WHAT SHOULD FEDERATED LORA SHARE? FEDSAIL VIA INPUT-AWARE SUBSPACE ALIGNMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated low-rank adaptation (LoRA) requires identifying an update structure that is shared across heterogeneous clients. Prior work reports strong similarity among trained LoRA projection matrices across clients; however, such agreement may be largely induced by common initialization and collapses toward random overlap under independent initialization. More crucially, relying solely on parameter similarity inherently ignores the influence of local input regime. To uncover a more robust shared structure, we introduce an input-aware action matrix that weights the adapter update by the second-moment statistics of local layer inputs. Empirically, while parameter similarity vanishes, the leading right singular directions of this action matrix remain strongly aligned across clients. This shared geometry preserves task-conditioned differences and naturally varies across network depths. Motivated by these findings, we propose Federated Subspace-Guided Action-Informed Learning (FedSAIL). Instead of averaging weights, FedSAIL estimates a shared action subspace to regularize local training while preserving client-specific coefficients. Across several benchmarks, our approach consistently improves predictive performance over competing federated LoRA methods while reducing communication cost significantly.

1 INTRODUCTION

Low-rank adaptation (LoRA) fine-tunes a frozen pretrained layer through a compact update $\mathbf{\Delta} = \mathbf{BA}$ (Hu et al., 2022). In federated learning, clients can exchange such adapters while keeping their data local (McMahan et al., 2017; Zhang et al., 2024). A central question is therefore what information should be shared across clients. Existing methods address this by specifying which components of the adapter are communicated and how they are aggregated. A representative example is FedSA-LoRA, which trains both factors but shares only $\mathbf{A}$, while retaining client-specific $\mathbf{B}_k$ for personalization (Guo et al., 2025). For illustration, Figure 1 (b) depicts the FedSA-LoRA mechanism, with the shared LoRA component highlighted in yellow. Other approaches share $\mathbf{B}$ against either a common frozen $\mathbf{A}$ (Sun et al., 2024) or locally trained $\mathbf{A}_k$ (Ban & Ji, 2026). FLoRA aggregates full adapter updates by stacking factors (Wang et al., 2024), while LoRA-FAIR refines aggregation and next-round initialization (Bian et al., 2025). FedRot-LoRA instead aligns factor coordinates through orthogonal rotations before aggregation, preserving each adapter product while addressing rotational misalignment (Zhang et al., 2026). These choices determine how clients represent and combine their updates, but parameter agreement alone need not identify structure learned in common.

Initialization provides one reason for this distinction. FedSA-LoRA motivates sharing $\mathbf{A}$ by its higher cross-client similarity relative to $\mathbf{B}$ (Guo et al., 2025). However, recent analyses suggest that this apparent similarity may be largely inherited from common initialization (Zhu et al., 2024; Ban & Ji, 2026). Our controlled RoBERTa/RTE (Liu et al., 2019; Wang et al., 2018) experiment further supports this interpretation: the rowspace agreement of the final $\mathbf{A}$ falls to near-random levels under independent initialization, while the learned changes remain only weakly aligned under either initialization scheme.

A further limitation of sharing only LoRA parameters is that it does not explicitly account for client-specific input distributions. Since each client induces a local input regime at every adapted layer, an update's effect depends jointly on its weights and the hidden states it encounters. Differences in domain, prompts, or sequence length can therefore make even identical adapters behave differently

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.

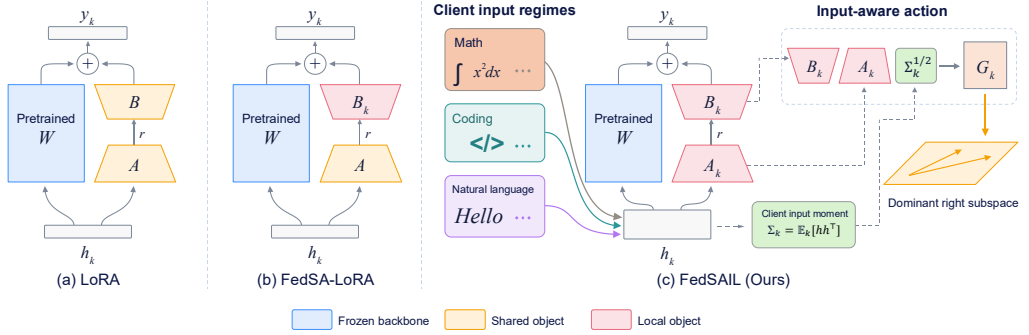


Figure 1: Sharing in federated LoRA. (a) LoRA. (b) FedSA-LoRA shares $\mathbf{A}$ and retains $\mathbf{B}_k$ locally. (c) FedSAIL shares a spectral summary of the dominant right subspace of $\mathbf{G}_k = \mathbf{B}_k\mathbf{A}_k\mathbf{\Sigma}_k^{1/2}$.

across clients. These local input regimes are reflected in the hidden-state activations, motivating the use of activation statistics, such as second moments, to summarize their local geometry. Such statistics have already been used for adapter construction: EVA initializes LoRA input factors from dominant activation directions and allocates rank according to explained activation variance (Paischer et al., 2025), while CorDA uses activation covariance to orient decompositions of pretrained weights (Yang et al., 2024a). While these methods use activation statistics to construct adapters, we instead ask whether such statistics can reveal shared structure in learned updates across client-specific input regimes, motivating a shift beyond raw parameter sharing.

To address this question, we introduce the input-aware action matrix $\mathbf{G}_k = \mathbf{B}_k\mathbf{A}_k\mathbf{\Sigma}_k^{1/2}$, where $\mathbf{\Sigma}_k = \mathbb{E}_k\left[\mathbf{h}\mathbf{h}^\top\right]$ is the uncentered second moment of the local layer inputs. We defer its formal properties to Section 2.2, but intuitively, the leading right singular directions of $\mathbf{G}_k$ identify the input subspace where the adapter's effect is most concentrated. This provides a natural and robust representation of cross-client common structure. Figure 1 (c) contrasts this action-based perspective with existing parameter-sharing approaches.

Our empirical studies confirm that $\mathbf{G}_k$ uncovers a robust shared structure often obscured by raw parameter comparisons. In our RoBERTa/RTE diagnostic, the leading action direction remains aligned across clients even under independent initialization and captures substantially more action energy on a held-out client than matched geometric controls. Furthermore, a factorial GPT-2 study reveals that this geometric alignment is highly task-conditioned and depth-dependent: clients solving the same task remain more aligned across domains than clients solving different tasks. These results suggest a simple design principle: share only a compact common action subspace, retain client-specific flexibility, and use layerwise agreement to decide where collaboration is beneficial.

We realize this principle through *Federated Subspace-Guided Action-Informed Learning* (FedSAIL). Instead of exchanging raw weights, each client uploads a compact spectral summary of its action matrix. The server then estimates a common geometric basis, which guides local training via a soft penalty on actions falling outside this shared subspace. Since both LoRA factors remain local, clients preserve their specific coefficients and trainable residual directions. Crucially, this design fundamentally decouples communication cost from local adapter capacity, allowing clients to scale up their models locally without incurring additional communication costs. Furthermore, by evaluating cross-client agreement after a local warmup phase, FedSAIL can restrict collaboration only to structurally aligned layers. Evaluated on personalized GLUE, FedSAIL consistently outperforms existing federated LoRA baselines. In highly heterogeneous domain-and-task settings, our layer selection strategy further reduces communication by 19.26% while preserving comparable predictive accuracy, validating FedSAIL as a highly efficient and data-driven approach to federated fine-tuning.

We summarize our main contributions as follows:

- We introduce an input-aware action matrix $\mathbf{G}_k$ that exposes shared structure in federated LoRA beyond raw parameter similarity. Controlled diagnostics show that its leading

directions remain aligned even under independent initialization, capturing structures that are inherently task-conditioned and depth-dependent.

- We develop FedSAIL, a novel federated fine-tuning framework that guides personalized adaptation via a shared action subspace. By keeping both LoRA factors strictly local and exchanging only compact spectral summaries, FedSAIL fundamentally decouples network communication overhead from local adapter capacity.
- We evaluate FedSAIL across heterogeneous client settings, including ablations over local rank and communicated summaries. Selective layer sharing further reduces communication while maintaining comparable mean accuracy in the studied settings.

2 IDENTIFYING SHARED STRUCTURE IN FEDERATED LORA

In this paper, we consider a frozen linear layer with input dimension p and output dimension q. During local adaptation, client k learns a rank-r weight update parameterized as

$$\mathbf{\Delta}_k = \mathbf{B}_k \mathbf{A}_k \in \mathbb{R}^{q \times p}, \quad r < \min(p, q), \tag{1}$$

where $\mathbf{A}_k \in \mathbb{R}^{r \times p}$ and $\mathbf{B}_k \in \mathbb{R}^{q \times r}$ denote the down-projection and up-projection matrices, respectively. For notational simplicity, we omit the layer index and the LoRA scaling factor.

Given a layer input $\mathbf{h} \in \mathbb{R}^p$, the corresponding LoRA correction is $\mathbf{B}_k\big(\mathbf{A}_k \mathbf{h}\big)$ (Hu et al., 2022). This operation can be viewed as sequential feature-extraction and output-mapping: $\mathbf{A}_k$ projects $\mathbf{h}$ into an r-dimensional latent feature space, while $\mathbf{B}_k$ maps the resulting features to a correction vector in $\mathbb{R}^q$ (Zhu et al., 2024). A fundamental question in federated LoRA then arises: what underlying structure in this adaptation process is shared across clients, and how can it be used to improve local adaptation?

2.1 LIMITS OF RAW PARAMETER SIMILARITY

To answer this question, we first examine the pitfalls of existing parameter-centric approaches. In federated learning, sharing a LoRA component typically means aggregating locally trained parameters across clients and broadcasting the result for the next round of local training. Existing methods share either the projection matrix $\mathbf{A}_k$, $\mathbf{B}_k$, or the full adapter update $\mathbf{\Delta}_k$, each reflecting a different assumption about transferable structure. For example, sharing $\mathbf{A}_k$ imposes a common input feature map while allowing client-specific $\mathbf{B}_k$ to produce personalized output corrections. Conversely, sharing $\mathbf{B}_k$ imposes a common mapping from latent representation to the output space while retaining client-specific input projections (Sun et al., 2024). As shown in Table 1, we summarize some shared structures and the components retained locally across existing methods.

Table 1: Comparison of shared components, aggregation strategies, and local training rules across different federated LoRA frameworks.

