PILOT: TRANSFERRING LORA ADAPTERS WITH ONE-TIME META-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

LoRA adapters trained on one base model may not retain their task performance when transferred to an updated version. Transferring them requires recovering their task behavior while preserving the target model's instruction-following ability. We propose PILOT, a transfer operator trained once for a source and target model pair and reused across adapters. PILOT first constructs a closed-form approximation of an adapter's effect on the target from unlabeled activations, then applies learned low-rank corrections. Matching intermediate module outputs alone does not ensure correct or complete answers, so the corrections receive feedback through the frozen target model. During supervised meta-training, these corrections are optimized for accurate, complete task responses, with a KL penalty that limits changes to the target model's behavior on general instructions. Within the same task family and matching architectures, the trained operator transfers held-out adapters without ground-truth task labels or gradient computation at transfer time. For different architectures, PILOT-X constructs a transfer basis from source-generated answers using one backward sweep over the calibration data, avoiding iterative fine-tuning for each adapter. Experiments within and across model families show that transferred adapters achieve task accuracy competitive with supervised target adaptation while largely retaining the target model's instruction-following ability.

1 INTRODUCTION

LoRA adapters encode task-specific adaptations as low-rank weight updates learned on a frozen base model (Hu et al., 2022). This parameterization reduces the number of trainable parameters and the training cost of adapting a base model to downstream tasks (Hayou et al., 2024). As base models evolve, adapter transfer offers a way to reuse task-specific capabilities across model versions (Khan et al., 2025). Transfer should preserve these capabilities while retaining the target model's knowledge, consistent with the goal of knowledge-preserving adaptation (Yang et al., 2024). Such reuse can reduce the need for repeated supervised adaptation (Lin et al., 2025) and facilitate the migration of large adapter collections when the original training data are unavailable (Wang et al., 2024).

Existing approaches follow two directions. Data-free approaches transfer low-rank updates by direct reuse or weight-space projection. They rely on architectural compatibility (Khan et al., 2025; Farhadzadeh et al., 2025a), while extensions to heterogeneous models yield gains that vary across model pairs and tasks (Xia et al., 2025). Distillation-based approaches train each target adapter to reproduce the adapted source model's outputs (Wang et al., 2024; Jung & Kim, 2026). In our experiments, direct adapter transfer retains most of the task accuracy gain on two of three model pairs and falls below the unmodified target model on the third. Supervised fine-tuning and direct adapter transfer can degrade the target model's instruction-following ability and cause substantial deviations from its original output distribution on general instructions.

A LoRA update is learned for a particular base model, so changes in base weights and activations can alter its effect after transfer. For models with corresponding modules, we approximate this effect on the target using unlabeled activations, while penalizing changes to module outputs on general instructions. This approximation admits a closed-form solution. This local matching objective does not directly optimize task responses, so we introduce a correction guided by output-level feedback. To avoid optimizing this correction for every new adapter, we learn a shared transfer rule jointly across training adapters and freeze it for reuse within the same model pair and task family.

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.

