GRAFT: Cross-Backbone LoRA Reuse for Continual Language and Multimodal Learning

Ha Pham Thien Dinh
Deakin University
s.dinh@deakin.edu.au

Ming Liu
Deakin University
m.liu@deakin.edu.au

Jiesi Guo
Australian Catholic University
jiesi.guo@acu.edu.au

Matteo Vergani
Deakin University
matteo.vergani@deakin.edu.au

Kewen Liao
Deakin University
kewen.liao@deakin.edu.au

Abstract

In continual learning, parameter-efficient fine-tuning enables large language models to acquire new capabilities through lightweight adapters such as LoRA, yet the reuse of pretrained adapters is typically restricted to models sharing the same backbone. This limits the transfer of learned capabilities from language models to multimodal models. We propose **GRAFT** (Generalist Reuse of Adapters via Feature-space Transfer), a framework for cross-backbone LoRA reuse. GRAFT retrieves the most relevant pretrained LoRA adapter from a modality-aware library and maps it into the target backbone through a learned linear bridge between corresponding source and target feature spaces. We evaluate GRAFT on ten Super-NaturalInstructions text tasks, transferring Qwen2.5 LoRAs to Qwen2.5-VL, and five multimodal tasks, where Qwen2.5-VL LoRAs are directly reused. Compared with randomly initialized LoRA adapters, GRAFT improves performance on most evaluated tasks. These results show that GRAFT provides a simple and effective mechanism for reusing learned capabilities across language and multimodal models in continual learning.

1 Introduction

Large language models (LLMs) and vision-language models (VLMs) increasingly need to acquire new capabilities over time as tasks, domains, and modalities evolve (Shi et al., 2025). Parameter-efficient fine-tuning (PEFT), particularly Low-Rank Adaptation (LoRA) (Hu et al., 2022), provides an attractive mechanism for continual learning by adapting only a small number of parameters while keeping the backbone frozen. New capabilities can therefore be stored as lightweight adapters rather than separate copies of the full model, reducing both training and deployment costs. The growing availability of pretrained LoRA adapters creates an additional opportunity for continual learning: instead of learning each new adapter from random initialization, previously trained adapters can provide useful priors for subsequent tasks. AdaEvol (Liu et al., 2026) introduced this idea on LLMs by retrieving relevant pretrained LoRAs for new-task adaptation, showing that knowledge encoded in existing adapters can remain useful after further supervised fine-tuning. However, such reuse is typically restricted to adapters trained on the same LLM backbone. This substantially limits the knowledge available to a continual multimodal learner (e.g., VLMs) when relevant capabilities have been learned by a closely related but different model (e.g., LLMs).

We study this problem using Qwen2.5-7B-Instruct (Qwen et al., 2025) as a source language model and Qwen2.5-VL-7B-Instruct (Bai et al., 2025) as the target multimodal model. The two models belong to the same family and contain closely related language components, making them a natural testbed for cross-backbone adapter reuse. Many text-side capabilities, including reasoning, question answering, and domain-specific knowledge, may already be encoded in Qwen2.5 LoRAs even when equivalent Qwen2.5-VL adapters are unavailable. Reusing these adapters would allow a multimodal continual learner to access a broader pool of pretrained capabilities while consolidating them into a single Qwen2.5-VL backbone.

Cross-backbone reuse is nevertheless non-trivial. A LoRA adapter is optimized in the feature space induced by its source backbone, and directly attaching it to a different backbone does not preserve the activation environment in which its parameters were learned (Wang et al., 2024b). We therefore propose **GRAFT** (Generalist Reuse of Adapters via Feature-space Transfer), a cross-backbone warm-start framework for multimodal continual learning. GRAFT retrieves a relevant pretrained LoRA from a modality-aware LoRA library and estimates a lightweight linear compatibility bridge

between corresponding source and target feature spaces. This enables a generalist continual learner to reuse both text-only LoRAs from Qwen2.5 and multimodal LoRAs from Qwen2.5-VL. For cross-backbone tasks, retrieved Qwen2.5 LoRAs are transferred through GRAFT and fine-tuned into Qwen2.5-VL-compatible adapters; for same-backbone tasks, Qwen2.5-VL LoRAs are reused directly. In both cases, the resulting LoRA adapters are retained for future retrieval, allowing the library to grow over time while deployment remains centered on a single multimodal backbone. Our main contributions are as follows:

- We introduce GRAFT, a single-backbone multimodal continual learning framework that reuses both text-only and multimodal LoRA libraries through modality-aware retrieval, enabling a Qwen2.5-VL learner to access a broader pool of previously acquired capabilities while keeping deployment centered on a single multimodal backbone.
- We propose a lightweight cross-backbone warm-start mechanism in GRAFT that learns an activation-space compatibility bridge between Qwen2.5 and Qwen2.5-VL and applies it to the retrieved LoRA adapter's A matrix. The bridge is folded into the adapted LoRA after training, yielding a standard Qwen2.5-VL adapter with no inference-time overhead.
- We evaluate cross-backbone adapter reuse on ten Super-NaturalInstructions text tasks and same-backbone reuse on five multimodal tasks. GRAFT consistently outperforms adaptation from randomly initialized Qwen2.5-VL LoRAs, demonstrating its superiority of adapter reuse for continual capability acquisition across language and multimodal settings.

2 Related Work

2.1 Parameter-Efficient Continual Learning

LoRA represents a task update as a low-rank correction to frozen pretrained weights, so adapters can be stored and exchanged without a full copy of the model (Hu et al., 2022). Later variants refine how that budget is spent (Zhang et al., 2023b; Kopiczko et al., 2024; Liu et al., 2024) or reduce the cost of training them (Dettmers et al., 2023; Hayou et al., 2024; Zhang et al., 2023a). Closest to our own concern is PiSSA (Meng et al., 2024), which builds the trainable matrices from the pretrained weight's principal singular directions rather than at random, showing that a stronger starting point already exists in the backbone.

Continual-learning methods also isolate task-specific changes to protect earlier capabilities, through a separate adapter or prompt per task (Madotto et al., 2021; Razdaibiedina et al., 2023), approximately orthogonal low-rank updates (Wang et al., 2023), or blocks coordinated by shared attention (Zhao et al., 2024). These works treat adapters as persistent, reusable units of knowledge, but construct and reuse them around a single backbone.