Method	**Shared object**	**Aggregation**	**Local training**
FedIT (Zhang et al., 2024)	$\mathbf{A}, \mathbf{B}$	$\bar{\mathbf{A}} = \sum_k \omega_k \mathbf{A}_k$ $\bar{\mathbf{B}} = \sum_k \omega_k \mathbf{B}_k$	$\mathbf{A}_k, \mathbf{B}_k$
FLoRA (Wang et al., 2024)	Adapter update $\mathbf{\Delta}$	$\bar{\mathbf{\Delta}} = \sum_k \omega_k \mathbf{\Delta}_k$	$\mathbf{A}_k, \mathbf{B}_k$
FFA-LoRA (Sun et al., 2024)	$\mathbf{B}$; common frozen $\mathbf{A}^{(0)}$	$\bar{\mathbf{B}} = \sum_k \omega_k \mathbf{B}_k$	Only $\mathbf{B}_k$
FedSA-LoRA (Guo et al., 2025)	$\mathbf{A}$	$\bar{\mathbf{A}} = \sum_k \omega_k \mathbf{A}_k$	$\mathbf{A}_k, \mathbf{B}_k$
FedDPA-LoRA (Yang et al., 2024b)	Global $\mathbf{A}^g, \mathbf{B}^g$	$\bar{\mathbf{A}}^g = \frac{1}{K}\sum_k \mathbf{A}_k^g$ $\bar{\mathbf{B}}^g = \frac{1}{K}\sum_k \mathbf{B}_k^g$	Global $\mathbf{A}_k^g, \mathbf{B}_k^g$ Local $\mathbf{A}_k^\ell, \mathbf{B}_k^\ell$
FedSAIL (ours)	Dominant right subspace of $\mathbf{G}_k$ in (3)	$\widehat{\mathbf{M}} = \sum_k \omega_k \widehat{\mathbf{M}}_k$ in (6) $\widehat{\mathbf{V}} = \mathrm{TopEig}_s(\widehat{\mathbf{M}})$	$\mathbf{A}_k, \mathbf{B}_k$

However, these parameter-sharing schemes do not by themselves reveal what structure is genuinely learned in common across clients. As a representative example, Guo et al. (2025) motivate

sharing $\mathbf{A}_k$ based on its higher cross-client similarity relative to $\mathbf{B}_k$. Yet, the factors in low-rank matrix factorizations are nonunique: $\mathbf{B}_k\mathbf{A}_k = \big(\mathbf{B}_k\mathbf{R}\big)\big(\mathbf{R}^{-1}\mathbf{A}_k\big)$ for any invertible $\mathbf{R} \in \mathbb{R}^{r\times r}$. Consequently, directly sharing raw factor parameters constrains specific coordinate representations rather than coordinate-invariant subspaces. Furthermore, prior studies suggest that high similarity among trained $\mathbf{A}$ factors can arise from common initialization (Zhu et al., 2024; Ban & Ji, 2026), which highlights the need to distinguish inherited similarity from genuine learned agreement.

As an illustration, we compare common and independent initializations in a controlled RoBERTa-base study on RTE. We examine both the trained factors and their learned changes to assess whether clients develop a common representation during early adaptation. To rigorously quantify this cross-client geometric alignment, we employ the basis-invariant *Projector Dice*. For two orthogonal projectors $\mathbf{P}_1$ and $\mathbf{P}_2$ of ranks r_1 and r_2, it is defined as:

$$\mathrm{Dice}(\mathbf{P}_1, \mathbf{P}_2) := \frac{2\,\mathrm{tr}(\mathbf{P}_1\mathbf{P}_2)}{r_1 + r_2}, \tag{2}$$

which yields a value of one for perfectly identical subspaces and zero for completely orthogonal ones. Evaluated under this metric, our comparison reveals that high trained-$\mathbf{A}$ similarity need not indicate genuinely shared learning. Figure 2 illustrates this pattern across layers and heterogeneity levels. In both conditions, the final row spaces remain close to their respective initializations, while the learned changes are weakly aligned across clients. Thus, within this diagnostic, the apparent agreement primarily reflects inherited initialization geometry and is insufficient evidence of a common representation learned through adaptation. We refer to Appendix B for the detailed experimental setting and numerical results.

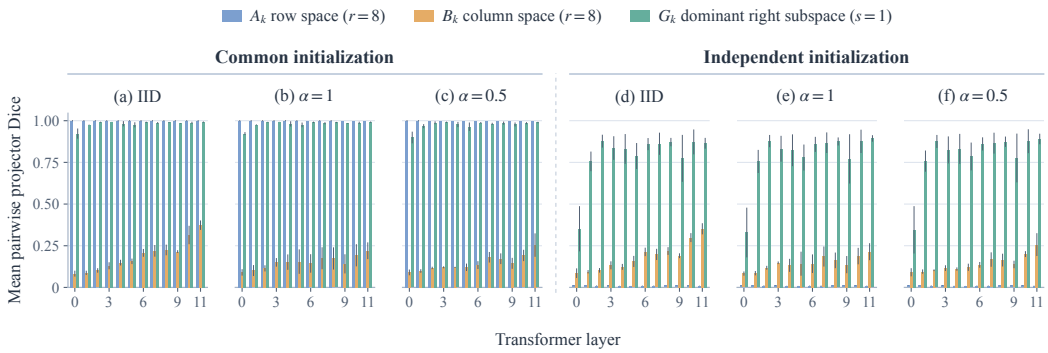


Figure 2: Cross-client subspace agreement for RoBERTa-base on RTE. For each initialization group, label heterogeneity increases from left to right, with $\alpha \in \{0.5, 1.0\}$ controlling Dirichlet label skew.

2.2 Input-Aware Action Matrix

If raw parameter similarity is an unreliable proxy for shared learning, where does the true cross-client shared component lie? We argue that the effect of a local update depends jointly on its weights and the hidden states it encounters. To characterize this, let $\mathbf{h} \in \mathbb{R}^p$ denote the hidden-state vector entering the adapted layer on client k, and let $\mathbb{E}_k$ denote the expectation under its local input distribution. Assuming $\mathbb{E}_k|\mathbf{h}|_2^2 < \infty$, define $\mathbf{\Sigma}_k = \mathbb{E}_k\big[\mathbf{h}\mathbf{h}^\top\big]$. Empirical version of $\mathbf{\Sigma}_k$ has been used to guide model merging (Jin et al., 2022; Salami et al., 2025), weight pruning (Frantar & Alistarh, 2023), and model compression (Wang et al., 2025), revealing its potential for shaping parameter effects. Here, we define the input-aware action matrix

$$\mathbf{G}_k := \mathbf{\Delta}_k\mathbf{\Sigma}_k^{1/2}. \tag{3}$$

Note that when $\mathbf{\Sigma}_k = \mathbf{I}_p$, (3) reduces to $\mathbf{G}_k = \mathbf{\Delta}_k$. More generally, if $\mathbf{\Sigma}_k = c_k\mathbf{I}_p$ for some $c_k > 0$, the right singular subspaces of $\mathbf{\Delta}_k$ are preserved. In contrast, anisotropic input moments can alter both the orientation and relative strength of these singular directions, revealing shared structure that may be hidden in the raw updates.Crucially, this representation evaluates update differences strictly through their effects on the local input distribution. For any alternative update $\widetilde{\mathbf{\Delta}}$, we have:

$$\big\|(\mathbf{\Delta}_k - \widetilde{\mathbf{\Delta}})\mathbf{\Sigma}_k^{1/2}\big\|_F^2 = \mathbb{E}_k\big[\big\|(\mathbf{\Delta}_k - \widetilde{\mathbf{\Delta}})\mathbf{h}\big\|_2^2\big].$$

Thus, update differences are weighted by the local input distribution, where directions carrying little input energy are naturally discounted. Moreover, since $\mathbf{G}_k$ depends on the LoRA factors only through their product $\mathbf{B}_k\mathbf{A}_k$, it is invariant to any invertible reparameterization of the raw factors.

Rather than synchronizing the entire action matrix, we can isolate its most critical geometric components. Let $\mathbf{V}_{k,s}$ contain the leading s right singular vectors of $\mathbf{G}_k$, and define $\mathbf{P}_{k,s} = \mathbf{V}_{k,s}\mathbf{V}_{k,s}^\top$. Among rank-$s$ orthogonal projectors, $\mathbf{P}_{k,s}$ maximizes $\|\mathbf{G}_k\mathbf{P}\|_F^2$, thereby identifying the low-dimensional subspace where the adapter's effect is most concentrated. This projector naturally provides a basis-invariant target for cross-client alignment. By employing guidance toward this subspace, clients are able to preserve their personalized coefficients and residual directions without the constraints of rigid parameter averaging. To demonstrate how this action-based approach differs from traditional parameter-based views, the following remark shows that the nearly orthogonal LoRA rowspaces can yield an identical right action subspace after accounting for client-specific input moments. Besides, in Appendix A, we provide an intuitive geometric illustration.

Remark 1 (Shared action geometry allows misaligned LoRA rowspaces)**.** *Mathematically, nearly orthogonal LoRA rowspaces can induce identical right action subspaces after accounting for client-specific input moments. As a simple rank-one example, let* $\mathbf{B}_1 = \mathbf{B}_2 = 1$, $\mathbf{A}_1 = \big(a^{-1}, 1\big)$ *and* $\mathbf{A}_2 = \big(1, a^{-1}\big)$*, with* $\mathrm{diag}\big(a, 1\big)$ *and* $\mathrm{diag}\big(1, a\big)$ *for* $a > 1$*. Then* $\mathbf{G}_1 = \mathbf{G}_2 = \big(1, 1\big)$*, while the Dice similarity between the raw LoRA rowspaces converges to zero as* $a \to \infty$*. Appendix A provides an intuitive illustration.*

Beyond this illustrative construction, our empirical diagnostic with rank-8 LoRA adapters also reveals that cross-client agreement is concentrated in the dominant rank-1 right action subspace, while the remaining directions exhibit substantially weaker alignment. This observation motivates sharing a compact geometric subspace while retaining greater local capacity for personalization. Detailed experimental evidence is provided in Appendix B.3.

2.3 Task and Layer Dependence of Shared Structure

Beyond robustness to initialization, a meaningful shared component should also reflect the actual learning objectives of the clients, e.g., clients solving the same task should exhibit stronger geometric alignment than those solving different tasks. Furthermore, this alignment should naturally vary across the network's depth, as deeper layers typically specialize in task-specific representations.

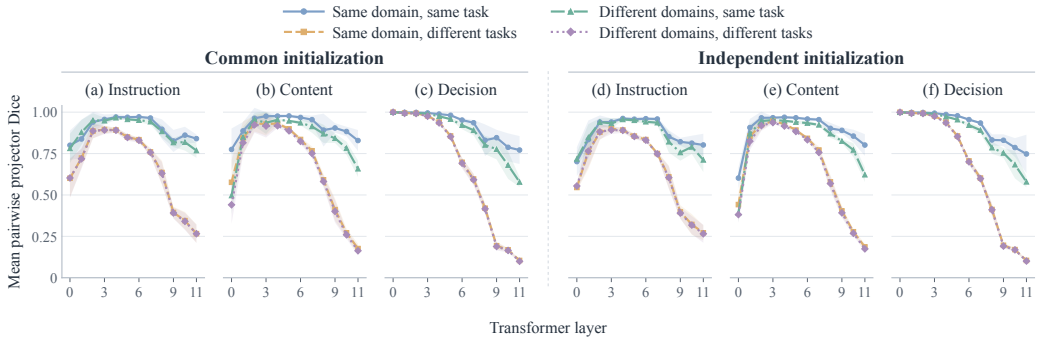


Figure 3: Task and layer dependence of action-subspace agreement in GPT-2. Curves report rank-1 Projector Dice averaged over query and value projections, while shaded bands indicate 95% confidence intervals.

To verify this, we evaluate the action-subspace agreement in a controlled GPT-2 study, crossing two input domains (movie vs. product reviews) with two distinct classification objectives (sentiment vs. tense). As shown in Figure 3, under both initialization schemes, same-task clients across domains exhibit stronger dominant-direction agreement on average than different-task clients within the same domain. Besides, this distinction varies with depth: early Decision layers remain highly aligned across tasks, whereas deeper Content and Decision layers show substantially lower agreement between different-task clients. These observations complement our initialization analysis in Section 2.1. Detailed diagnostics are provided in Appendix B.4.

3 FEDSAIL

In Section 2, we demonstrate that parameter similarity alone is an unreliable proxy for client agreement, whereas the action geometry retains a robust and task-conditioned structure. Motivated by these, we propose *Federated Subspace-Guided Action-Informed Learning* (FedSAIL). Instead of directly averaging adapter weights, FedSAIL extracts and shares the dominant right subspace of the input-aware action matrix. This allows the server to compute a basis-invariant geometric consensus to regularize local training, while keeping both LoRA factors and client-specific residual directions strictly local.

