_{\text{out}} \times r}$ and $\mathbf{V} \in \mathbb{R}^{d_{\text{in}} \times r}$ are fixed and shared across tasks, while $\mathbf{\Sigma}_i \in \mathbb{R}^{r \times r}$ contains the task-specific low-rank coordinates. To predict these coordinates for unseen tasks at test time, we train a hypernetwork $h_\phi$ to map a dataset representation $\mathbf{z}_i$ to $\widehat{\mathbf{\Sigma}}_i$ as shown in Figure 1. The predicted coordinates are then combined with the shared bases to generate the new LoRA adapter:

$$\widehat{\mathbf{\Sigma}}_i = h_\phi(\mathbf{z}_i), \qquad \widehat{\Delta \mathbf{W}}_i = \mathbf{U} \widehat{\mathbf{\Sigma}}_i V^\top. \tag{19}$$

This reduces the output dimension of the hypernetwork to $r^2$, or to $r$ when $\widehat{\mathbf{\Sigma}}_i$ is constrained to be diagonal. The shared bases capture the common structure across adapters, leaving the hypernetwork to model only task-specific variation within the learned subspace. Consequently, our parameterization substantially reduces the complexity of the prediction problem and makes the hypernetwork easier and more stable to train. Our method is orthogonal to existing approaches and can be easily integrated with existing baselines by replacing the prediction of the complete LoRA matrices or multiplied-out weight updates with the prediction of their low-rank coordinates in the learned shared basis.

**Hypernetwork Training.** Following DnD, we construct a hypernetwork training set by pairing a batch of prompts from a dataset with a randomly sampled LoRA checkpoint trained on the same dataset. The prompts are sampled randomly and encoded using a frozen pre-trained text encoder (e.g., Sentence-BERT (Reimers & Gurevych, 2019)) to obtain a dataset representation $\mathbf{z}_i$. Given this representation, the hypernetwork predicts the task-specific coordinates $\widehat{\Sigma}_i = h_\phi(\mathbf{z}_i)$. We combine these coordinates with the fixed shared bases to reconstruct the weight update, $\widehat{\Delta W}_i = \mathbf{U} \widehat{\mathbf{\Sigma}}_i \mathbf{V}^\top$. The hypernetwork is trained by minimizing the mean squared error between the reconstructed and original weight updates:

$$\mathcal{L}_{\text{MSE}}(\phi) = \frac{1}{N} \sum_{i=1}^{N} \left\| \mathbf{U} h_\phi(\mathbf{z}_i) V^\top - \mathbf{B}_i \mathbf{A}_i \right\|_F^2, \tag{20}$$

where $N$ is the number of training samples. Thus, our method retains DnD's prompt–checkpoint pairing and conditioning strategy while replacing the high-dimensional parameter-prediction objective with a compact, structured coordinate-prediction problem. We provide more details in the appendix section A.1.

### 5.1 Experimental Results

**Datasets and LoRA Zoo.** We use the LoRA zoo released by DnD (Liang et al., 2025) and adapt their hypernetwork training methodology to our shared-basis parameterization for a fair comparison. We use the LoRA adapters trained on the CSR and MATH datasets for the Qwen2.5 model family (Qwen et al., 2025). The common-sense zoo consists of Qwen2.5-0.5B LoRA adapters trained on seven datasets: ARC-Easy, ARC-Challenge (Clark et al., 2018), BoolQ (Clark et al., 2019), OpenBookQA (OBQA) (Mihaylov et al., 2018), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), and WinoGrande (Sakaguchi et al., 2021). The mathematical-reasoning zoo consists of Qwen2.5-1.5B LoRA adapters trained on six datasets: Competition-MATH, MathQA, Math-IIO-68K-Mini, Math-Plus, Mu-Math, and ToT-Math-V1. Each zoo contains multiple intermediate checkpoints collected throughout fine-tuning, providing a diverse set of adapters across datasets and training stages.