2.2 Adapter Selection, Routing, and Composition

A second line treats trained adapters as a pool to select from or compose. AdapterFusion trains task-specific adapters independently, then learns how their representations contribute to a target task (Pfeiffer et al., 2021), and LoraHub extends this to LoRA libraries by optimizing combination weights from few examples (Huang et al., 2024). Dynamic variants gate those contributions or treat LoRAs as hierarchically gated experts (Wang et al., 2024a; Wu et al., 2024), others route among adapters as tasks arrive (Araujo et al., 2024), and AdaEvol retrieves pretrained LoRAs to initialise adaptation on the same backbone (Liu et al., 2026). All of these assume a shared backbone; we ask how such a pool can span two.

2.3 Cross-Model Adapter Transfer

Closest to our setting are works that transfer adapters across model backbones. Trans-LoRA distills a source LoRA into a newly initialized target adapter using synthetic target-task data filtered by a discriminator (Wang et al., 2024b). LoRA-X instead constrains adapters during training to the source backbone's singular subspace and projects them into compatible target subspaces (Farhadzadeh et al., 2025). Following either method requires inputs our setting does not provide. LoRA-X transfers only adapters trained under its own subspace constraint; imposing that constraint on a retrieved adapter would mean retraining it on the corpus it was originally built from, and several of those corpora were never released. Trans-LoRA instead substitutes synthetic data for target-task data where none is available, whereas each of our tasks supplies its own. We therefore use an ordinary pretrained LoRA as an initialization prior.

3 Methodology

In this section, we describe and formulate GRAFT, a generalist continual-learning framework that retrieves reusable LoRAs from text and multimodal adapter libraries and consolidates their capabilities into Qwen2.5-VL-7B-Instruct. When the retrieved adapter originates from Qwen2.5-7B-Instruct, GRAFT applies a cross-backbone warm-start mechanism based on an activation-space bridge; adapters already trained for Qwen2.5-VL are used directly. The bridged adapter is then optimized on the incoming task, after which the bridge is folded into the LoRA A matrix to produce a standard Qwen2.5-VL LoRA. Figure 1 gives an overview of the complete framework.

3.1 Problem Formulation

We consider a sequence of tasks $\{\mathcal{T}_1, \ldots, \mathcal{T}_K\}$ arriving over time, where each task $\mathcal{T}_t$ provides training data $\mathcal{D}_t$. The target model is Qwen2.5-VL-7B-Instruct, denoted by M_T, whose backbone remains frozen during parameter-efficient adaptation. For a target weight W_T, LoRA introduces a trainable low-rank update

$$W_T' = W_T + \Delta W, \qquad \Delta W = BA, \quad (1)$$

where $A \in \mathbb{R}^{r \times d_{\text{in}}}$ and $B \in \mathbb{R}^{d_{\text{out}} \times r}$ are the trainable LoRA parameters.

Alongside adapters previously trained on M_T, we consider pretrained LoRAs originating from the related Qwen2.5-7B-Instruct source model M_S. Such an adapter ΔW_S was trained inside M_S, and everything it learned was learned against the activations that model produces.

That is what makes reuse across the two backbones non-trivial. A LoRA's contribution to a module's output, $xA^\top B^\top$, is a function of the activations x that the module receives, and those activations belong to the backbone rather than to the adapter. Continued multimodal training has moved them, so the same adapter attached to M_T reads inputs unlike any it was fitted on, and the correction it applies is no longer the correction it learned. We therefore locate cross-backbone compatibility in the feature spaces the two models produce, and build the bridge there: for each module we estimate, by least squares on paired activations, the linear map that carries target-backbone features back into the source backbone's feature space.

Our objective is not to make the source LoRA directly transferable without additional training. Given ΔW_S and the incoming task data $\mathcal{D}_t$, we use the pretrained adapter as an informative initialization and optimise it into a LoRA ΔW_t^* for Qwen2.5-VL, so that the final adapter keeps the benefit of pretrained knowledge while operating entirely on the Qwen2.5-VL backbone.

3.2 Activation-Space Compatibility Bridge

Qwen2.5 and Qwen2.5-VL belong to the same model family and contain closely corresponding language layers, so a module's input activations in the two backbones are in principle comparable. Figure 2 illustrates offline bridge fitting, LoRA training with the bridge fixed, and folding for deployment. We seek a linear map that carries target-backbone features back into the source backbone's feature space, so that the source adapter's read-out map remains functionally valid.

Let $X_S, X_T \in \mathbb{R}^{n \times d_{\text{in}}}$ collect the input activations of a given module when the *same* token sequences are pushed through M_S and M_T respectively. We seek a transfer matrix T under which the adapter's output change is preserved,

$$X_T T \approx X_S, \quad (2)$$

which we obtain in closed form as a ridge regression with coefficient λ,

$$T = \left(X_T^\top X_T + \lambda I\right)^{-1} X_T^\top X_S. \quad (3)$$

T is computed offline from a separate dataset, external to the task sequence, that contains no test item; details are given in Appendix C.2.

To approximate $X_S \Delta W_S^\top$ on target features, we use $\Delta W_T = \Delta W_S T^\top$. Substituting the factored form $\Delta W = BA$ shows that the correction applies to the LoRA A matrix:

$$A' = AT^\top, \qquad B' = B, \qquad s' = s, \quad (4)$$

where s is the adapter's scaling factor. The LoRA B matrix and the scaling are left unchanged, while the rank budget is preserved and the update remains represented by two low-rank matrices. The correction is applied only to the LoRA A matrix because A is the matrix that reads the backbone activations, where the discrepancy between the two models arises.