3.1 LOCAL ACTION SUBSPACE EXTRACTION

To capture input-conditioned action geometry while keeping raw data local, in our approach, each client k estimates the uncentered second moment of its layer-input hidden states. After that, we apply shrinkage toward a scaled identity matrix to improve the conditioning of this estimation and stabilize subsequent spectral computations:

$$\mathbf{S}_k = \frac{1}{n_k}\sum_{i=1}^{n_k}\mathbf{h}_{k,i}\mathbf{h}_{k,i}^\top, \qquad \widehat{\mathbf{\Sigma}}_k = (1-\gamma)\mathbf{S}_k + \gamma\frac{\mathrm{tr}(\mathbf{S}_k)}{p}\mathbf{I}_p. \tag{4}$$

where n_k is the number of local layer-input hidden states and $\gamma \in (0,1)$ determines the shrinkage level. Since constructing and sharing the full $q \times p$ action matrix $\widehat{\mathbf{G}}_k$ is computationally inefficient, we instead form a compact equivalent:

$$\widehat{\mathbf{C}}_k = (\mathbf{B}_k^\top\mathbf{B}_k)^{1/2}\mathbf{A}_k\widehat{\mathbf{\Sigma}}_k^{1/2} \in \mathbb{R}^{r_\mathrm{L}\times p}. \tag{5}$$

Since $\widehat{\mathbf{C}}_k^\top\widehat{\mathbf{C}}_k = \widehat{\mathbf{G}}_k^\top\widehat{\mathbf{G}}_k$, the compact matrix $\widehat{\mathbf{C}}_k$ has the same right singular subspaces and singular values as the input-aware action matrix $\widehat{\mathbf{G}}_k$. Client k then isolates the highest-energy geometric directions by computing an orthonormal basis $\mathbf{Q}_k = \mathrm{TopEig}_s(\widehat{\mathbf{C}}_k^\top\widehat{\mathbf{C}}_k)$ for the leading rank-s eigenspace, alongside its corresponding eigenvalues $\boldsymbol{\lambda}_k \in \mathbb{R}_+^s$. By sharing this subspace representation, the server captures the dominant directions of the input-weighted actions without requiring clients to synchronize their raw weight matrices.

3.2 SERVER AGGREGATION AND PERSONALIZED FEEDBACK

Upon receiving the local uploads, aggregating these subspaces presents a geometric alignment problem: directly averaging eigenvectors is ill-posed due to sign and basis-rotation ambiguities. Our approach resolves this by reconstructing a stabilized, trace-normalized rank-s action Gram matrix for each client, and then aggregating them strictly as positive-semidefinite (PSD) matrices:

$$\widehat{\mathbf{M}}_k = \frac{\mathbf{Q}_k\,\mathrm{diag}(\boldsymbol{\lambda}_k)\mathbf{Q}_k^\top}{\mathbf{1}^\top\boldsymbol{\lambda}_k + \varepsilon_0}, \qquad \widehat{\mathbf{M}} = \sum_{k=1}^{K}\frac{n_k}{\sum_j n_j}\widehat{\mathbf{M}}_k. \tag{6}$$

Here, $\varepsilon_0 > 0$ acts as a numerical stabilizer. The trace normalization is a critical design choice: it controls for varying top-s action magnitudes across heterogeneous clients, ensuring that clients with larger absolute updates do not disproportionately dominate the geometric consensus. The server then extracts the top-s eigenspace of the aggregated Gram matrix, $\widehat{\mathbf{V}} = \mathrm{TopEig}_s(\widehat{\mathbf{M}})$, and broadcasts the global projector $\widehat{\mathbf{P}} = \widehat{\mathbf{V}}\widehat{\mathbf{V}}^\top$.

Armed with this global geometric consensus, clients resume training their personalized adapters. Rather than overwriting local weights, FedSAIL introduces a penalty that regularizes the local update toward the shared subspace:

$$\min_{\mathbf{A}_k,\mathbf{B}_k}\ \widehat{\mathcal{L}}_k^{\mathrm{task}}(\mathbf{B}_k\mathbf{A}_k) + \frac{\beta}{2}\left\|\mathbf{B}_k\mathbf{A}_k\widehat{\mathbf{\Sigma}}_k^{1/2}(\mathbf{I}_p - \widehat{\mathbf{P}})\right\|_F^2. \tag{7}$$

This objective function selectively penalizes action energy that falls outside the globally shared space. Coefficients operating within that space remain entirely client-specific. Furthermore, the finite penalty weight β provides local adapters with the necessary flexibility to retain unshared, residual directions

if they sufficiently reduce the local task loss. This aligns with our observation in Figure 3: while different tasks share a dominant direction, they still require task-conditioned geometric variations to optimize performance.

Algorithm 1 FedSAIL: Federated Subspace-Guided Action-Informed Learning

Require: Clients $\{\mathcal{D}_k\}_{k=1}^K$; LoRA rank r_{L}; shared rank s; shrinkage γ; penalty β; stabilizer ε_0; rounds T.

for all clients k in parallel **do**
  Estimate $\widehat{\mathbf{\Sigma}}_k$ by (4); locally warm up $(\mathbf{A}_k^0, \mathbf{B}_k^0)$ with $\beta = 0$.
end for
for $t = 1, \ldots, T$ **do**
  for all clients k in parallel **do**
    Form $\widehat{\mathbf{C}}_k^{t-1}$ by (5).
    $\mathbf{Q}_k^t \leftarrow \mathrm{TopEig}_s((\widehat{\mathbf{C}}_k^{t-1})^\top \widehat{\mathbf{C}}_k^{t-1})$; collect the corresponding eigenvalues in $\boldsymbol{\lambda}_k^t$.
    Upload $(\mathbf{Q}_k^t, \boldsymbol{\lambda}_k^t, n_k)$.
  end for
  Server forms $\widehat{\mathbf{M}}^t$ by (6) and broadcasts $\widehat{\mathbf{V}}^t \leftarrow \mathrm{TopEig}_s(\widehat{\mathbf{M}}^t)$.
  for all clients k in parallel **do**
    Update $(\mathbf{A}_k^t, \mathbf{B}_k^t)$ using (7) with $\widehat{\mathbf{P}}^t = \widehat{\mathbf{V}}^t(\widehat{\mathbf{V}}^t)^\top$.
  end for
end for

Ensure: Personalized adapters $\{\mathbf{B}_k^T \mathbf{A}_k^T\}_{k=1}^K$.

4 Experiments

In this section, we evaluate FedSAIL to validate the effectiveness of sharing input-aware action subspaces for personalized adaptation under data heterogeneity. We begin by comparing its predictive performance with existing federated LoRA methods. We then study how local adapter rank and shared subspace rank could affect accuracy and communication cost, before examining the contribution of $\mathbf{G}_k$ sharing through ablation studies. Finally, we evaluate whether layer selection based on cross-client subspace agreement can reduce communication while retaining comparable predictive performance.

4.1 Personalized Adaptation on GLUE

We first evaluate RoBERTa-large on five GLUE benchmarks (RTE, SST-2, QNLI, MNLI, and QQP). To simulate a heterogeneous federated environment, we distribute each task across three clients using a Dirichlet label skew ($\alpha \in \{0.5, 0.75, 1.0\}$) and initialize each client's adapters independently. We compare FedSAIL against standard LoRA (Hu et al., 2022), FFA-LoRA (Sun et al., 2024), FedSA-LoRA (Guo et al., 2025), and FedDPA-LoRA (Yang et al., 2024b). Full optimization details and hyperparameters are provided in Appendix C.

Table 2 shows that FedSAIL achieves the highest six-split mean validation accuracy at all three heterogeneity levels. Its overall mean is 89.33%, compared with 88.69% for FedSA-LoRA and 88.57% for FedDPA-LoRA, although the gains vary across tasks. FFA-LoRA averages 74.62% under the independent-initialization protocol. This is a stress test of aggregating $\mathbf{B}_k$ with different frozen $\mathbf{A}_k$ bases, rather than its canonical shared-$\mathbf{A}$ setting (Appendix C). The results support input-aware subspace guidance for personalized adaptation with independently initialized adapters.

4.2 Shared Rank vs. Local Rank

A key architectural advantage of FedSAIL is its ability to decouple the rank of the globally shared subspace (s) from the capacity of the local adapters (r_{L}). Standard federated parameter-averaging methods usually tie these two dimensions together. In contrast, FedSAIL's asymmetric design allows us to disentangle communication cost from client-side learning capacity. Table 3 reports the performance and communication costs as we vary these ranks.

We first evaluate the communication-accuracy trade-off by fixing the local rank at $r_{\mathrm{L}} = 8$ and varying the shared rank. Expanding the shared subspace from $s = 1$ to $s = 4$ quadruples the communication

Table 2: Personalized GLUE validation accuracy (%, mean ± sample std.) after 500 rounds, with the highest mean in each column for each α shown in bold. Smaller α indicates stronger label heterogeneity.

α	Method	RTE	SST-2	QNLI	MNLI-m	MNLI-mm	QQP	Avg.
0.5	LoRA	$78.70_{\pm 2.60}$	$95.83_{\pm 0.93}$	$90.13_{\pm 1.93}$	$88.68_{\pm 1.60}$	$88.96_{\pm 1.73}$	$89.41_{\pm 5.75}$	$88.62_{\pm 1.14}$
	FFA-LoRA	$70.04_{\pm 5.91}$	$90.75_{\pm 1.78}$	$79.46_{\pm 2.06}$	$65.46_{\pm 9.63}$	$66.65_{\pm 9.11}$	$81.28_{\pm 11.53}$	$75.61_{\pm 3.27}$
	FedDPA-LoRA	$78.22_{\pm 6.83}$	$95.99_{\pm 1.20}$	$91.62_{\pm 1.21}$	$\mathbf{89.06}_{\pm 1.58}$	$\mathbf{89.29}_{\pm 1.71}$	$89.62_{\pm 5.76}$	$88.97_{\pm 1.33}$
	FedSA-LoRA	$79.90_{\pm 8.61}$	$95.95_{\pm 1.13}$	$91.76_{\pm 0.61}$	$88.56_{\pm 1.72}$	$88.92_{\pm 1.76}$	$89.53_{\pm 5.63}$	$89.10_{\pm 1.60}$
	FedSAIL	$\mathbf{80.87}_{\pm 7.50}$	$\mathbf{96.37}_{\pm 1.17}$	$\mathbf{93.06}_{\pm 0.57}$	$88.98_{\pm 1.54}$	$89.07_{\pm 1.75}$	$\mathbf{89.80}_{\pm 5.39}$	$\mathbf{89.69}_{\pm 1.28}$
0.75	LoRA	$\mathbf{77.26}_{\pm 0.96}$	$95.45_{\pm 0.35}$	$90.76_{\pm 0.89}$	$88.39_{\pm 1.43}$	$88.62_{\pm 1.44}$	$88.68_{\pm 4.49}$	$88.19_{\pm 0.54}$
	FFA-LoRA	$61.13_{\pm 3.47}$	$89.22_{\pm 2.81}$	$81.77_{\pm 2.42}$	$62.33_{\pm 10.84}$	$63.60_{\pm 10.45}$	$81.53_{\pm 8.88}$	$73.26_{\pm 1.81}$
	FedDPA-LoRA	$74.61_{\pm 0.91}$	$95.49_{\pm 0.46}$	$92.12_{\pm 0.17}$	$\mathbf{88.81}_{\pm 1.44}$	$88.83_{\pm 1.45}$	$88.67_{\pm 4.71}$	$88.09_{\pm 0.27}$
	FedSA-LoRA	$76.65_{\pm 3.86}$	$95.41_{\pm 0.41}$	$92.34_{\pm 0.29}$	$88.22_{\pm 1.58}$	$88.42_{\pm 1.75}$	$88.70_{\pm 4.87}$	$88.29_{\pm 0.39}$
	FedSAIL	$\mathbf{77.26}_{\pm 5.05}$	$\mathbf{95.64}_{\pm 0.20}$	$\mathbf{93.23}_{\pm 0.61}$	$88.61_{\pm 1.53}$	$\mathbf{88.84}_{\pm 1.60}$	$\mathbf{89.05}_{\pm 4.69}$	$\mathbf{88.77}_{\pm 0.56}$
1.0	LoRA	$79.78_{\pm 2.53}$	$95.83_{\pm 0.69}$	$92.20_{\pm 0.94}$	$88.18_{\pm 1.25}$	$88.41_{\pm 1.51}$	$88.00_{\pm 2.33}$	$88.73_{\pm 1.00}$
	FFA-LoRA	$76.29_{\pm 4.22}$	$87.88_{\pm 0.98}$	$80.08_{\pm 1.36}$	$61.48_{\pm 11.53}$	$62.48_{\pm 10.99}$	$81.69_{\pm 4.80}$	$74.98_{\pm 4.05}$
	FedDPA-LoRA	$77.98_{\pm 2.89}$	$95.80_{\pm 1.26}$	$92.80_{\pm 0.55}$	$\mathbf{88.40}_{\pm 1.32}$	$\mathbf{88.80}_{\pm 1.70}$	$88.18_{\pm 2.16}$	$88.66_{\pm 1.11}$
	FedSA-LoRA	$79.54_{\pm 4.35}$	$95.83_{\pm 0.81}$	$92.32_{\pm 0.66}$	$88.10_{\pm 1.52}$	$88.17_{\pm 1.60}$	$88.18_{\pm 2.14}$	$88.69_{\pm 1.31}$
	FedSAIL	$\mathbf{81.95}_{\pm 3.61}$	$\mathbf{96.25}_{\pm 1.10}$	$\mathbf{93.28}_{\pm 0.41}$	$\mathbf{88.40}_{\pm 1.40}$	$88.52_{\pm 1.35}$	$\mathbf{88.72}_{\pm 2.11}$	$\mathbf{89.52}_{\pm 1.14}$