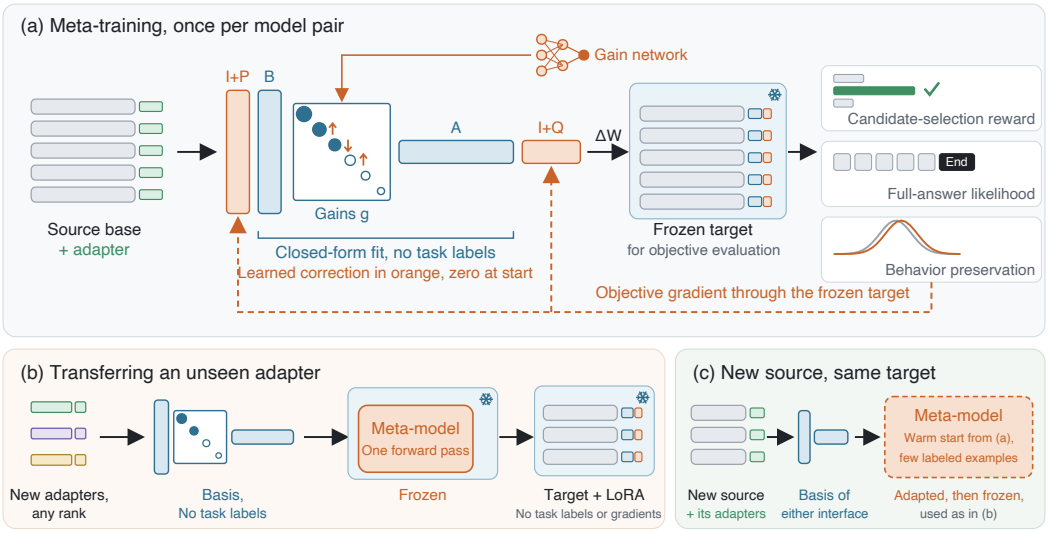

Figure 1: Overview of PILOT. (a) For one model pair, the meta-model θ corrects the closed-form approximation of a training adapter through gradients from the frozen target. (b) An unseen adapter is transferred by one closed-form solution and one forward pass of the frozen meta-model into a standard LoRA. (c) For a new source of the same target, (a) initializes meta-training with few labels.

PILOT, short for Pairwise Induced Low-rank Operator Transfer, implements this approach, as illustrated in Figure 1. Shared low-rank operators adjust the input and output directions of each adapter's transfer basis, producing an update specific to that adapter. The operators are meta-trained once per model pair and task family with an objective that promotes correct, complete task responses and penalizes deviations from the target's behavior on general instructions. Gradients propagate through the frozen target to optimize only the transfer parameters. For a new same-architecture adapter, transfer constructs the basis and applies the frozen operators without ground-truth task labels, gradient computation or parameter optimization. The correction uses the target modules' coordinates and acts on each adapter's transfer basis, allowing the frozen meta-model to transfer adapters with different ranks and training configurations.

Across architectures, source and target modules may not correspond. PILOT-X instead constructs the target-side basis from answers generated by the adapted source on unlabeled prompts. Each transfer requires one backward sweep over the calibration data, without iterative fine-tuning. Both interfaces provide a basis in target-module coordinates, allowing a meta-model trained with one source to initialize meta-training for another, including across architectures, with the target fixed and a small labeled budget. Experiments evaluate held-out adapters from the same task family, including different ranks and training configurations, across three same-architecture pairs and three cross-architecture directions. In the same-architecture comparison, PILOT meta-trained on as many labeled examples as LoRA exceeds supervised LoRA and transfer baselines in mean task accuracy, and transferred adapters largely preserve the target's instruction following and output distribution, whereas supervised target adaptation substantially degrades instruction following. Our contributions are as follows.

- We introduce PILOT, which constructs a preservation-aware closed-form basis and corrects it with low-rank operators meta-trained once per model pair through feedback from the frozen target. New adapters are transferred without ground-truth task labels, gradients or an optimizer, across the evaluated adapter ranks and training configurations.
- PILOT-X extends the framework to heterogeneous architectures through a behavioral interface. It constructs a target-side basis from source-generated answers with one backward sweep, without iterative fine-tuning for each adapter.
- For a fixed target and task family, a meta-model trained with one source initializes meta-training for another, including across architectures. On four targets, this warm start outperforms training from scratch at the same small supervised adaptation budget.
- On three same-architecture pairs and three cross-architecture directions, held-out adapters exceed supervised LoRA in mean accuracy with as many training labels while largely preserving instruction following and output distributions. Ablations and controls assess the correction.