Table 2: **Results on the math reasoning benchmarks.** We compare our method with training LoRA adapters and DnD on MATH and GSM8K using Qwen2.5-1.5B-Instruct. **B2L** outperforms the baseline method on both the benchmarks.

| Method | MATH | GSM8K |
|---|---|---|
| Training LoRAs | 14.8 | 42.9 |
| DnD | 23.9 | 66.3 |
| **B2L (ours)** | **33.7±0.2** | **69.3±0.3** |

Table 3: **Cross-domain generalization to a science dataset.** We evaluate the hypernetwork trained on the CSR datasets. For each row, the indicated CSR dataset is excluded from training. $\Delta$ denotes the improvement of our method over DnD.

| Excluded CSR dataset | Base Model | Official DnD | Ours (Sigma-Net) | $\Delta$ |
|---|---|---|---|---|
| ARC-c | | 28.87% | **71.41%** | +42.54 |
| ARC-e | | 29.11% | **58.46%** | +29.35 |
| BoolQ | | 40.29% | **72.20%** | +31.91 |
| HellaSwag | 40.13% | 50.31% | **69.54%** | +19.23 |
| OBQA | | 46.04% | **70.37%** | +24.33 |
| PIQA | | 62.70% | **69.14%** | +6.44 |
| WinoGrande | | 61.42% | **69.06%** | +7.64 |
| **Average** | *(40.13%)* | 45.53% | **68.60%** | **+23.06** |

**Common-Sense Reasoning (CSR) Benchmark.** We benchmark our method on the CSR benchmark, which contains multiple-choice and binary-choice questions across seven datasets: ARC-Easy and ARC-Challenge (Clark et al., 2018), BoolQ (Clark et al., 2019), OpenBookQA (OBQA) (Mihaylov et al., 2018), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), and WinoGrande (Sakaguchi et al., 2021). Following a leave-one-out setting, we train the hypernetwork and learn the shared basis using LoRA adapters from six datasets, then evaluate the generated adapter on the held-out dataset. As shown in Table 1, our method consistently outperforms DnD across all seven held-out datasets. We also compare against a LoRA adapter trained on the remaining datasets; both DnD and our method outperform this baseline, showing that WSL can generate task-relevant adapters for unseen datasets.

The difference between the generated adapters is particularly evident on OBQA, where DnD achieves only $40.8\%$ and fails to follow the required output instructions in many cases (section A.2). In contrast, the adapter generated by B2L follows the instructions for all prompts, as the base model does. One possible explanation is that B2L constrains the generated adapter to the subspace spanned by shared bases learned from trained LoRA adapters, which may help it retain instruction-following capabilities.

**Math Benchmark.** We also evaluate our method on the GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) benchmarks to assess its reasoning capabilities beyond multiple-choice questions. For the MATH benchmark, we train the hypernetwork using adapters from Math-Plus and Math-IIO-68K-Mini. As shown in table 2, our method outperforms both the base model and the DnD baseline on both benchmarks.

**Cross-Domain Generalization.** Our method constrains each generated adapter to lie within the shared subspace learned from source-domain adapters. We hypothesize that this constraint improves robustness on unseen datasets and prompts. To evaluate this hypothesis, we train the hypernetworks for DnD and our method on the CSR datasets and evaluate them on an unseen science dataset using Qwen2.5-0.5B-Instruct model. As shown in Table 3, our method outperforms both the base model and the DnD baseline by a significant margin.

**Ablation.** To further understand whether the trained hypernetwork learns a meaningful mapping from the task representation to task-specific coordinates, we compare B2L with RandOpt/Neural

Thickets (Gan & Isola, 2026). We use RandOpt to randomly sample candidate coordinates in the learned shared basis and select them based on their performance on a small validation set. This allows us to test whether the hypernetwork learns a useful mapping beyond what can be achieved by randomly sampling coordinates in the same basis. As shown in Table 4, B2L achieves higher accuracy on the CSR datasets, validating that the hypernetwork learns a useful mapping from task representations to task-specific coordinates.