payload, yet yields only small, non-monotonic changes in accuracy. This confirms that sharing dominant action direction captures the most critical geometric alignment information. Then we fix the shared rank at the communication-efficient $s = 1$ and scale up the local adapter rank r_{L} from 2 to 8, predictive performance steadily improves across all heterogeneity levels. Crucially, since the shared spectral message depends entirely on s, scaling up r_{L} enhances local personalization capacity without incurring any additional communication overhead.

Table 3: Sensitivity to shared and local ranks on personalized GLUE . We report averaged validation accuracy (%, mean ± sample std.), adapter parameter counts, and total communication costs.

r_{L}	s	Params. (M)	Comm. (GiB)	$\alpha = 0.5$	$\alpha = 0.75$	$\alpha = 1.0$
8	1	0.786	0.4949	$89.69_{\pm 1.28}$	$88.77_{\pm 0.56}$	$89.52_{\pm 1.14}$
8	2	0.786	0.9895	$89.83_{\pm 1.08}$	$88.87_{\pm 0.49}$	$\mathbf{89.65}_{\pm 0.98}$
8	4	0.786	1.9787	$\mathbf{89.91}_{\pm 1.48}$	$89.15_{\pm 0.18}$	$89.47_{\pm 1.03}$
2	1	0.197	0.4949	$88.74_{\pm 1.43}$	$87.92_{\pm 0.55}$	$88.65_{\pm 1.41}$
4	1	0.393	0.4949	$89.21_{\pm 1.51}$	$88.24_{\pm 0.81}$	$89.10_{\pm 1.09}$
16	1	1.573	0.4949	$89.84_{\pm 1.43}$	$\mathbf{89.22}_{\pm 0.29}$	$89.61_{\pm 0.95}$

4.3 Ablation Study: Input-Moment Weighting in the Shared Message

To isolate the source of FedSAIL's empirical gains, we ablate the rank-1 spectral message uploaded by the clients. To evaluate these properties, we compare shared subspaces extracted from the raw input factor ($\mathbf{A}_k$), the full raw update ($\mathbf{\Delta}_k = \mathbf{B}_k\mathbf{A}_k$), and the input-aware action matrix ($\widehat{\mathbf{G}}_k = \mathbf{\Delta}_k\widehat{\mathbf{\Sigma}}_k^{1/2}$). All variants use identical communication payloads and retain the same $\widehat{\mathbf{G}}_k$-based local regularizer. In Appendix D, we provide the protocol details and task-level results.

As shown in Table 4, the input-aware $\widehat{\mathbf{G}}_k$ message outperforms both unweighted messages across all three heterogeneity settings. At fixed communication cost, input-moment weighting therefore improves the guidance provided by the shared subspace. Although $\mathbf{\Delta}_k = \mathbf{B}_k\mathbf{A}_k$ is already invariant to invertible LoRA reparameterizations, its unweighted message performs poorly under the same action-based regularizer. This supports selecting shared directions that reflect local input distributions even after removing parameterization ambiguity. We shall note that in Appendix D, an additional ablation study reveals that extracting the subspace from $\mathbf{A}_k\widehat{\mathbf{\Sigma}}_k^{1/2}$ performs only marginally lower than the full $\widehat{\mathbf{G}}_k$. Therefore, we do not claim that including the output factor $\mathbf{B}_k$ is strictly necessary

for empirical performance improvements. However, in comparison to $\mathbf{A}_k\widehat{\mathbf{\Sigma}}_k^{1/2}$, the full action matrix $\widehat{\mathbf{G}}_k$ is invariant to invertible LoRA reparameterizations, which is a desirable property for a shared geometric object.

Table 4: Ablation of the shared message on personalized GLUE. We report validation accuracy (%, mean $\pm$ sample std.) with rank-1 payloads.

Shared message	Input-aware	$\alpha = 0.5$	$\alpha = 0.75$	$\alpha = 1.0$	Overall
$\mathbf{A}_k$	No	$68.75_{\pm 2.92}$	$66.17_{\pm 1.41}$	$68.95_{\pm 3.47}$	$67.96_{\pm 1.20}$
$\mathbf{\Delta}_k = \mathbf{B}_k\mathbf{A}_k$	No	$68.79_{\pm 3.05}$	$66.17_{\pm 1.42}$	$69.04_{\pm 3.41}$	$68.00_{\pm 1.22}$
$\widehat{\mathbf{G}}_k = \mathbf{\Delta}_k\widehat{\mathbf{\Sigma}}_k^{1/2}$	Yes	$\mathbf{89.69}_{\pm 1.28}$	$\mathbf{88.77}_{\pm 0.56}$	$\mathbf{89.52}_{\pm 1.14}$	$\mathbf{89.33}_{\pm 0.36}$

4.4 Layer selection and communication

As observed in Section 2.3, under highly heterogeneous environments, such as when clients operate across different domains and different tasks (DDDT), cross-client geometric agreement tends to decline sharply in deeper layers. To exploit this property, we construct a DDDT benchmark with eight RoBERTa-large clients split across IMDb/Amazon (sentiment) and QQP/QNLI (paraphrase/entailment). After a local warmup phase, we evaluate the cross-client geometric agreement at each layer. If the subspace agreement falls below a threshold τ, that layer is excluded from subspace sharing entirely and remains purely local.

Table 5: Layer selection performance on DDDT. We report averaged held-out accuracy (%, mean $\pm$ sample std.) at round 64 and total communication cost, including selector overhead.

FedSAIL variant	Accuracy (%)	Comm. (MiB)	Saving (%)
Full	90.033 ± 0.225	96.094	–
Selection, $\tau = 0.6$	90.133 ± 0.038	77.589	19.26
Selection, $\tau = 0.7$	89.908 ± 0.800	58.904	38.70

At $\tau = 0.6$, the selector leaves the final six layers (zero-based layers 18–23) local across all three training seeds, without an imposed contiguity constraint. This pattern is consistent with the decline in cross-client agreement at deeper layers observed in Section 2.3. Table 5 shows that restricting guidance to layers with greater action-subspace agreement reduces communication by 19.26%, including selector overhead, while retaining comparable mean held-out accuracy. A higher threshold further reduces communication, with a modest decrease in mean accuracy. These results support using action geometry to guide where clients share subspaces, allowing the depth of collaboration to adapt to observed agreement while preserving local training in unshared layers. Appendix F provides the selection rule, complete threshold sweep, and communication accounting.

5 Conclusion

In this paper, we demonstrate that high parameter similarity in federated LoRA often reflects common initialization rather than genuine task learning. By shifting the focus from weight space to input-aware action space, we identify a robust, task-conditioned structure that genuinely generalizes across heterogeneous clients. Leveraging this geometric insight, we introduce FedSAIL. Instead of enforcing rigid parameter synchronization, FedSAIL aligns clients through a basis-invariant spectral consensus, providing regularization while preserving local coefficients and residual directions. Extensive evaluations confirm that FedSAIL consistently outperforms existing federated LoRA baselines. Additionally, by measuring layer-wise subspace agreement, our framework can use warmup-based selection to suspend communication at deeper layers with low cross-client agreement, reducing payload costs significantly while retaining comparable mean predictive accuracy. Ultimately, we establish that effective federated fine-tuning must be data-driven: by anchoring consensus in local input distributions, it ensures clients collaborate where tasks align and diverge where personalization is essential.

AI USE STATEMENT

We used generative AI tools to provide feedback on the experimental methodology and to assist with interpreting results. We also used these tools to revise manuscript and figures. The authors take responsibility for the final content, including AI-assisted text, analyses, and figures.

ETHICS STATEMENT

This work focuses on algorithmic design in federated learning and utilizes standard, publicly available datasets. The research does not involve human subjects, sensitive personal information, or proprietary data. Consequently, this paper does not present any specific ethical issues or potential for negative societal impact.

REPRODUCIBILITY STATEMENT

Algorithm 1 details the core FedSAIL framework. To ensure full reproducibility, comprehensive experimental details—including dataset construction, client partitioning, hyperparameter configurations, and exact communication accounting—are provided in Appendices B through F. Furthermore, the core implementation of FedSAIL, along with numerical tests and baseline methods, is anonymously available at https://anonymous.4open.science/r/FedSAIL-4A7C/.

REFERENCES

Hao Ban and Kaiyi Ji. Rethinking parameter sharing for llm fine-tuning with multiple loras. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 12843–12863, 2026.

Jieming Bian, Lei Wang, Letian Zhang, and Jie Xu. Lora-fair: Federated lora fine-tuning with aggregation and initialization refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3737–3746, 2025.

Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International conference on machine learning*, pp. 10323–10337. PMLR, 2023.

Pengxin Guo, Shuang Zeng, Yanran Wang, Huijie Fan, Feifei Wang, and Liangqiong Qu. Selective aggregation for low-rank adaptation in federated learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

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 *The Tenth International Conference on Learning Representations*, 2022.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. *arXiv preprint arXiv:2212.09849*, 2022.

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

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017.

Fabian Paischer, Lukas Hauzenberger, Thomas Schmied, Benedikt Alkin, Marc Deisenroth, and Sepp Hochreiter. Parameter efficient fine-tuning via explained variance adaptation. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://doi.org/10.52202/085713-1539.

Riccardo Salami, Pietro Buzzega, Matteo Mosconi, Jacopo Bonato, Luigi Sabetta, and Simone Calderara. Closed-form merging of parameter-efficient modules for federated continual learning. In *International Conference on Learning Representations*, volume 2025, pp. 86612–86637, 2025.

Youbang Sun, Zitao Li, Yaliang Li, and Bolin Ding. Improving lora in privacy-preserving federated learning. In *The Twelfth International Conference on Learning Representations*, 2024.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP workshop BlackboxNLP: Analyzing and interpreting neural networks for NLP*, pp. 353–355, 2018.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. Svd-llm: Truncation-aware singular value decomposition for large language model compression. In *International Conference on Learning Representations*, volume 2025, pp. 19299–19319, 2025.

Ziyao Wang, Zheyu Shen, Yexiao He, Guoheng Sun, Hongyi Wang, Lingjuan Lyu, and Ang Li. Flora: Federated fine-tuning large language models with heterogeneous low-rank adaptations. *Advances in Neural Information Processing Systems*, 37:22513–22533, 2024.