2 RELATED WORK

Transferring adapters across base models. PortLLM (Khan et al., 2025) and Lin et al. (2025) copy task updates to a new model version, which is effective when parameter dimensions match and the base weights are similar. LoRA-X (Farhadzadeh et al., 2025a) and Cross-LoRA (Xia et al., 2025) project updates between weight subspaces without data or gradients. Trans-LoRA (Wang et al., 2024) trains a target adapter on synthetic examples that are labeled by the adapted source and filtered by a learned discriminator, and TiTok (Jung & Kim, 2026) filters such examples by a contrastive log-likelihood score comparing the source model with and without its adapter. Both methods require target-side training for each adapter, as does unfiltered answer distillation. PILOT trains once per model pair and then transfers adapters without an optimizer.

Learned weight generation and reward gradients. Hypernetworks (Ha et al., 2017) generate weights from a conditioning input. PILOT generates no weights from scratch. It rescales and mixes adapter-derived directions, allowing the meta-model to transfer adapters excluded from meta-training. The training signal follows gradient-based reward optimization in diffusion models, which backpropagates through a frozen evaluator to update the generator (Xu et al., 2023; Clark et al., 2024; Prabhudesai et al., 2023). Here the frozen target provides differentiable objective evaluation. The expected candidate reward has a REINFORCE-equivalent gradient (Williams, 1992) computed exactly by summing over candidates, and reference-model KL regularization constrains changes to the target distribution, as in RLHF (Ouyang et al., 2022).

Closed-form editing and preservation. ROME (Meng et al., 2022) and MEMIT (Meng et al., 2023) solve least-squares edits over key-value associations, AlphaEdit (Fang et al., 2025) restricts edits to the null space of activations that should be preserved, RegMean (Jin et al., 2023) merges models by a regression on Gram matrices, and CorDA (Yang et al., 2024) and LoRA-Null (Tang et al., 2026) construct adapters from activation covariances to preserve model behavior. Our initialization belongs to this family. The learned correction of PILOT is trained with answer-level supervision, which activation-level objectives do not use.

3 METHOD

PILOT transfers an adapter in two steps. A closed-form fit on unlabeled activations expresses its effect on the target in a basis of target-module directions (Section 3.2). Low-rank operators, meta-trained once per model pair through the frozen target, then correct this fit for every new adapter (Sections 3.3 and 3.4). A behavioral interface replaces the fit across architectures (Section 3.5).

3.1 PROBLEM SETTING AND REUSABLE ADAPTER TRANSFER

Fix a source model, a target model and a task family. Let $\mathcal{A}_s$ be the LoRA adapters of the source that were trained on tasks of this family, and $\mathcal{A}_t$ the adapters of the target. PILOT learns a map

$$\mathcal{F}_\theta : \mathcal{A}_s \to \mathcal{A}_t, \tag{1}$$

whose parameters θ are shared by all adapters. The learned map defines a reusable transfer rule. The procedure has two stages. In the one-time stage, θ is fitted on a few source adapters with labeled task examples, and gradients pass through the target model. In the transfer stage, θ is frozen and $\mathcal{F}_\theta(a)$ is computed for a new adapter a from unlabeled prompts. Both base models stay frozen throughout.

We first assume that source and target share an architecture, so that corresponding linear modules have weights $W_s, W_t \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, and the source adapter contributes $\Delta W_s = \frac{\alpha_s}{r_s} B_s A_s$ to its module. Section 3.5 removes this assumption. Two sets of unlabeled prompts are available. The adapted source continues each task prompt, and the unmodified target continues each general instruction. The columns of $X_s, X_t \in \mathbb{R}^{d_{\text{in}} \times n}$ hold the inputs of the two modules at the n task positions, and $Z \in \mathbb{R}^{d_{\text{in}} \times m}$ holds the inputs of the target module at the m general positions.

3.2 A PRESERVATION-AWARE TRANSFER BASIS

Copying ΔW_s onto W_t does not necessarily preserve the adapter's effect. The update was trained on source inputs X_s to modify the output of W_s. On the target, it acts on different inputs X_t and base

weights W_t, which can alter its effect. We therefore seek to preserve the adapter's contribution to module outputs. On task inputs, this contribution is $D = \Delta W_s X_s$, the *task effect*.