Table 4: **Comparison with RandOpt**. B2L outperforms randomly sampled adapters (RandOpt) within the same shared bases on most tasks, showing that the hypernetwork learns a meaningful mapping from task representations to task-specific coordinates.

| Method | ARC-c | ARC-e | BoolQ | HellaSwag | OBQA | PIQA | WinoGrande | Avgs |
|---|---|---|---|---|---|---|---|---|
| Base model | 37.2 | 54.3 | 60.9 | 31.0 | 42.6 | 30.1 | 29.4 | 40.8 |
| RandOpt (Neural Thickets) | 39.7 | 53.0 | **61.1** | **30.1** | 43.6 | 31.3 | 49.6 | 44.1 |
| **B2L (ours)** | **52.7** | **69.7** | 54.5 | 27.8 | **58.7** | **58.4** | **51.7** | **53.4** |

## 6 Conclusion and Limitations

With the growing availability of open-source models and datasets, WSL offers a promising way to leverage the information contained in model zoos for efficient adaptation to downstream tasks. In this work, we exploit the shared subspaces in which LoRA checkpoints trained on related tasks approximately lie and train a hypernetwork to predict task-specific LoRA weights directly from dataset prompts. By predicting only the coordinates within fixed shared bases, our formulation substantially reduces the hypernetwork's output dimensionality while avoiding the non-identifiability arising from the $\mathrm{GL}(r)$ symmetry of the LoRA factors. Our results demonstrate that incorporating the shared geometry of a model zoo provides an effective inductive bias for generating adapters that generalize to previously unseen tasks and domains.

Although WSL has the potential to transform how models are adapted and fine-tuned, current methods remain largely restricted to controlled settings. Even when trained on related tasks, adapters can learn substantially different weights because of weight-space symmetries and variations in initialization, optimization, hyperparameters, and training trajectories. Existing methods commonly mitigate this difficulty by constructing model zoos using a fixed base model and standardized training configurations. Our approach addresses the ambiguity arising from LoRA factorization by operating on invariant weight updates, but it still assumes that the adapters share compatible architectures and sufficiently aligned task structure. Extending WSL to heterogeneous model zoos containing different base models, adapter configurations, optimizers, and training procedures remains an important direction for future work.

### AI use statement

In this work, we used generative AI tools to edit a research paper to improve readability, propose a title or keywords for a research paper, implement methods and create or edit software code. We have **not** used generative AI tools to generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, interpret results. We have reviewed all AI-assisted work: original meaning of the text was not changed when AI tools were used for editing some part of the manuscript; we verified the code to the best of our abilities. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

## References

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th Annual Meeting of the*

*Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, Online, 2021. Association for Computational Linguistics.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI*, pp. 7432–7439, 2020.

Rujikorn Charakorn, Edoardo Cetin, Yujin Tang, and Robert Tjarko Lange. Text-to-loRA: Instant transformer adaption. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=zWskCdu3QA.

Rujikorn Charakorn, Edoardo Cetin, Shinnosuke Uesaka, and Robert Tjarko Lange. Doc-to-loRA: Learning to instantly internalize contexts. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=iW1oBBO72S.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Marvin F. da Silva, Mohammed Adnan, Felix Dangel, and Sageev Oore. Generalizing the geometry of model merging through frechet averages, 2026. URL https://arxiv.org/abs/2604.27155.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=OUIFPHEgJU.

Rickard Brüel Gabrielsson, Jiacheng Zhu, Onkar Bhardwaj, Leshem Choshen, Kristjan Greenewald, Mikhail Yurochkin, and Justin Solomon. Compress then serve: Serving thousands of loRA adapters with little overhead. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=3XMA8RDJu2.

Yulu Gan and Phillip Isola. Neural thickets: Diverse task experts are dense around pretrained weights. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=92oF5bU4cU.

David Ha, Andrew M. Dai, and Quoc V. Le. Hypernetworks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=rkpACe1lx.