Yibo Yang, Xiaojie Li, Zhongzhu Zhou, Shuaiwen Leon Song, Jianlong Wu, Liqiang Nie, and Bernard Ghanem. CorDA: Context-oriented decomposition adaptation of large language models for task-aware parameter-efficient fine-tuning. In *Advances in Neural Information Processing Systems*, volume 37, 2024a. URL https://doi.org/10.52202/079017-2292.

Yiyuan Yang, Guodong Long, Tao Shen, Jing Jiang, and Michael Blumenstein. Dual-personalizing adapter for federated foundation models. *Advances in Neural Information Processing Systems*, 37: 39409–39433, 2024b.

Haoran Zhang, Dongjun Kim, Seohyeon Cha, and Haris Vikalo. FedRot-LoRA: Mitigating rotational misalignment in federated LoRA. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.23638.

Jianyi Zhang, Saeed Vahidian, Martin Kuo, Chunyuan Li, Ruiyi Zhang, Tong Yu, Guoyin Wang, and Yiran Chen. Towards building the federatedgpt: Federated instruction tuning. In *ICASSP 2024-2024 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 6915–6919. IEEE, 2024.

Jiacheng Zhu, Kristjan Greenewald, Kimia Nadjahi, Haitz Sáez de Ocáriz Borde, Rickard Brüel Gabrielsson, Leshem Choshen, Marzyeh Ghassemi, Mikhail Yurochkin, and Justin Solomon. Asymmetry in low-rank adapters of foundation models. In *Forty-first International Conference on Machine Learning*, 2024.

Overview of the appendices

The appendices provide the construction, diagnostic protocols, and additional results supporting the main text. They are organized as follows.

- Appendix A provides further details for Remark 1, showing how shared action geometry can coexist with misaligned LoRA rowspaces.
- Appendix B describes the RoBERTa and GPT-2 diagnostics for initialization effects, concentration in dominant action directions, and task and layer dependence, including their controls and statistical summaries.
- Appendix C specifies the client partitions, training settings, baseline implementations, and evaluation protocol for personalized GLUE.
- Appendix D reports the shared-message ablation, which varies the use of output-factor and input-moment weighting while keeping the local action penalty fixed.
- Appendix E studies shared and local rank sensitivity, with task-level accuracy, adapter parameter counts, and communication accounting.
- Appendix F details the real-data DDDT layer-selection study, including the benchmark protocol, geometric selection rule, threshold sweep, and selector overhead.
- Appendix G describes the anonymous code release, its reproducibility scope, and the computational resources.

A Details of Remark 1

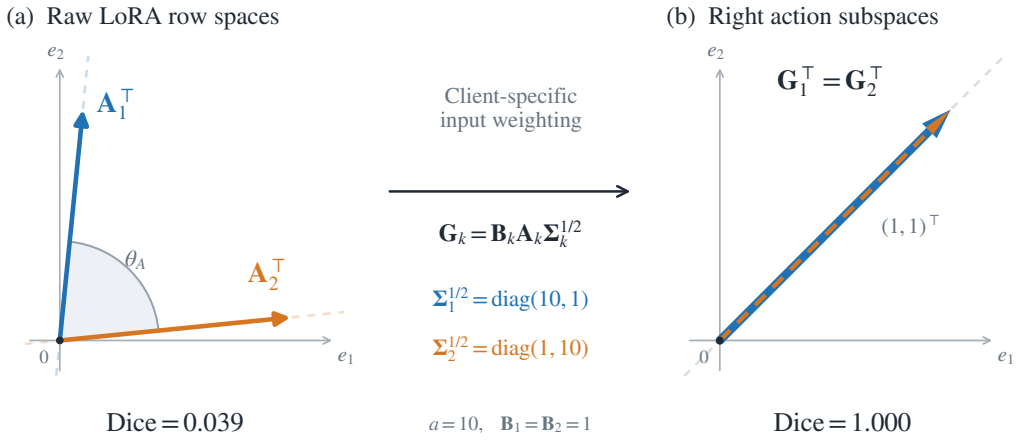


Figure 4: Illustration of Remark 1, with $a = 10$ and $\mathbf{B}_1 = \mathbf{B}_2 = 1$. (a) The raw $\mathbf{A}_k$ row spaces have Dice 0.039. (b) After client-specific input weighting, $\mathbf{G}_1 = \mathbf{G}_2 = (1, 1)$, so the right action subspaces coincide with Dice one. Blue and orange denote clients 1 and 2, respectively.

The construction in Remark 1 relies on each adapter compensating for its client's input scaling. With $\mathbf{\Sigma}_1^{1/2} = \mathrm{diag}(a, 1)$ and $\mathbf{\Sigma}_2^{1/2} = \mathrm{diag}(1, a)$, we obtain

$$\mathbf{G}_1 = (a^{-1}, 1)\,\mathrm{diag}(a, 1) = (1, 1),$$
$$\mathbf{G}_2 = (1, a^{-1})\,\mathrm{diag}(1, a) = (1, 1).$$

For each client, the smaller adapter coefficient exactly offsets the larger input scale in the corresponding coordinate. Both clients therefore have the same right action direction $\mathbf{v} = 2^{-1/2}(1, 1)^\top$. The corresponding projector $\mathbf{P} = \mathbf{v}\mathbf{v}^\top$ satisfies

$$\mathbf{G}_k(\mathbf{I}_2 - \mathbf{P}) = 0, \qquad k = 1, 2,$$

and hence captures all action energy for both clients. Despite this agreement, the raw LoRA rowspaces become increasingly misaligned. Their Projector Dice is

$$\frac{\langle \mathbf{A}_1, \mathbf{A}_2 \rangle_F^2}{\|\mathbf{A}_1\|_F^2 \|\mathbf{A}_2\|_F^2} = \frac{\left(2a^{-1}\right)^2}{\left(1 + a^{-2}\right)^2} = \left(\frac{2a}{a^2+1}\right)^2 \to 0$$

as $a \to \infty$. Thus, the raw rowspaces are nearly orthogonal even though their input-weighted action directions remain identical. Figure 4 illustrates this contrast at $a = 10$.

This example also clarifies the distinction between sharing an action subspace and sharing an input factor. For every $a > 1$, the raw updates $\mathbf{\Delta}_1 = \mathbf{A}_1$ and $\mathbf{\Delta}_2 = \mathbf{A}_2$ are nonproportional. They cannot therefore be reproduced by a single shared rank-one factor $\mathbf{A}$ with client-specific scalar coefficients $\mathbf{B}_k$, since all products $\mathbf{B}_k\mathbf{A}$ would be proportional. A common action subspace thus accommodates both local updates under their respective input regimes, whereas exact input-factor sharing cannot represent them at the same rank.

B Additional Diagnostics of Shared Action Subspaces

In this section, we supply the protocols and measurements behind Section 2. We first examine initialization effects in RoBERTa/RTE diagnostic, then compare dominant and complete action subspaces, and finally separate task, domain, and layer effects in GPT-2. The RoBERTa comparison isolates the choice of adapter initialization under a fixed local-training protocol. These are diagnostics of locally trained adapters; the federated training evaluations are reported separately in Section 4.1 and Appendices C–F.

B.1 Experimental setup and measurement

Training and initialization. Table 6 summarizes the two studies. In both studies we freeze the pretrained backbone and place LoRA on the query and value projections in all 12 transformer layers. Common initialization uses identical initial $\mathbf{A}$ matrices at corresponding projections across clients; independent initialization draws them separately. Both conditions initialize $\mathbf{B} = 0$. Client partitions, optimization settings, minibatch orders, and training dropout streams are held fixed across conditions. For RoBERTa, the dropout seed is reset after injecting the initial adapters; the classification head has a common initialization within each seed. Client 0 uses the same initial $\mathbf{A}$ in both conditions and serves as a paired anchor, not an independent replicate. RoBERTa uses SGD without momentum, weight decay, or gradient clipping. GPT-2 uses AdamW with zero weight decay and gradient clipping at norm one.

Table 6: Local-training protocols for the structural diagnostics.

Setting	RoBERTa-base / RTE	GPT-2 124M
Clients	3 per partition setting	2 domains $\times$ 2 tasks $\times$ 2 replicas
Train / calibration / evaluation	161–1543 / 9–86 / 10–87	$500/250/250$
Local LoRA rank / scale	$8/16$	$4/8$
LoRA dropout	0.05	0
Steps / effective batch size	$200/128$	$200/16$
Maximum sequence length	128	64
Optimizer	SGD	AdamW
LoRA learning rate	10^{-2}	3×10^{-4}
Classifier learning rate	10^{-2}	No additional classifier
Training seeds	$123, 124, 125$	$11, 23, 37, 53, 71$

We emphasize that the studies in this section is not to train high-performing models (full federated evaluations are deferred to Section 4). Instead, we simply want to examine how different adapter initializations affect local parameter updates across heterogeneous clients. 200 local steps without server aggregation are fully sufficient for this purpose.

RoBERTa client construction. We partition the 2,490 official RTE training examples across three clients using either IID shuffling or label-based Dirichlet allocation ($\alpha \in \{0.5, 1.0\}$). Under IID partitioning, each client receives exactly 830 examples. Under Dirichlet allocation, local dataset sizes vary from 161 to 1,543, with entailment ratios ranging between 8.67% and 94.41%. The 277 official validation examples are distributed uniformly for IID clients and according to local training-label priors for Dirichlet clients. Each client's data is further stratified by label into calibration and evaluation subsets, retaining minor strata (e.g., the smallest calibration subset contains 9 examples from a single class).

GPT-2 tasks and token regimes. We evaluate controlled synthetic tasks using template-generated movie and product sentences across two classification objectives: sentiment and tense. For a given domain and replica, both tasks share identical underlying content but differ in instructions and labels. Each prompt comprises a task instruction ending in `Text:`, the content sentence, and an `Answer:` suffix. Training minimizes cross-entropy over the two candidate answer-token logits at the final prompt position, without appending answer tokens to the input. We divide each prompt into three disjoint diagnostic masks: *content* (sentence tokens), *decision* (the final prompt token predicting the answer), and *instruction* (all remaining tokens, including non-decision suffix tokens). Held-out examples are jointly stratified by sentiment and tense into calibration and evaluation splits.

B.2 Initialization and factor agreement

Table 7: Cross-client projector Dice in RoBERTa-base/RTE averaged across client pairs, layers, and projections. All entries use the fixed step-200 checkpoint.

Partition	Initialization	$\mathbf{A}$	$\mathbf{B}$	$\mathbf{G}$: top 1	$\mathbf{G}$: top 2	$\mathbf{G}$: full
IID	Common	1.000	0.199	0.978	0.825	0.876
$\alpha = 1$	Common	1.000	0.185	0.974	0.834	0.871
$\alpha = 0.5$	Common	1.000	0.177	0.965	0.818	0.835
IID	Independent	0.011	0.193	0.814	0.452	0.159
$\alpha = 1$	Independent	0.011	0.180	0.810	0.454	0.160
$\alpha = 0.5$	Independent	0.011	0.177	0.810	0.455	0.159

Following Guo et al. (2025), here we evaluate structural alignment using both the flattened-factor cosine similarity and basis-invariant subspace agreement (Table 7). Under common initialization, the flattened cosine for the final matrix $\mathbf{A}$ reaches 1, whereas under independent initialization, it drops to -0.0003 (with the corresponding $\mathbf{B}$ cosines at 0.2238 and -0.0016, respectively). Similarly, the final-$\mathbf{A}$ rowspace Dice coefficient is 1.0000 under common initialization versus 0.0105 under independent initialization. Notably, 0.0105 coincides with the random baseline expectation of $8/768 \approx 0.0104$ for independent rank-8 subspaces in 768 dimensions.