The activation map addresses a different question from a correction based on backbone weight

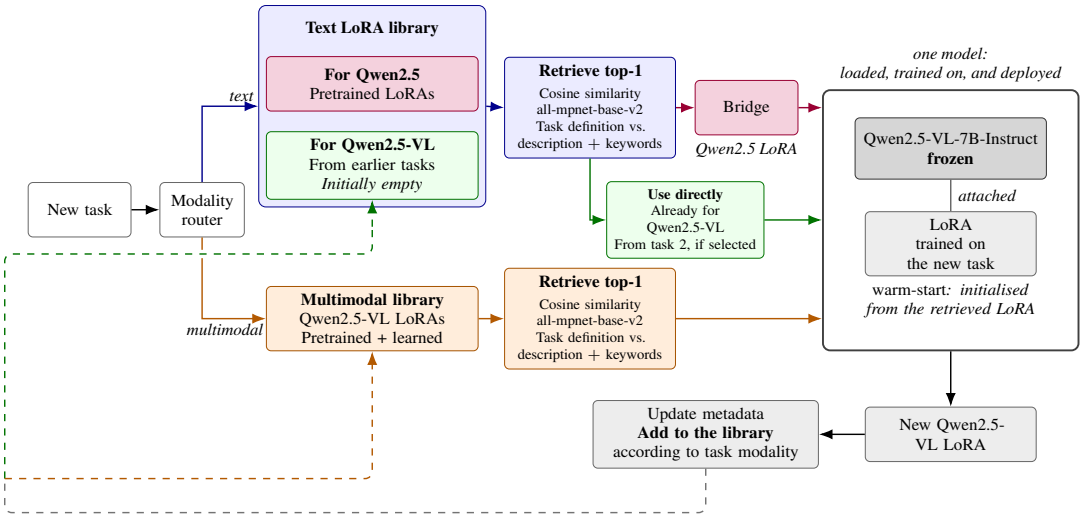

Figure 1: GRAFT pipeline. A task retrieves one LoRA from its modality's library. The text library separates pretrained Qwen2.5 LoRAs (purple, bridge required) from Qwen2.5-VL LoRAs learned on earlier tasks (green, used directly). The latter group is initially empty and can be used from task 2 onward, if selected. Multimodal LoRAs already work with Qwen2.5-VL. After training, metadata is updated and the LoRA is added to the appropriate library. Solid arrows show task processing; dashed arrows show library additions. The bridge is detailed in Figure 2.

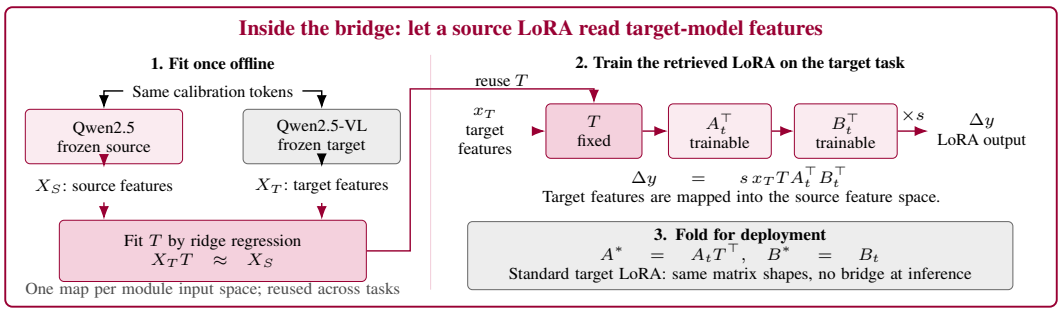

Figure 2: Activation-space bridge. Paired activations from the frozen backbones define T once offline. During target-task adaptation, T stays fixed while the LoRA A and B matrices train. Folding T into the trained A matrix produces a Qwen2.5-VL LoRA with no separate bridge at inference. Only offline calibration needs the source model.

differences. Weight differences can identify where the models have changed, and can be used to discount an update or project out selected directions. They do not directly estimate how the features presented to an adapter correspond across models. The bridge instead fits that correspondence on paired activations: it can mix input coordinates so that the source adapter reads a transformed version of the target features. Its objective is to recover the source read-out behaviour before task-specific optimisation, not simply to reduce the magnitude of a transplanted update. This remains an approximation whose quality depends on how well a linear map describes the two feature spaces; fitting a square matrix does not guarantee invertibility or exact preservation of the source model's behaviour.

Within a decoder layer several projections read the same tensor: the query, key and value projections share one input, and the two MLP input projections share another. There are therefore fewer distinct input spaces than target modules, and a single T is solved per space and shared by the modules that read it. We omit the MLP output projection, whose input dimension makes its transfer matrix disproportionately expensive to store, so the bridge covers six of the seven LoRA-target projections; modules without a transfer matrix pass through unchanged.

The transfer matrix depends only on the two backbones and the calibration data, never on any particular adapter. It is therefore computed once, offline, and reused for every source LoRA transferred between the same pair of models: it represents model-level compatibility rather than a trans-

formation learned per adapter. Because the fit is a regression, its residual on held-out sequences also serves as a direct, training-free test of whether the two feature spaces are linearly relatable, which we assess by comparing the bridged map against the uncorrected identity map.

3.3 Warm-Start Adaptation

For an incoming task, the retrieved Qwen2.5 LoRA supplies the initial parameters A_0 and B_0. The transfer matrices are retained in the forward computation throughout the single fine-tuning stage. At optimisation step t, the module's LoRA contribution is

$$x_T \longmapsto s\, x_T T A_t^\top B_t^\top, \tag{5}$$

where x_T comes from Qwen2.5-VL. The backbone and T are frozen, while both A_t and B_t are trained on the new task. The bridge supplies the feature mapping through which the retrieved matrices initially operate; supervision then adapts their response to the target model. Warm-starting therefore names where this training begins, rather than an additional training stage.

Keeping T active also distinguishes this training parameterisation from using $AT^\top$ as an initial value and then optimising that product freely. Here, optimisation updates A_t behind the same fixed map at every step. The adapter can change its response to the incoming task, while the correspondence estimated during calibration continues to define how target features enter it. This does not require the trained LoRA matrices to remain close to their source values or guarantee preservation of the original task.

Because T enters only through the LoRA A matrix, the effective update is

$$\Delta W_{\text{eff}} = s\, B_t \big(A_t T^\top\big). \tag{6}$$

The consecutive linear maps can be composed without storing a dense weight update. On completion of fine-tuning we fold the bridge into the LoRA A matrix and store

$$A^* = A_t T^\top, \qquad B^* = B_t. \tag{7}$$

Since T is square, A^* retains shape $r \times d_{\text{in}}$ and B^* retains shape $d_{\text{out}} \times r$. These are precisely the matrix shapes required by a standard LoRA attached to the corresponding Qwen2.5-VL module. Its output satisfies $s\, x_T (A^*)^\top (B^*)^\top = s\, x_T T A_t^\top B_t^\top$, so the saved adapter accepts target-model activations directly and computes the same update as the trained bridged adapter. The activation fit is approximate, but this folding step is algebraically exact and does not require T to be invertible.