Xiaolong Han, Zehong Wang, Bo Zhao, Binchi Zhang, Jundong Li, Damian Borth, Rose Yu, Haggai Maron, Yanfang Ye, Lu Yin, and Ferrante Neri. A survey of weight space learning: Understanding, representation, and generation. *arXiv preprint arXiv:2603.10090*, 2026. URL https://arxiv.org/abs/2603.10090.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

Eliahu Horwitz, Nitzan Kurer, Jonathan Kahana, Liel Amar, and Yedid Hoshen. We should chart an atlas of all the world's models. In *The Thirty-Ninth Annual Conference on Neural Information Processing Systems Position Paper Track*, 2025. URL https://openreview.net/forum?id=BzFMBNqg7R.

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. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Prakhar Kaushik, Shravan Chaudhari, Ankit Vaidya, Rama Chellappa, and Alan Yuille. The universal weight subspace hypothesis. *arXiv preprint arXiv:2512.05117*, 2025. URL https://arxiv.org/abs/2512.05117.

Boris Knyazev, Michal Drozdzal, Graham W. Taylor, and Adriana Romero. Parameter prediction for unseen deep architectures. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=vqHak8NLk25.

Zhiyuan Liang, Dongwen Tang, Yuhao Zhou, Xuanlei Zhao, Mingjia Shi, Wangbo Zhao, Zekai Li, Peihao Wang, Konstantin Schürholt, Damian Borth, Michael M. Bronstein, Yang You, Zhangyang Wang, and Kai Wang. Drag-and-drop LLMs: Zero-shot prompt-to-weights. 2025. URL https://openreview.net/forum?id=fTkBZLxBzV.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1260. URL https://aclanthology.org/D18-1260/.

Jason Phang, Yi Mao, Pengcheng He, and Weizhu Chen. HyperTuning: Toward adapting large language models without back-propagation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 27854–27875. PMLR, 2023.

Theo Putterman, Derek Lim, Yoav Gelberg, Stefanie Jegelka, and Haggai Maron. Learning on loRAs: GL-equivariant processing of low-rank weight spaces for large finetuned models. In *Workshop on Neural Network Weights as a New Data Modality*, 2025. URL https://openreview.net/forum?id=MZmPapNtih.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *ArXiv*, abs/1908.10084, 2019. URL https://api.semanticscholar.org/CorpusID:201646309.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Konstantin Schürholt, Boris Knyazev, Xavier Giró-i Nieto, and Damian Borth. Hyper-representations as generative models: Sampling unseen neural network weights. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://arxiv.org/abs/2209.14733.

Nilesh Tripuraneni, Chi Jin, and Michael Jordan. Provable meta-learning of linear representations. In *International conference on machine learning*, pp. 10434–10443. Pmlr, 2021.

Kai Wang, Dongwen Tang, Wangbo Zhao, Konstantin Schürholt, Zhangyang Wang, and Yang You. Scaling up parameter generation: A recurrent diffusion approach. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=65llKR17s4.

Zhuoqian Yang, Mathieu Salzmann, and Sabine Süsstrunk. Weight space representation learning via neural field adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17734–17743, June 2026.

Qinyuan Ye and Xiang Ren. Learning to generate task-specific adapters from task description. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pp. 646–653, Online, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-short.82. URL https://aclanthology.org/2021.acl-short.82/.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

# A Appendix

## A.1 Experimental Setup and Hyperparameters

**Base model and benchmarks.** We use `Qwen2.5-0.5B-Instruct` as the base model for CSR benchmark and Qwen2.5-1.5B for the MATH benchmark. LoRA adapters are applied to all attention and MLP projections (`q`, `k`, `v`, `o`, `gate`, `up`, and `down`), resulting in 168 adapted matrices. We evaluate on seven common-sense reasoning benchmarks following DnD: ARC-c, ARC-e, BoolQ, HellaSwag, OBQA, PIQA, and WinoGrande. All experiments follow a leave-one-dataset-out protocol: for each held-out dataset, the shared bases and meta-network are trained using adapters and prompts from the remaining six datasets, and evaluated zero-shot on the held-out dataset.

**LoRA adapter zoo and DnD baseline.** We use the trained LoRA adapter zoo released by Liang et al. (2025) and do not retrain these adapters. The adapters use rank $r = 8$ and $\alpha = 16$ and target all seven attention and MLP projections. We similarly evaluate DnD using the authors' released checkpoints and generation pipeline without retraining.