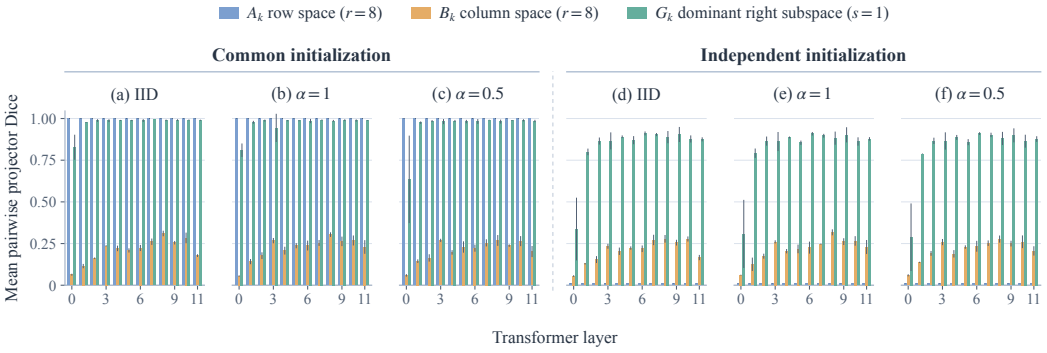


Figure 5: Query-projection subspace agreement in RoBERTa-base/RTE. For each initialization group, label heterogeneity increases from left to right, with $\alpha \in \{0.5, 1.0\}$ controlling Dirichlet label skew.

Figure 5 provides the query-projection counterpart to Figure 2. Both include all 12 layers, with visibly lower independent-initialization $\mathbf{G}$ agreement at layer 0. At each layer, the three unordered client pairs are averaged within seed; the plotted mean and standard deviation are computed across three seeds. Pairs sharing a client are not treated as independent replications.

Table 8: Rank-4 $\mathbf{A}$-rowspace Dice for GPT-2, averaged over layers and query/value projections. S_{11}: same domain and task, S_{10}: same domain and different tasks, S_{01}: different domains and same task, S_{00}: different domains and tasks.

Initialization	S_{11}	S_{10}	S_{01}	S_{00}
Common	0.9412	0.8288	0.9082	0.8221
Independent	0.0806	0.0383	0.0687	0.0355

The same initialization dependence is corroborated by the GPT-2 study (Table 8). Here, S_{dt} identifies pairs with same-domain indicator d and same-task indicator t. While independent initialization sharply reduces the complete $\mathbf{A}$-rowspace overlap across all pairs, same-task pairs (S_{11}, S_{01}) retain slightly more overlap than different-task pairs. Thus, the loss of high absolute agreement in weight space should not be equated with the complete absence of task-specific geometry; rather, it demonstrates that raw parameter similarity is too fragile to serve as a robust medium for federated sharing.

B.3 Dominant and complete action-subspace agreement

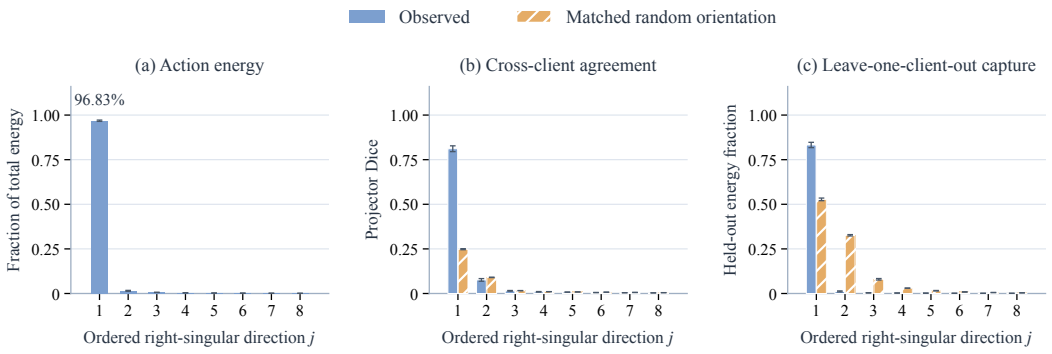


Figure 6: Ordered-direction diagnostics on RoBERTa-base/RTE with independent initialization and LoRA rank eight. Panels show (a) evaluation action energy fraction, (b) cross-client rank-one projector Dice, and (c) LOO action-energy capture. Metrics are averaged across partitions, layers, projections, and client pairs.

Agreement is concentrated in the leading direction. The leading direction captures 96.90% and 96.83% of the evaluation action energy under common and independent initialization, respectively, with a calibration/evaluation Dice coefficient of 0.9813 in both cases. Under independent initialization, cross-client Dice agreement drops sharply as rank increases: 0.8113 for the leading direction, 0.4536 for the top-two subspace, and 0.1593 for the full numerical-rank space (note that these rank-normalized overlaps do not measure captured energy). To examine individual directions, Figure 6 evaluates all eight components separately. For each direction, we construct a leave-one-client-out (LOO) projector using the leading eigenvector of the mean calibration projector from the other two clients; LOO capture then measures the fraction of held-out evaluation action energy retained by this projector. As a control baseline, we replace the right singular basis of $(\mathbf{B}_k^\top \mathbf{B}_k)^{1/2}\mathbf{A}_k$ with a uniformly sampled orthonormal basis while preserving its singular spectrum and local moments (averaged over 200 draws per seed, partition, layer, and projection). The leading direction substantially outperforms this control in both Dice agreement (0.811 vs. 0.248) and LOO capture (0.832 vs. 0.528), showing consistent positive gains across all seed means. In contrast, directions 2 through 8 perform below control on average across both metrics. These diagnostic findings confirm that cross-client agreement is strongly concentrated in the leading direction. We note that direction-wise LOO captures

are non-additive and serve as diagnostic insights rather than guarantees of downstream aggregation performance.

B.4 Task, domain, and layer dependence in GPT-2

For Figure 3, S_{dt} denotes the mean fixed-top-1 action-subspace Dice in each of the four pair classes. There are 4 pairs in S_{11} and 8 each in S_{10}, S_{01}, and S_{00}. We first average within each class, then compute the balanced factorial contrasts

$$C_{\text{task}} = \tfrac{1}{2}(S_{11} + S_{01} - S_{10} - S_{00}), \qquad C_{\text{domain}} = \tfrac{1}{2}(S_{11} + S_{10} - S_{01} - S_{00}), \tag{8}$$

which give each pair class equal weight. A positive C_{task} indicates greater agreement between same-task clients after averaging equally over same-domain and different-domain pairs. The comparison $S_{01} - S_{10} = C_{\text{task}} - C_{\text{domain}}$ asks the stronger question of whether task-related differences exceed domain-related differences. Its sign describes their relative strength; a negative value can coexist with a positive task contrast.

Table 9: Task and domain organization of the fixed rank-1 dominant right direction of $\mathbf{G}_k$. Mean values and 95% t intervals (in brackets) are averaged over query and value projections.

Initialization	Tokens	C_{task}	C_{domain}	$S_{01} - S_{10}$
Common	Instruction	.224[.207, .240]	.008[.001, .016]	.215[.195, .235]
Common	Content	.212[.202, .222]	.047[.042, .052]	.165[.153, .176]
Common	Decision	.242[.221, .263]	.023[.017, .030]	.218[.196, .241]
Independent	Instruction	.201[.190, .213]	.011[.003, .018]	.191[.178, .203]
Independent	Content	.201[.192, .211]	.039[.033, .045]	.162[.153, .171]
Independent	Decision	.237[.222, .252]	.023[.017, .029]	.214[.198, .230]

At $s = 1$, task contrasts are positive and exceed domain contrasts under both initialization conditions in all three token regions (Table 9). Different-task pairs nevertheless retain mean Dice of approximately 0.65–0.67. Task dependence therefore appears as a difference within partly shared geometry, not as orthogonality between tasks. In the layerwise curves, this difference is most pronounced in later content and decision layers. Early decision layers remain highly aligned even across tasks. These depth-dependent patterns motivate a layerwise sharing criterion; the effects of actually disabling guidance are tested separately in Appendix F.

Table 10: Task-minus-domain contrasts in GPT-2 across projections and retained ranks, averaged over replicates and 12 layers. Entries report $S_{01} - S_{10} = C_{\text{task}} - C_{\text{domain}}$, with negative values indicating a larger domain contrast. Q and V denote query and value.

Initialization	Tokens	Projection	$s = 1$	$s = 2$	$s = 3$	$s = 4$
Common	Instruction	Q	.211	.246	.288	.318
		V	.220	.287	.328	.356
	Content	Q	.126	.097	.016	−.061
		V	.204	.150	.016	−.067
	Decision	Q	.210	.188	.129	.083
		V	.227	.207	.144	.082
Independent	Instruction	Q	.191	.138	.132	.129
		V	.190	.137	.123	.117
	Content	Q	.126	.079	.047	.028
		V	.199	.123	.082	.052
	Decision	Q	.210	.156	.121	.092
		V	.219	.203	.148	.107

Projection and retained-rank checks. Table 10 separates query and value and varies the retained analysis dimension s on the same rank-4 adapters. The balanced task contrast C_{task}, averaged over query and value, remains positive in every token region under both initialization schemes for all $s \in \{1, 2, 3, 4\}$. Task-conditioned differences therefore remain detectable in larger retained subspaces in this experiment. Under independent initialization, $S_{01} - S_{10}$ remains positive for both projections

at every tested rank, with all seed-level intervals excluding zero; $s = 1$ gives the largest contrast in each token region. Under common initialization, the full content-token subspace ($s = 4$) has a larger domain contrast than task contrast, yielding negative $S_{01} - S_{10}$ for both projections. Its projection-averaged task contrast nevertheless remains positive: $C_{\text{task}} = 0.190$ with a 95% seed-level t interval of $[0.183, 0.197]$. Thus, the reversal concerns the relative strength of task and domain structure, while task dependence remains present. The relative contributions vary with initialization and retained rank.

C Personalized GLUE: Training and Evaluation Protocol

C.1 Tasks, client partitions, and common settings

We evaluate RTE, SST-2, QNLI, MNLI, and QQP, with a separate three-client federation for each task. MNLI-m and MNLI-mm are reported separately, giving six evaluation splits. Training examples are partitioned by a label-skew Dirichlet distribution with $\alpha \in \{0.5, 0.75, 1.0\}$; smaller α produces stronger label heterogeneity. Each client's validation partition follows the empirical label mixture of its training partition. For each task, α, and seed, all methods use the same client training and validation indices.Table 11 gives the common settings for the main comparison and the GLUE ablation and rank studies. The latter vary only the settings identified in Appendices D–E.

Table 11: Common training settings for personalized GLUE. FedSAIL-specific settings also apply to its message and rank ablations unless stated otherwise.

Setting	Configuration
Backbone and adapters	RoBERTa-large; query and value projections in all 24 layers; default $r_{\text{L}} = 8$, $\alpha_{\text{LoRA}} = 16$, LoRA dropout 0.05.
Initialization	Independent client $\mathbf{A}_k$, zero $\mathbf{B}_k$; identical initial classification heads, trained locally.
Data and batching	Full task training/validation sets; maximum length 128; batch and micro-batch size 128.
Optimization	SGD with CUDA AMP; 500 rounds; 10 local steps per client and phase per round.
Main-comparison seeds	123, 124, and 125.
Evaluation	Personalized validation every 25 rounds; no global-model evaluation.
FedSAIL guidance	Shared rank $s = 1$, logged penalty $\beta = 1$; 50 local-only warmup rounds, then 450 guided rounds; layer selection disabled.
Input moments	Estimated from the first 128 examples of each client's training partition before round 1; shrinkage $\gamma = 0.05$; frozen throughout training.

C.2 Method-specific settings and scale convention

In Table 12 we distinguish the communication and optimization rules. All classification heads remain client-local. Baselines aggregate uniformly from the first round. Our FedSAIL approach starts sharing after warmup and weights its trace-normalized spectral messages by the number of local tokens used to estimate the moments.