This explains why the result is compatible with Qwen2.5-VL: its matrices have been optimised on that backbone, and its stored LoRA A matrix already includes the feature transformation used during training. The source model and a separate bridge are no longer needed at inference. Evaluation and any later warm-start can load the saved matrices as an ordinary Qwen2.5-VL LoRA; applying the bridge again would transform them twice. Each task keeps its own adapter and the shared backbone remains frozen, so subsequent adaptation cannot overwrite the parameters used to serve an earlier task.

3.4 Generalist Adapter Library

GRAFT maintains two libraries organised by task modality, $\mathcal{L} = \mathcal{L}_{\text{Text}} \cup \mathcal{L}_{\text{VL}}$. The public LoRAs used to initialise them vary in rank, scaling factor, dropout and target-module selection. The bridge operates on the A matrix an adapter provides without changing its shape; it requires no common rank or module configuration across the library. The collected adapters and their configurations are described in the appendix.

The text library contains the two groups shown in Figure 1. The first holds pretrained LoRAs *for Qwen2.5*; these need the bridge when attached to Qwen2.5-VL. The second holds LoRAs *for Qwen2.5-VL* learned on earlier text tasks. This group is initially empty and can first supply an adapter at task 2, provided an earlier text task has completed. The multimodal library contains pretrained and newly learned Qwen2.5-VL LoRAs, which can be used directly. Library membership thus identifies the kind of task an adapter can be retrieved for, while its recorded backbone compatibility determines whether it needs the bridge.

A modality router selects the appropriate library from a task's modality flag, or otherwise from whether its samples contain images. Routing is rule-based and deterministic. Within that library, a single retrieval step ranks all candidates. In particular, the two groups in the text library are ranked together: a Qwen2.5 adapter takes the bridge path, while a selected Qwen2.5-VL adapter takes the *Use directly* path. The latter is an option for later tasks, not an additional source of adapters available at the start of the sequence.

Each public adapter carries a description and key-

words generated once offline by prompting Claude Opus 5 with its model card. A fixed schema records the task, domain, inputs, outputs and capabilities. We encode the incoming task definition and both metadata fields with `all-mpnet-base-v2`, score each adapter by the equally weighted cosine similarities to its description and keywords, and retrieve the top-ranked LoRA. This requires neither loading candidate weights nor inspecting test examples. No loss-based reranking is used for the reported results.

After fine-tuning and any bridge folding, metadata is updated to describe the new task, identify the initialising adapter and record compatibility with Qwen2.5-VL. The trained LoRA is then *added to the library* for its task's modality. Text outputs enter the Qwen2.5-VL group, so any later retrieval uses them without the bridge. Updating the task description makes the newly learned capability visible to subsequent similarity searches. The library can therefore grow across a sequence while every completed task remains served by its own saved adapter.

4 Experiments

4.1 Datasets and Tasks

We evaluate GRAFT on both text-only and multimodal tasks, with all experiments using Qwen2.5-VL-7B-Instruct as the frozen target backbone. For text evaluation, we use ten tasks from Super-NaturalInstructions (Wang et al., 2022), covering entailment, sentiment analysis and verification, review rating prediction, emotion classification, and answerability. For multimodal evaluation, we use five tasks: four image classification tasks (aerial scene, fine-grained dog breed, document image and waste classification) and OCRVQA, an OCR-based visual question answering task. The tasks are not selected arbitrarily: each has a semantically related capability in our pretrained LoRA libraries, which hold 28 text and 22 multimodal adapters, so the evaluation tests cross-backbone warm-starting under the intended retrieval scenario. Full task and library details are given in the appendix. The bridge is computed offline from the task set of (Liu et al., 2026), before the task sequence begins.

Because the library grows, task order determines which earlier adapters are available for retrieval. We evaluate four orders fixed before running: text first, multimodal first, interleaved, and fully shuffled. Only the last changes the within-modality sequence. Appendix D lists the sequences, and Table 6 reports their results.

4.2 Baselines

LoRA-Random is compared against every warm-started configuration using the same target-task data and number of optimisation steps in both settings: it fine-tunes a newly initialised Qwen2.5-VL LoRA on each task. *GRAFT* instead initialises that adapter from a retrieved Qwen2.5 LoRA, keeps the bridge fixed during training, and folds it into the saved adapter. Because the multimodal library already holds Qwen2.5-VL adapters, a retrieved LoRA needs no bridge there; we call that configuration *LoRA-Warm*. GRAFT is therefore the configuration used on the ten text tasks, where the source and target backbones differ, and LoRA-Warm the one used on the five multimodal tasks, where they coincide. For the text setting we additionally run a third configuration, *No bridge*, as an ablation: it transplants the retrieved A and B into Qwen2.5-VL unchanged. All other factors (retrieval, the selected anchor, the data and the training budget) are kept fixed, so it differs from GRAFT only in the bridge. The two comparisons differ. The multimodal one matches rank, scaling and module coverage, so it isolates warm-starting on a single backbone rather than the bridge; the text one is the cross-backbone test. Modality routing means that a Qwen2.5 text adapter is never offered to an image-dependent task.

4.3 Evaluation

All ten text tasks are in the closed label setting, and four of the five multimodal tasks are as well; only OCRVQA has an open answer space and is thus measured by exact match. All tasks are reported with Accuracy, with the change in macro-F1 alongside it. Both task types use the same epoch protocol: both configurations are trained for eight epochs, and each is evaluated at its own best-validation epoch on a 200-instance split disjoint from the test set, so the baseline has the same opportunity to select its strongest checkpoint as the warm-started one. Every reported cell is the mean over all four orders; the per-order values are given in Table 6. Text and multimodal means are reported separately, with a pooled mean over all fifteen tasks. Each task is served by its own adapter, so we report per-task performance directly.

4.4 Implementation Details

Only LoRA parameters are updated; the Qwen2.5-VL backbone remains frozen. Both configurations receive the same eight-epoch budget, with checkpoint selection as described above. This measures adaptation under a finite training budget, rather than assuming convergence, and matches the intended setting of inexpensive adaptation as tasks arrive.