**Shared basis construction.** For each leave-one-dataset-out split, we learn a shared basis independently for each adapted layer from the training adapter zoo using JD-Full. We use a target reconstruction NMSE of 0.10, a convergence tolerance of $10^{-3}$, and a maximum of 50 iterations. The resulting shared-basis ranks vary across layers.

**B2L training.** Our meta-network uses the `HyperConvDecoder` architecture from DnD to predict coordinates in the learned shared bases. Task prompts are encoded using a frozen `all-MiniLM-L12-v2` encoder. We train a separate meta-network for each leave-one-dataset-out split using only adapters and prompts from the remaining six datasets. The main training hyperparameters are summarized in Table 5.

Table 5: Training hyperparameters for B2L.

| Hyperparameter | Value |
|---|---|
| Condition encoder | `all-MiniLM-L12-v2` (frozen) |
| Condition texts per adapter | 128 |
| Condition-embedding noise | Gaussian, $\sigma = 0.02$ |
| Loss | Masked MSE |
| Optimizer | AdamW |
| Learning rate | $3 \times 10^{-5}$ |
| Weight decay | 0.05 |
| Gradient clipping | 1.0 |
| Batch size | 64 |
| Training steps | 2000 |

## A.2 Shared Bases Preserve Instruction-Following Capability

In Table 1, we observed that B2L outperforms the baseline for all the seven datasets, in particular for the OBQA. To further understand how our method improves over the baseline, we examined the model output for the OBQA dataset and found that the accuracy gap has a clear behavioral cause: DnD generated adapter causes the merged model to *stop following the task's answer format*, while our shared-basis reconstruction does not.

Table 6 presents representative examples from the 46 held-out OBQA questions for which the real adapter and all 10 generations of B2L produce the correct answer, while the base model and all 10 generations of the official DnD checkpoint are incorrect. All models receive the same prompt and are instructed to respond with one of `[A]`, `[B]`, `[C]`, or `[D]`. While the real adapter and B2L consistently follow this format, the official DnD generations produce free-form responses rather than the required answer labels.

Table 6: **Qualitative examples on OBQA.** Representative held-out questions for which the real adapter and all 10 generations of B2L produce the correct answer, while the base model and all 10 generations of DnD are incorrect and fails to follow the instruction.

| Question | Real zoo | B2L | Base | DnD |
|---|---|---|---|---|
| What are iron nails made out of? A: wood B: plastic **C: metal** D: glass | `[C]` | `[C]` | `D: glass.` | "Iron nails are made out of wood. The answer is D: glass. Iron nails are commonly made from soft, brittle wood…" |
| Which type of organism urinates? A: airplane B: car **C: mammal** D: boat | `[C]` | `[C]` | `D: boat.` | "The urination process is actually the True process urinates. The answer is True: airplane.…" |
| In general, how many times per month is there a full moon? A: twice B: three times **C: once** D: four times | `[C]` | `[C]` | `D: four times.` | "True, the full moon occurs twice per month. Therefore, the answer is In general, there is a full moon once per…" |
| Which of these is not required for photosynthesis? A: sunlight **B: oxygen** C: water D: carbon dioxide | `[B]` | `[B]` | `D: carbon dioxide.` | "True, sunlight is not required for photosynthesis. Photosynthesis is a process that occurs in plants, algae…" |

### A.3 Additional Ablation

For the MATH benchmark, we observed that further tuning the merge-time delta scale $\alpha$ can improve the performance. To find the best the value of $\alpha$ we run a sweep over different $\alpha$ values as reported in Table 7

Table 7: Effect of the merge-time delta scale $\alpha$ (native value $\alpha$=2.0 $= \mathrm{lora_alpha}/r$) on MATH and GSM8K accuracy.

| $\alpha$ (delta scale) | MATH | GSM8K |
|---|---|---|
| 0.5 | **33.7** | 65.1 |
| 1.0 | 33.5 | 66.9 |
| 1.5 | 31.8 | **69.5** |
| 2.0 (native) | 30.2 | 69.3 |