Table 12: Method-specific GLUE settings. Steps are local steps per client and round; FedDPA-LoRA uses separate global and local phases.

Method	Federated exchange	Learning rate	Steps
LoRA	Local $\mathbf{A}_k$ and $\mathbf{B}_k$	0.005	10
FFA-LoRA	Average $\mathbf{B}_k$; freeze local $\mathbf{A}_k$	0.005	10
FedSA-LoRA	Average $\mathbf{A}_k$; keep $\mathbf{B}_k$ local	0.005	10
FedDPA-LoRA	Average global factors; retain local branch	0.005	$10 + 10$
FedSAIL	Share action spectra and consensus bases	0.005	10

We shall note that FedDPA-LoRA has separate rank-8 global and local adapters and uses a fixed global/local prediction mixture of $0.5/0.5$. Its adapter capacity and scheduled local-step budget are twice those of the single-branch methods. Another thing that worth attention is that for FFA-LoRA, freezing independently initialized $\mathbf{A}_k$ differs from its canonical shared-$\mathbf{A}$ setting, so that the reported performance may not achieve its best possible performance.

D SHARED-MESSAGE ABLATION ON PERSONALIZED GLUE

Controlled comparison. To understand exactly what makes the shared consensus effective, we isolate the components of our transmitted message. Specifically, we ask whether the performance gains stem from incorporating the local input moments ($\widehat{\mathbf{\Sigma}}_k^{1/2}$), the up-projection ($\mathbf{B}_k$), or their combination. In our experiments, we compare the rank-1 right spectral messages extracted from four variants: $\mathbf{A}_k$, the full raw update $\mathbf{\Delta}_k$, $\mathbf{A}_k\widehat{\mathbf{\Sigma}}_k^{1/2}$, and our full input-aware action matrix $\widehat{\mathbf{G}}_k = \mathbf{\Delta}_k\widehat{\mathbf{\Sigma}}_k^{1/2}$. Besides, to ensure a strict comparison of the messages themselves, we hold the local optimization objective constant across all variants. We evaluate these variants using seed 123 across all three Dirichlet heterogeneity settings. To rule out confounding initialization effects, all configurations are strictly matched from step zero, sharing identical client partitions, data-loader states, and warmup trajectories up to round 50.

Results and interpretation. As shown in Table 13, incorporating the input-moment weighting improves performance across all 18 task-by-heterogeneity configurations. Without $\widehat{\mathbf{\Sigma}}_k^{1/2}$, both unweighted messages ($\mathbf{A}_k$ and $\mathbf{\Delta}_k$) suffer severe performance degradation, averaging only 66.68%. Notably, even though sharing the full raw update $\mathbf{\Delta}_k$ resolves the non-uniqueness (reparameterization ambiguity) of individual LoRA factors, it still performs poorly under the fixed action-based regularizer. This demonstrates a key insight: merely fixing factorization ambiguity is insufficient. To provide meaningful cross-client guidance, the shared geometric consensus must be explicitly anchored in the local input distribution. We also observe that extracting the shared subspace from $\mathbf{A}_k\widehat{\mathbf{\Sigma}}_k^{1/2}$ yields an overall mean (88.92%) that is only marginally lower than using the full action matrix $\widehat{\mathbf{G}}_k$ (89.02%). Consistent with our discussion in the main text, this minor empirical gap suggests that including the output factor $\mathbf{B}_k$ in the shared message is not strictly necessary for achieving high predictive performance on these tasks. However, our preference for the full action matrix $\widehat{\mathbf{G}}_k$ is mathematically motivated. Unlike the right subspace of $\mathbf{A}_k\widehat{\mathbf{\Sigma}}_k^{1/2}$, $\widehat{\mathbf{G}}_k$ is inherently invariant to invertible LoRA reparameterizations. This coordinate-free property makes it a fundamentally more robust and principled geometric object to share across heterogeneous clients. Finally, we note that these single-seed results specifically assess the composition of the shared message; to maintain a strictly controlled comparison, all evaluated variants still retain both $\mathbf{B}_k$ and the input moment in their local optimization objective.

Table 13: Ablation of shared-message on personalized GLUE. We report validation accuracy (%) over 500 rounds.

α	Shared message	RTE	SST-2	QNLI	MNLI-m	MNLI-mm	QQP	Avg.
0.5	$\mathbf{A}_k$	74.73	81.42	70.49	48.04	49.14	69.11	65.49
	$\mathbf{\Delta}_k$	74.37	81.65	70.55	48.13	49.15	68.56	65.40
	$\mathbf{A}_k\widehat{\mathbf{\Sigma}}_k^{1/2}$	**85.20**	95.30	93.25	87.45	87.39	**86.01**	89.10
	$\widehat{\mathbf{G}}_k$	**85.20**	**95.53**	**93.37**	**87.56**	**87.55**	85.92	**89.19**
0.75	$\mathbf{A}_k$	63.54	76.72	69.25	62.65	63.10	68.35	67.27
	$\mathbf{\Delta}_k$	61.73	78.67	69.34	62.69	63.13	68.87	67.40
	$\mathbf{A}_k\widehat{\mathbf{\Sigma}}_k^{1/2}$	**83.39**	95.41	93.04	89.04	89.25	85.35	89.25
	$\widehat{\mathbf{G}}_k$	83.03	**95.53**	**93.26**	**89.06**	**89.37**	**85.67**	**89.32**
1.0	$\mathbf{A}_k$	70.40	72.25	70.25	59.36	59.43	71.95	67.27
	$\mathbf{\Delta}_k$	70.04	72.48	70.14	59.36	59.43	71.99	67.24
	$\mathbf{A}_k\widehat{\mathbf{\Sigma}}_k^{1/2}$	77.98	95.99	**93.06**	88.51	**88.52**	86.40	88.41
	$\widehat{\mathbf{G}}_k$	**78.34**	**96.44**	92.84	**88.65**	88.44	**86.63**	**88.56**

E SHARED AND LOCAL RANK SENSITIVITY ON PERSONALIZED GLUE

A defining architectural advantage of FedSAIL over traditional parameter-averaging methods is its ability to decouple the communication payload from the client's local adapter capacity. In this section, we provide the detailed communication accounting and empirical sensitivity analysis for the shared rank (s) and local rank (r_{L}).

Communication accounting. At each adapted module, a client uploads an orthonormal basis of size $1024 \times s$, s eigenvalues, and a single integer token count, and subsequently receives a $1024 \times s$ global consensus basis. Assuming float32 precision for bases and eigenvalues and int32 for the count, the total bidirectional payload per run is:

$$C_{\mathrm{total}}(s) = 3 \text{ (clients)} \times 48 \text{ (modules)} \times 450 \text{ (rounds)} \times 4\,(2049s + 1) \quad \text{bytes.} \tag{9}$$

Each client has $98{,}304r_{\mathrm{L}}$ trainable LoRA parameters across all 24 layers. Under FedSAIL, the communication payload (Equation 9) scales exclusively with s and remains completely independent of r_{L}. For a fixed $s = 1$, the total communication is 0.4949 GiB, strictly governing the network overhead regardless of how large the local adapters grow.

Experimental setup. To empirically evaluate this decoupling, we conduct two complementary sweeps around the default configuration of $(r_{\mathrm{L}}, s) = (8, 1)$ following the protocol before.

1. **Shared-rank sweep**: We vary $s \in \{1, 2, 4, 8\}$ while fixing $r_{\mathrm{L}} = 8$ to examine the communication-accuracy trade-off.
2. **Local-rank sweep**: We vary $r_{\mathrm{L}} \in \{2, 4, 8, 16\}$ while fixing $s = 1$ to examine local capacity scaling under a strict communication budget.

To maintain isolation between variables, we fix the effective unscaled-factor penalty coefficient at $\beta_{\text{eff}} = 4$ by setting the LoRA scaling factor $\alpha_{\mathrm{LoRA}} = 2r_{\mathrm{L}}$. Neither the learning rate nor the penalty weight is retuned across either sweep. Tables 14 and 15 report the personalized validation accuracy over 500 rounds. Replicated entries use seeds 123, 124, and 125; we compute the six-split mean within each seed before calculating its sample standard deviation across seeds.

Table 14: Shared-rank sensitivity with fixed local rank $r_{\mathrm{L}} = 8$. We report personalized validation accuracy and total communication cost, with accuracy averaged across the six splits.

α	s	RTE	SST-2	QNLI	MNLI-m	MNLI-mm	QQP	Avg.	Comm. (GiB)
0.5	1	$80.87_{\pm 7.50}$	$96.37_{\pm 1.17}$	$93.06_{\pm 0.57}$	$88.98_{\pm 1.54}$	$89.07_{\pm 1.75}$	$89.80_{\pm 5.39}$	$89.69_{\pm 1.28}$	0.4949
	2	$80.39_{\pm 5.53}$	$96.56_{\pm 1.30}$	$93.56_{\pm 0.57}$	$89.12_{\pm 1.60}$	$89.23_{\pm 1.80}$	$90.11_{\pm 5.34}$	$89.83_{\pm 1.08}$	0.9895
	4	$80.51_{\pm 7.82}$	$96.64_{\pm 1.32}$	$93.40_{\pm 0.31}$	$89.37_{\pm 1.50}$	$89.45_{\pm 1.81}$	$90.07_{\pm 5.39}$	$89.91_{\pm 1.48}$	1.9787
	8	$80.75_{\pm 8.08}$	$96.60_{\pm 1.30}$	$93.37_{\pm 0.59}$	$89.34_{\pm 1.53}$	$89.41_{\pm 1.80}$	$90.19_{\pm 5.44}$	$89.94_{\pm 1.50}$	3.9573
0.75	1	$77.26_{\pm 5.05}$	$95.64_{\pm 0.20}$	$93.23_{\pm 0.61}$	$88.61_{\pm 1.53}$	$88.84_{\pm 1.60}$	$89.05_{\pm 4.69}$	$88.77_{\pm 0.56}$	0.4949
	2	$76.90_{\pm 5.32}$	$95.80_{\pm 0.52}$	$93.84_{\pm 1.04}$	$88.72_{\pm 1.49}$	$88.85_{\pm 1.60}$	$89.10_{\pm 4.79}$	$88.87_{\pm 0.49}$	0.9895
	4	$77.86_{\pm 3.62}$	$95.68_{\pm 0.46}$	$93.78_{\pm 1.06}$	$89.04_{\pm 1.44}$	$89.12_{\pm 1.55}$	$89.43_{\pm 4.56}$	$89.15_{\pm 0.18}$	1.9787
	8	$77.86_{\pm 3.71}$	$95.68_{\pm 0.18}$	$93.65_{\pm 1.32}$	$89.12_{\pm 1.54}$	$89.10_{\pm 1.66}$	$89.40_{\pm 4.60}$	$89.13_{\pm 0.18}$	3.9573
1.0	1	$81.95_{\pm 3.61}$	$96.25_{\pm 1.10}$	$93.28_{\pm 0.41}$	$88.40_{\pm 1.40}$	$88.52_{\pm 1.35}$	$88.72_{\pm 2.11}$	$89.52_{\pm 1.14}$	0.4949
	2	$81.71_{\pm 2.71}$	$96.29_{\pm 1.09}$	$93.87_{\pm 0.60}$	$88.47_{\pm 1.29}$	$88.70_{\pm 1.49}$	$88.87_{\pm 2.13}$	$89.65_{\pm 0.98}$	0.9895
	4	$80.63_{\pm 2.71}$	$96.02_{\pm 0.81}$	$93.79_{\pm 0.61}$	$88.67_{\pm 1.31}$	$88.75_{\pm 1.50}$	$88.96_{\pm 2.11}$	$89.47_{\pm 1.03}$	1.9787
	8	$80.75_{\pm 2.94}$	$96.06_{\pm 1.10}$	$93.81_{\pm 0.60}$	$88.64_{\pm 1.44}$	$88.78_{\pm 1.47}$	$88.84_{\pm 1.97}$	$89.48_{\pm 1.04}$	3.9573

Decoupling geometric alignment from local personalization. The results from the two sweeps illustrate the core mechanism of FedSAIL: structural alignment and task fitting utilize different forms of capacity. As shown in Table 14, increasing the shared rank s yields only small, non-monotonic changes in accuracy. Expanding s from 1 to 4 quadruples the communication payload (from 0.49 GiB to 1.98 GiB) but results in heavily diminishing returns in predictive performance. This pattern is qualitatively consistent with our diagnostic findings in Section 2.2, where cross-client agreement is concentrated in the leading direction. These results support $s = 1$ as a communication-efficient

default, while larger shared ranks offer modest, task-dependent gains at increased communication cost.