5 Experimental Results

Table 1 reports both task types. For text tasks, each adapter is initialised from a Qwen2.5 LoRA re-expressed through the activation-space bridge; mean accuracy rises from 82.2 to 83.3, a gain of 1.17 points (Wilcoxon signed-rank, $p = 0.02$; Table 3). For multimodal tasks, the retrieved adapters already work with Qwen2.5-VL and are used directly; mean accuracy rises from 76.2 to 78.3, a gain of 2.13 points. Table 2 isolates the bridge's share: transplanting the retrieved adapter without it reaches 83.00 against GRAFT's 83.66 on Order A, a bridge contribution of 0.66 points.

6 Discussion

Across the reported tasks, retrieved initialisation improves average accuracy by 1.17 points on text and 2.13 on multimodal tasks. The gains are uneven: TellMeWhy and OCRVQA benefit substantially, while Dog Breed is already close to the baseline's ceiling. The practical value of reuse therefore depends on both the retrieved capability and the remaining headroom under the training budget, rather than on the mere availability of a pretrained adapter. Two separate points follow. Table 2 shows GRAFT exceeding the ablated *No bridge* configuration in mean accuracy, 83.66 against 83.00, so the bridge adds to warm-starting on average. And the text gain over LoRA-Random in Table 3 is significant: 1.17 points, 95% interval $[0.11, 2.23]$, $p = 0.02$.

Sentiment140 Answer is the only task with a negative mean accuracy delta, falling by 0.95 points. Sentiment140 Polarity uses the same underlying tweet corpus and retrieves the same source adapter, yet gains 1.45 points on average. The contrast suggests that transfer depends on task formulation: direct polarity prediction benefits, while the yes/no task that verifies a supplied sentiment does not. LoRA-Random already reaches 91.2% on this task, leaving limited headroom within the eight-epoch budget. GRAFT performs best on Sentiment140 Answer under Order D (Table 6), the only schedule that changes its position within the text-task sequence (Appendix D); this is the strongest result observed within that budget and suggests that the effectiveness of the warm start is also sensitive to where the task occurs in the sequence. MNLI Neutral is another boundary case. LoRA-Random already reaches 93.55%, leaving limited headroom; the two configurations tie in mean accuracy, while macro-F1 differs by only -0.03 points.

Order D changes the sequence within each modality (Appendix D) and has the largest number of negative cells (Table 6). Amazon Rating and ANLI also change sign between orders, whereas TellMeWhy and UC Merced Aerial remain positive in all four. Orders A–C preserve the same within-modality sequence (Appendix D), so their interleaving does not expose a task to a different set of earlier same-modality adapters. The order comparison should accordingly be read as a stability check on the full pipeline rather than a causal estimate of which schedule is best.

These comparisons include numerical variation as well as changes in library state. Each configuration selects its checkpoint using validation accuracy, so small training differences can change the reported epoch.

Accuracy and macro-F1 never reverse direction on the closed-label tasks; MNLI Neutral's accuracy tie accompanies only a 0.03-point F1 loss. OCRVQA is the exception: its accuracy gain accompanies a macro-F1 loss, reflecting the different weights the metrics give to frequent answers and rare answer strings. Its benefit should therefore be read as improved exact-match accuracy, rather than a uniform improvement across answer types.

These results support adapting existing Qwen2.5 LoRAs on Qwen2.5-VL, without distillation or specially designed transferable adapters (Wang et al., 2024b; Farhadzadeh et al., 2025). Bridge folding removes inference overhead, while separate task adapters prevent later training from overwriting earlier parameters (Wang et al., 2023).

Shared calibration and task-specific adaptation serve different purposes. Calibration estimates a reusable relationship between the two backbones; task supervision determines which parts of a retrieved capability are useful for the new objective. Separate adapters retain task specialisations, while folding lets them share one target backbone. Adding a task therefore requires another lightweight adapter without retaining another

Task	Type	LoRA-Random	Warm-start	ΔAcc	ΔF1
ANLI Entailment	Text	72.9	**73.2**	0.25	0.41
Sentiment140 Polarity	Text	88.2	**89.7**	1.45	1.45
Sentiment140 Answer	Text	**91.2**	90.3	−0.95	−0.94
DVD Polarity	Text	95.4	**96.4**	1.05	1.05
Amazon Rating	Text	62.8	**64.5**	1.70	1.80
Amazon Food Rating	Text	72.4	**73.0**	0.55	0.95
Emotion	Text	90.8	**92.6**	1.80	1.77
MNLI Neutral	Text	93.6	93.6	0.00	−0.03
TellMeWhy	Text	79.0	**83.6**	4.60	2.76
DoQA Movies	Text	75.5	**76.7**	1.25	0.59
Text mean	Text	82.2	**83.3**	1.17	0.98
UC Merced Aerial	Multimodal	70.6	**73.0**	2.40	3.36
Dog Breed	Multimodal	97.9	**98.1**	0.20	0.18
RVL-CDIP	Multimodal	66.6	**67.5**	0.90	0.65
Waste Types	Multimodal	84.0	**85.2**	1.25	0.90
OCRVQA	Multimodal	61.8	**67.7**	5.90	−2.72
Multimodal mean	Multimodal	76.2	**78.3**	2.13	0.47
Overall mean	All	80.2	**81.7**	1.49	0.81

Table 1: Warm-start results across text and multimodal tasks. Text tasks use GRAFT: the top-ranked retrieved Qwen2.5-7B adapter is re-expressed for Qwen2.5-VL-7B through the bridge; multimodal tasks use the top-ranked Qwen2.5-VL-7B adapter directly. No adapters are merged. Both configurations train for eight epochs and are scored at their own best-validation epoch on 500 test items per task. Accuracy is a percentage; ΔAcc and ΔF1 are absolute differences in points, from unrounded scores. Every value is the mean over four task orders, text and multimodal means computed separately; per-order values are in Table 6. Class counts and training-set sizes are in the appendix.

Initialisation	Acc.
No bridge	83.00
GRAFT	**83.66**

Table 2: Bridge ablation. Mean accuracy on the ten text tasks of Order A; the two configurations differ only in the bridge.