Table 15: Local-rank sensitivity with fixed shared rank $s = 1$. We report six-split validation accuracy (%, mean $\pm$ sample std.) and per-client LoRA parameter counts. Total communication remains fixed at 0.4949 GiB per run.

r_{L}	Params. (M)	$\alpha = 0.5$	$\alpha = 0.75$	$\alpha = 1.0$
2	0.197	$88.74_{\pm 1.43}$	$87.92_{\pm 0.55}$	$88.65_{\pm 1.41}$
4	0.393	$89.21_{\pm 1.51}$	$88.24_{\pm 0.81}$	$89.10_{\pm 1.09}$
8	0.786	$89.69_{\pm 1.28}$	$88.77_{\pm 0.56}$	$89.52_{\pm 1.14}$
16	1.573	$89.84_{\pm 1.43}$	$89.22_{\pm 0.29}$	$89.61_{\pm 0.95}$

Conversely, Table 15 shows that additional local capacity can improve accuracy without increasing the message size, although the gains depend on the heterogeneity setting. Clients retain the necessary degrees of freedom ($r_{\mathrm{L}} > s$) to model task-specific residuals, while collaborating strictly within a low-dimensional ($s = 1$) consensus geometry.

F Real-DDDT Layer Selection and Communication

F.1 Benchmark and Training Protocol

To evaluate layer selection, we construct two disjoint client replicas for each of the following datasets: IMDb (sentiment), Amazon (polarity), QQP (paraphrase), and QNLI (entailment). Within each replica, IMDb is paired with QQP, and Amazon with QNLI, yielding four disjoint different-domain/different-task dyads. Guidance-basis aggregation is strictly restricted to each dyad, while both LoRA factors remain entirely local. The geometric selector pools their subspace alignments to compute one common layer-sharing mask per training seed. Table 16 details the dataset allocation and hyperparameter configurations.

Table 16: DDDT data, training, and evaluation protocol.

Component	Setting
Clients and splits	Eight clients; per client: 1000 training, 256 calibration, 244 validation, and 500 final-test examples. All splits are label-balanced and disjoint across splits and replicas.
Data sources	Training/calibration: official training pools. Validation/final test: IMDb/Amazon test pools and labeled QQP/QNLI validation pools; no GLUE test-server evaluation.
Seeds	Data partition: 2027. Training seeds: $83, 97, 109$.
Backbone and adapters	Frozen RoBERTa-large and MLM head; query/value LoRA in all 24 layers, $r_{\mathrm{L}} = 4$, $\alpha_{\mathrm{LoRA}} = 16$, dropout 0.05. Independent $\mathbf{A}_k$ initialization and $\mathbf{B}_k = 0$.
Optimization	AdamW, learning rate 2×10^{-4}, weight decay 0, gradient clipping 1, batch size 16, CUDA AMP; 10 local iterations per client per round. Optimizer/scaler states remain local across rounds.
Training schedule	64 rounds: 32 local warmup rounds and 32 guided rounds, with shared rank $s = 1$ and guidance coefficient $\beta = 1$.
Inputs and prediction	Maximum 128 tokens, preserving instructions, field markers, decision mask, and content heads/tails. Mask-token verbalizers: negative/positive (sentiment), different/same (paraphrase), and yes/no (entailment).
Evaluation	Terminal round-64 checkpoint. Average accuracy over eight clients within each seed, then report mean and sample SD over three seeds. Task columns average two replicas and three seeds.

To prevent data leakage and ensure rigorous evaluation, review-body and question-identity deduplication are applied to control direct overlap. Equal final-test sizes are maintained to strictly align client-macro and sample-micro accuracy metrics; sentiment tasks contribute half the weight, while paraphrase and entailment each contribute a quarter.

F.2 WARMUP-BASED LAYER SELECTION

During local warmup ($\beta = 0$), we collect snapshots at rounds $t \in \{24, 28, 32\}$. For client k, layer $\ell \in \{0, \ldots, 23\}$, query/value projection $a \in \{Q, V\}$, and token regime $z \in \mathcal{Z} = \{\text{instruction}, \text{content}, \text{decision}\}$, let $\mathbf{q}_{k\ell azt}$ be a unit leading right singular vector of $\mathbf{G}_{k\ell azt} = \mathbf{B}^t_{k\ell a}\mathbf{A}^t_{k\ell a}\mathbf{\Sigma}^{1/2}_{k\ell zt}$. Here $\mathbf{\Sigma}_{k\ell zt}$ is the uncentered second moment of local calibration inputs in regime z at layer ℓ and round t, computed without shrinkage and with negative eigenvalues clipped to zero. For each of the four dyads $d \in \{1, 2, 3, 4\}$ described above, i_d and j_d denote its two client indices. To ensure stability against stochastic training noise, we define the layer agreement score s_ℓ using a temporal median over the collected snapshots:

$$s_\ell = \text{median}_{t\in\{24,28,32\}} \left[\frac{1}{24} \sum_{d=1}^{4} \sum_{a\in\{Q,V\}} \sum_{z\in\mathcal{Z}} \left| \mathbf{q}^\top_{i_d \ell azt} \mathbf{q}_{j_d \ell azt} \right|^2 \right], \qquad \mathcal{S}_\tau = \{\ell : s_\ell \geq \tau\}. \quad (10)$$

Each snapshot averages $4 \times 2 \times 3 = 24$ rank-one Projector Dice values. This threshold τ applies to the pooled score rather than individual dyads. Once the mask $\mathcal{S}_\tau$ is computed at round 32, it remains fixed. Notably, we impose no contiguity constraints or minimum layer requirements. For instance, at $\tau = 0.6$, the selector naturally excludes the final six layers in every random seed, consistently identifying that deeper layers encode representations too task-specific to benefit from federated sharing.

Following selection, all arms branch from identical round-32 optimizer and scaler states. Selected layers use the subspace penalty defined in (7) with $\beta = 1$, while unselected layers continue purely local adaptation using only the task loss.

F.3 THRESHOLD SENSITIVITY

Table 17 compares full-layer guidance against four different thresholds (τ) applied to the exact same warmup geometry. Increasing τ imposes a stricter geometric consensus requirement, restricting sharing to layers with robust cross-client agreement.

Table 17: FedSAIL threshold sensitivity on DDDT. We report round-64 accuracy (%, mean ± sample std.) and total communication cost, including selector snapshots and mask delivery.

FedSAIL variant	τ	IMDb	Amazon	QQP	QNLI	Avg.	Comm.
Full	–	93.33	95.93	**83.27**	87.60	$90.033_{\pm 0.225}$	96.094
Selection	0.4	93.20	95.70	83.07	88.07	$90.008_{\pm 0.644}$	104.282
Selection	0.5	93.23	96.17	82.37	**88.37**	$90.033_{\pm 0.284}$	89.601
Selection	0.6	93.57	96.37	83.13	87.47	$\mathbf{90.133}_{\pm 0.038}$	77.589
Selection	0.7	**93.77**	**96.40**	81.97	87.50	$89.908_{\pm 0.800}$	**58.904**

The sensitivity sweep clearly demonstrates the efficacy of geometry-guided selection. At $\tau = 0.6$, restricting guidance exclusively to the selected layers reduces the total communication payload by 19.26% while retaining predictive performance entirely comparable to full-layer guidance (a minimal difference of $+0.100$ percentage points, with a 95% t interval of $[-0.530, 0.730]$). This validates our core hypothesis: layers with divergent action subspaces should be left entirely local, freeing them from negative transfer (unhelpful cross-client penalties) while saving network bandwidth.

Furthermore, this sweep serves as a comprehensive sensitivity analysis for our system. We observe a clear trade-off surface: while $\tau = 0.7$ minimizes communication maximally, it begins to degrade mean accuracy (particularly on QQP) due to overly aggressive isolation. Conversely, a highly permissive threshold like $\tau = 0.4$ retains so many layers that the overhead of the selector mechanism outpaces the communication saved.

F.4 COMMUNICATION ACCOUNTING

Using the per-module payload convention in Appendix E, each active query/value module at $s = 1$ communicates $4(2049 + 1)$ bytes per client per guided round. For this DDDT protocol, eight

clients, two modules per layer, and 32 guided rounds give $4{,}198{,}400L$ bytes for L active layers. The additional selector cost consists of three snapshots over all 24 layers and three token regimes, $3 \times 8 \times 24 \times 2 \times 3 \times 1026 \times 4 = 14{,}183{,}424$ bytes, plus 24 bytes to deliver a 24-bit mask to each of the eight clients. Thus,

$$C_{\text{full}} = 4{,}198{,}400 \times 24, \qquad C_{\text{selection}}(L) = 4{,}198{,}400L + 14{,}183{,}448 \quad \text{bytes.} \tag{11}$$

Full-layer FedSAIL requires no selector. At this training length, selection must retain at most 20 layers to offset its overhead. Table 17 reports the mean per-run cost over three seeds in MiB (2^{20} bytes), charging the full selector cost to each threshold without amortization across the sweep.

These are logical payloads counted for every client delivery, excluding initial model distribution, headers, serialization, checkpoints, optimizer states, and evaluation traffic. The reported deployment cost includes the three snapshots required by the selection rule. The research pilot recorded five additional diagnostic snapshots; charging all eight adds 22.544 MiB per selection run. Under this research-logging allocation, $\tau = 0.6$ costs 100.133 MiB rather than 77.589 MiB and does not save communication relative to full-layer FedSAIL. The savings reported in the main text therefore refer to the three-snapshot selection protocol.

G Code Availability and Computational Resources

Computational resources. Table 18 summarizes the primary compute server and benchmark training environment. GLUE and DDDT runs simulate federated clients within single-GPU jobs; independent jobs are scheduled across the available GPUs. Both protocols use CUDA automatic mixed precision, with training settings given in Appendices C and F.

Table 18: Primary compute server and benchmark training environment.

Component	Specification
GPUs	4× NVIDIA GeForce RTX 3090, 24 GiB memory each
CPUs	2× Intel Xeon Gold 5218R, 40 physical cores in total
Host memory	Approximately 504 GiB (OS-reported)
Operating system	Ubuntu 22.04.5 LTS
Training runtime	Python 3.9.21; PyTorch 2.5.1 with CUDA 12.4
Libraries	Transformers 4.42.3; Datasets 2.20.0

Anonymous code release. The core implementation of FedSAIL is available at

https://anonymous.4open.science/r/FedSAIL-4A7C/.

The repository implements input-moment estimation and shrinkage, compact spectral messages, server-side subspace aggregation, the local action residual penalty, layer selection, and communication accounting. Synthetic CPU tests check these components against dense reference computations, including reparameterization invariance and penalty gradients. Experimental settings and evaluation conventions are documented in the preceding appendices.

Installation and verification. The standalone package requires Python 3.10 or later and PyTorch. From the repository root, the following commands install the package and run its numerical tests without downloading models or datasets:

```
python -m pip install -e ".[test]"
PYTHONDONTWRITEBYTECODE=1 python -m pytest -p no:cacheprovider
```