ΔAcc	95% CI	p
1.17	[0.11, 2.23]	**0.02**

Table 3: Gain over LoRA-Random on the ten text tasks, averaged over the four orders, with an exact Wilcoxon signed-rank test and a t interval on the mean (Appendix C.1).

source model for inference.

Retrieval remains separated by modality, and evaluation covers one backbone pair. Qwen2.5-VL is built on Qwen2.5, so the two share a tokenizer and their language modules correspond one to one. The bridge needs both, because paired activations must come from the same token positions in matching modules. A different model family, such as Llama, satisfies neither by default: a tokenization alignment and a module correspondence would have to be established first, which we do not study. Ranking across both libraries and transfer between less closely related models remain open questions. The tasks were also chosen for capabilities the libraries already cover, and we vary task order rather than the random seed.

7 Conclusion

We presented GRAFT, a framework for reusing pretrained LoRA adapters across related language and multimodal backbones. GRAFT uses a learned linear bridge to map a retrieved source adapter into the target backbone before further adaptation. Experiments on ten text tasks and five multimodal tasks show that GRAFT consistently outperforms adaptation from randomly initialized LoRA adapters across most evaluated tasks. These results demonstrate the effectiveness of GRAFT's cross-backbone LoRA reuse mechanism for multimodal continual learning.

References

Vladimir Araujo, Marie-Francine Moens, and Tinne Tuytelaars. 2024. Learning to route for dynamic

adapter composition in continual learning with language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 687–696, Miami, Florida, USA. Association for Computational Linguistics.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025. Qwen2.5-vl technical report. *Preprint*, arXiv:2502.13923.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. In *Advances in Neural Information Processing Systems*, volume 36.

Farzad Farhadzadeh, Debasmit Das, Shubhankar Borse, and Fatih Porikli. 2025. Lora-x: Bridging foundation models with training-free cross-model adaptation. In *International Conference on Learning Representations*.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. 2024. Lora+: Efficient low rank adaptation of large models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 17783–17806. PMLR.

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

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. 2024. Lorahub: Efficient cross-task generalization via dynamic lora composition. In *Conference on Language Modeling*.

Dawid J. Kopiczko, Tijmen Blankevoort, and Yuki M. Asano. 2024. Vera: Vector-based random matrix adaptation. In *International Conference on Learning Representations*.

Ran Liu, Min Yu, Jianguo Jiang, Gang Li, Rongsheng Li, Mingqi Liu, Zhen Xu, Weiqing Huang, and Ming Liu. 2026. Adaevol: Dynamic adapter merging for effective continual learning and knowledge transfer in large language models. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 17237–17241. IEEE.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024. Dora: Weight-decomposed low-rank adaptation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 32100–32121. PMLR.

Andrea Madotto, Zhaojiang Lin, Zhenpeng Zhou, Seungwhan Moon, Paul Crook, Bing Liu, Zhou Yu, Eunjoon Cho, Pascale Fung, and Zhiguang Wang. 2021. Continual learning in task-oriented dialogue systems. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7452–7467, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. 2024. Pissa: Principal singular values and singular vectors adaptation of large language models. In *Advances in Neural Information Processing Systems*, volume 37.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online. Association for Computational Linguistics.

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, and 24 others. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. 2023. Progressive prompts: Continual learning for language models. In *International Conference on Learning Representations*.

Haizhou Shi, Zihao Xu, Hengyi Wang, Weiyi Qin, Wenyuan Wang, Yibin Wang, Zifeng Wang, Sayna Ebrahimi, and Hao Wang. 2025. Continual Learning of Large Language Models: A Comprehensive Survey. *ACM Computing Surveys*, 58(5).

Hanqing Wang, Bowen Ping, Shuo Wang, Xu Han, Yun Chen, Zhiyuan Liu, and Maosong Sun. 2024a. LoRA-flow: Dynamic LoRA fusion for large language models in generative tasks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12871–12882, Bangkok, Thailand. Association for Computational Linguistics.

Runqian Wang, Soumya Ghosh, David Cox, Diego Antognini, Aude Oliva, Rogerio Feris, and Leonid Karlinsky. 2024b. Trans-lora: Towards data-free transferable parameter efficient finetuning. In *Advances in Neural Information Processing Systems*, volume 37.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. 2023. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10658–10671, Singapore. Association for Computational Linguistics.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, and 16 others. 2022. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xun Wu, Shaohan Huang, and Furu Wei. 2024. Mixture of lora experts. In *International Conference on Learning Representations*.

Longteng Zhang, Lin Zhang, Shaohuai Shi, Xiaowen Chu, and Bo Li. 2023a. Lora-fa: Efficient and effective low rank representation fine-tuning. *arXiv preprint arXiv:2308.03303*.

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

Weixiang Zhao, Shilong Wang, Yulin Hu, Yanyan Zhao, Bing Qin, Xuanyu Zhang, Qing Yang, Dongliang Xu, and Wanxiang Che. 2024. SAPT: A shared attention framework for parameter-efficient continual learning of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11641–11661, Bangkok, Thailand. Association for Computational Linguistics.

Task	Super-NaturalInstructions identifier
ANLI Entailment	`task1385_anli_r1_entailment`, `task1386_anli_r2_entailment`, `task1387_anli_r3_entailment`
Amazon Rating	`task1310_amazonreview_rating_classification`
TellMeWhy	`task290_tellmewhy_question_answerability`
Sentiment140 Polarity	`task195_sentiment140_classification`
Sentiment140 Answer	`task196_sentiment140_answer_generation`
Amazon Food Rating	`task588_amazonfood_rating_classification`
Emotion	`task875_emotion_classification`
MNLI Neutral	`task201_mnli_neutral_classification`
DVD Polarity	`task477_cls_english_dvd_classification`
DoQA Movies	`task1442_doqa_movies_isanswerable`

Table 4: The ten Super-NaturalInstructions text tasks. All are evaluated with Accuracy. ANLI pools the three round-wise entailment tasks.

Task	Source dataset	$\|C\|$	|train|
UC Merced Aerial	UC Merced Land Use	21	1,400
Dog Breed	animime-dog-breeds	10	635
RVL-CDIP	RVL-CDIP (subset)	16	1,860
Waste Types	waste-classification	18	1,276
OCRVQA	OCR-VQA	n/a	3,000

Table 5: The five multimodal tasks. $|C|$ is the number of classes; OCRVQA has an open answer space and no fixed class list.

Appendix

A Dataset Details

Text tasks. All text tasks are drawn from Super-NaturalInstructions (Wang et al., 2022) and are fetched by task identifier at build time. For each task we cap training data at 3,000 instances, giving training sets of roughly 1,200 to 3,000 instances, evaluate on 500 held-out instances, and hold out a further 200 instances for epoch selection. Table 4 lists the identifier for each task. ANLI pools the three round-wise SNI entailment tasks into a single task, since the rounds share a label space and differ only in the adversarial round that produced them. The two Sentiment140 tasks share their tweet corpus but pose different outputs: one predicts polarity directly, while the other answers yes or no to a proposed polarity. Amazon Rating and Amazon Food Rating are two five-point rating tasks over different Amazon review collections; the remaining tasks draw on distinct source corpora.

Multimodal tasks. Four of the five multimodal tasks are built from publicly available image classification datasets, listed in Table 5; the fifth, OCRVQA, is an OCR-based visual question answering task with an open answer space rather than a fixed class list. As with the text tasks, training data is capped at 3,000 instances and evaluation uses 500 held-out instances. For the four classification tasks, each prompt presents the candidate class list, which is derived from the training split alone so that no evaluation-time label information enters the prompt.

B Adapter Library Composition

The multimodal pool comprises 22 adapters collected for Qwen2.5-VL-7B-Instruct, of which 20 are compatible with the generative backbone; the two exclusions are retrieval adapters whose multi-vector projection head has no counterpart in the generative model. The pool spans visual question answering, OCR and document conversion, aerial and driving scene understanding, visual and narrative reasoning, and multimodal quality evaluation.

Descriptions and keywords for all library adapters were generated in a single offline pass with Claude Opus 5, prompted with each adapter's HuggingFace model card. The schema is fixed: a one-paragraph description naming the task, the domain, the inputs, the outputs and four capabilities, followed by six to ten keywords. The generated text is used only to rank candidates against an incoming task definition; it never enters training or evaluation, so an inaccurate description can only cause a less suitable adapter to be retrieved, which works against the warm-started configuration rather than for it. We did not manually verify the generated fields against each adapter's configuration.

The text pool comprises 28 LoRAs trained on Qwen2.5-7B-Instruct, covering natural language inference, commonsense and multi-step reasoning, summarization, question answering, information extraction, sentiment and emotion classification, and domain-specific knowledge, principally medical.

The collected adapters are heterogeneous in configuration. Across the text pool, ranks span $r \in \{8, 16, 24, 32, 64, 128\}$ with $r = 16$ most common, LoRA coefficients α range from 8 to 128, dropout takes values in $\{0, 0.05, 0.075\}$, and target-module selections cover two, four, or all seven linear projections.

C Implementation Details

Adapters are optimized with AdamW at a learning rate of 3e-5 and no weight decay, with a per-step batch size of one and gradient accumulation over four steps. Gradient norms are clipped to 1.0 and sequences longer than 1,024 tokens are skipped.

All runs use seed 42 on a single NVIDIA A100 SXM4 40GB GPU.

The adapter configurations are as follows:

- *Text, LoRA-Random:* rank $r = 16$, LoRA coefficient $\alpha = 32$ (update scaling $s = \alpha/r = 2$), and all seven projection types: `q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, and `down_proj`. A is Kaiming-uniform and B is zero. Keeping this configuration fixed across text tasks provides a consistent reference for adaptation from scratch, without making the baseline's architecture depend on which adapter retrieval selects.

- *Text, GRAFT:* the retrieved adapter supplies the initial matrices, rank, scaling, and module set; the activation bridge is held fixed during training. Retaining this configuration allows reuse of existing adapters without an additional rank-conversion step or changes to their module coverage, either of which would modify the pretrained structure being reused. The text comparison therefore evaluates the complete reuse procedure against a fixed baseline for adaptation from scratch. Capacity is not matched, but the mismatch does not favour GRAFT. Rank coincides at $r = 16$ on nine of the ten tasks and the update scaling is $s = 2$ in both configurations; ANLI is the only task whose retrieved adapter is larger ($r = 24$). The retrieved adapters instead cover fewer module types: on seven tasks they adapt only the four attention projections, giving 10.1M adapter parameters against the baseline's 40.4M. Those seven tasks also show the largest mean gain, $+2.06$ points, against $+0.70$ on the two tasks where coverage coincides and $+1.00$ on ANLI, so the improvement is not explained by additional capacity. The GRAFT and no-bridge rows of Table 2 share rank, scaling, module set and anchor exactly, so that contrast isolates the bridge at matched capacity.

- *Multimodal, LoRA-Warm:* the retrieved adapter's pretrained matrices, rank, scaling, and module set are retained. It is used directly because it was trained on the same Qwen2.5-VL backbone.

- *Multimodal, LoRA-Random:* the retrieved adapter's matrix shapes, module set, and scaling are retained, but A is reinitialised with Kaiming-uniform values and B is set to zero. This matches capacity within each task so that the configurations differ only in the initial values of their LoRA A and B matrices.

For retrieval, candidates are ranked by cosine similarity between the encoded task definition and each adapter's encoded metadata, and the top-ranked adapter is used; no loss-based reranking is performed for any reported result, and the ranking is deterministic given the task description and current library contents.

Routing is rule-based and deterministic, with a fixed order of authority. An explicit modality declaration in the task file takes precedence; a declaration in the task registry is used next; failing both, up to 64 of the task's own samples are probed and the task is routed to the multimodal pool if any sample carries image content. A task registered under both modalities that declares neither is rejected rather than guessed. No learned classifier is involved, and the rule applied and the evidence for it are recorded for every task. The text library holds text adapters only, either pretrained Qwen2.5 LoRAs that take the bridge or Qwen2.5-VL text LoRAs warm-started on earlier text tasks, so a text task is never served an adapter from the multimodal library.

A returned adapter is registered with keywords naming the new task, and is then ranked by the same similarity score as the pretrained LoRAs.

C.1 Significance of the Text Result

Significance is assessed over tasks rather than over test items, because the task is the replicated unit: each task is served by its own adapter and scored on its own test split. Applied to the per-task ΔAcc values of Table 6, an exact two-sided Wilcoxon signed-rank test over the ten text tasks gives $p = 0.0195$ ($W^+ = 42$, $W^- = 3$; MNLI Neutral is tied at 0.00 and is dropped). A 95% interval on the mean gain, from the t distribution over the same ten values, is $[+0.11, +2.23]$, and is the interval reported in Table 3. Table 2 reports mean accuracy only; the difference between the two warm-started initialisations is not tested, since one order gives a single paired value per task. All reported numbers come from a single run of each configuration; significance is therefore assessed across tasks rather than across independent training runs.

Task	A	B	C	D	Mean
Text					
ANLI Entailment	+1.00	+1.80	−0.20	−1.60	+0.25
Sentiment140 Polarity	+2.00	+2.00	+2.00	−0.20	+1.45
Sentiment140 Answer	−1.20	−1.20	−1.20	−0.20	−0.95
DVD Polarity	+1.00	+1.20	+0.60	+1.40	+1.05
Amazon Rating	+5.00	−3.00	+1.00	+3.80	+1.70
Amazon Food Rating	+1.20	+1.20	+0.00	−0.20	+0.55
Emotion	+1.40	+1.40	+1.40	+3.00	+1.80
MNLI Neutral	+0.00	+0.00	−0.80	+0.80	+0.00
TellMeWhy	+4.60	+4.60	+4.60	+4.60	+4.60
DoQA Movies	+1.80	+0.20	+0.80	+2.20	+1.25
Multimodal					
UC Merced Aerial	+2.40	+2.40	+2.40	+2.40	+2.40
Dog Breed	+0.00	+0.20	+0.80	−0.20	+0.20
RVL-CDIP	+1.20	+2.00	+0.00	+0.40	+0.90
Waste Types	+2.20	+0.40	+0.80	+1.60	+1.25
OCRVQA	+6.00	+6.00	+6.00	+5.60	+5.90

Table 6: Change in accuracy, in points, under the four task orders listed in Appendix D. Each order includes the same ten text and five multimodal tasks. The Mean column averages A–D and matches ΔAcc in Table 1. Differences between orders include training and checkpoint-selection variation and do not isolate an effect of library contents.

C.2 Bridge Calibration

The calibration corpus is assembled round-robin from the training splits of a fixed set of six text tasks – SentimentTwitter, CommonsenseVal, ANLI, PubMedQA, ExtremeSumm and EnDeTranslation – held over from earlier single-backbone experiments and unchanged for every run reported here, after shuffling each split with seed 0. Prompts and responses are concatenated and truncated to 256 tokens. Identical token IDs are fed to both frozen backbones. We fit the bridge on 461 sequences (65,102 tokens) and reserve 51 further sequences for the held-out alignment diagnostic. For each input space, the ridge coefficient λ is 10^{-3} times the mean diagonal of $X_T^\top X_T$, with that mean floored at 10^{-6}. The systems are solved in float32 and the transfer matrices are cached in float16. Sharing a map among projections with the same input yields 84 matrices across 28 language layers, covering 168 modules. The `down_proj` modules have no transfer matrix and use their LoRA matrices without this transform. The cache is fitted once and shared across all source adapters. An activation-space map is fitted from paired activations, so a calibration corpus is required by construction. Like the frozen backbones, the map is estimated once before the sequence and is never updated by it: it belongs to the system's setup rather than to task learning. Calibration uses training splits only, with no test item, and the regression target is the source backbone's activations rather than the response, so no task supervision enters the map. One map serves every task and every adapter, so it cannot favour a particular one. The task set was fixed before these experiments and was not selected for them.

D Task Orders

The library grows as tasks arrive, so a task retrieves from a different pool depending on where it falls in the sequence. To evaluate performance across different task sequences, the benchmark is run in the four orders below, all fixed before any was run. Multimodal tasks are shown in *italics*.

Orders A, B and C share the same within-modality sequence and differ only in how the two modalities interleave.

Order A - all text, then all multimodal. Emotion → Sentiment140 Polarity → MNLI Neutral → TellMeWhy → Amazon Rating → Amazon Food Rating → ANLI Entailment → DVD Polarity → Sentiment140 Answer → DoQA Movies → *Dog Breed* → *OCRVQA* → *UC Merced Aerial* → *Waste Types* → *RVL-CDIP*

Order B - all multimodal, then all text. *Dog Breed* → *OCRVQA* → *UC Merced Aerial* → *Waste Types* → *RVL-CDIP* → Emotion → Sentiment140 Polarity → MNLI Neutral → TellMeWhy → Amazon Rating → Amazon Food Rating → ANLI Entailment → DVD Polarity → Sentiment140 Answer → DoQA Movies

Order C - interleaved, two text to one multimodal. Emotion → Sentiment140 Polarity → *Dog Breed* → MNLI Neutral → TellMeWhy → *OCRVQA* → Amazon Rating → Amazon Food Rating → *UC Merced Aerial* → ANLI Entailment → DVD Polarity → *Waste Types* → Sentiment140 Answer → DoQA Movies → *RVL-CDIP*

Order D - fully shuffled. This order shuffles all fifteen tasks together, changing both their within-modality positions and how text and multimodal tasks mix. It tests adaptation to a mixed task stream without modality blocks or a fixed alternating pattern.

Amazon Food Rating → *OCRVQA* → DVD Polarity → *Dog Breed* → DoQA Movies → TellMeWhy → ANLI Entailment → *RVL-CDIP* → Emotion → Sentiment140 Polarity → MNLI Neutral → *Waste Types* → Amazon Rating → *UC Merced Aerial* → Sentiment140 Answer

E Environment

All experiments were run on NVIDIA A100-SXM4 40 GB GPUs under PyTorch 2.11 (CUDA 12.8), Transformers 5.8.1 and Python 3.12. The seed, the train/validation/test splits, the task orders, the retrieval ranking and the initialisation of both configurations are fixed by the released setup.

Numerical results depend on the execution environment as well as on the configuration. We use PyTorch's fused scaled dot-product attention, which is considerably faster than the deterministic reference implementation at these sequence lengths; the fused kernels accumulate partial results in an order that is not fixed, and the resulting differences fall in the last bits of the backward pass, so small differences are expected on different hardware, with different kernel implementations, or under different library versions. Because each configuration is scored at the epoch with the best validation accuracy, such a difference can also change which epoch is selected. Every reported cell is the mean over four task orders.