Matching D alone neglects the adapter's dependence on its source base. The two bases differ on the same prompts by the *base drift* $\Omega = W_s X_s - W_t X_t$. The regression target $Y = D + \beta\Omega$ interpolates between matching the adapter's local effect with $\beta = 0$ and matching the full output of the adapted source module with $\beta = 1$. Full compensation can suppress target-specific behavior at the module level, so we compensate only a fraction of the drift, with $\beta \in \{0.25, 0.5\}$.

We also constrain interference with the target's general behavior. An update fitted to Y may alter outputs on other inputs, so we penalize its effect on the general-instruction inputs Z. For each module, in layer order, we solve

$$\min_{\operatorname{rank}(\Delta W)\le R} \|\Delta W X_t - Y\|_F^2 + \tilde{\mu}\|\Delta W Z\|_F^2 + \gamma\|\Delta W\|_F^2, \qquad \tilde{\mu} = \mu \operatorname{Tr}(X_t X_t^\top)/\operatorname{Tr}(ZZ^\top). \quad (2)$$

Let $M = X_t X_t^\top + \tilde{\mu} ZZ^\top + \gamma I \succ 0$ and $G = Y X_t^\top$. Completing the square turns the objective into $\|\Delta W M^{1/2} - G M^{-1/2}\|_F^2$ plus a constant, so by the Eckart-Young theorem the optimum is the truncated SVD of the whitened matrix. With $GM^{-1/2} = U\Sigma V^\top$ we obtain $\Delta W^\star = U_R \Sigma_R V_R^\top M^{-1/2}$. Later module updates are computed on activations from the partially transferred target, since each layer's update changes subsequent inputs. Computing the solution requires only forward passes. The fit and preservation constraint are local and do not guarantee model-level behavior, motivating the output-level objective in Section 3.4.

The closed form provides both an initialization and a common factorized representation for incoming adapters. We retain additional directions beyond the rank-R optimum. Per module the *transfer basis* consists of $B_f = U\Sigma \in \mathbb{R}^{d_{\text{out}} \times K}$ and $A_f = V^\top M^{-1/2} \in \mathbb{R}^{K \times d_{\text{in}}}$ with $K = R_{\max} + K_b$ directions, namely the $R_{\max}$ leading directions of $GM^{-1/2}$ and the K_b leading directions of the uncompensated drift, $(1-\beta)\Omega X_t^\top M^{-1/2}$. The closed-form approximation is $B_f \operatorname{diag}(g^0) A_f$, where $g^0 \in \{0,1\}^K$ marks the top R directions.

3.3 Reusable low-rank correction operators

The basis depends on the incoming adapter, while PILOT learns correction operators shared across adapters. Write the central update and the two operators as

$$\Delta_g = B_f \operatorname{diag}(g) A_f, \qquad L = P_a P_b, \qquad H = Q_a Q_b, \quad (3)$$

with $P_a \in \mathbb{R}^{d_{\text{out}} \times r}$, $P_b \in \mathbb{R}^{r \times d_{\text{out}}}$, $Q_a \in \mathbb{R}^{d_{\text{in}} \times r}$, $Q_b \in \mathbb{R}^{r \times d_{\text{in}}}$ and a small rank r. The transferred update is

$$\Delta W_\theta = (I + L)\,\Delta_g\,(I + H) = \Delta_g + L\Delta_g + \Delta_g H + L\Delta_g H. \quad (4)$$

The gains g rescale existing basis directions, so Δ_g remains within the row and column spaces of the transfer basis. The output-side operator L can introduce output directions outside the column space of B_f through $L\Delta_g$. The input-side operator H can introduce sensitivity to input directions outside the row space of A_f through $\Delta_g H$. Gains alone cannot express these changes in direction. The ablations and component-wise interventions evaluate their contribution (Tables 11 and 15).

We distinguish adapter-specific bases, module-specific operators and a shared gain network. The basis (B_f, A_f) is recomputed for every incoming adapter. The operators L and H are specific to each target module and shared by all adapters of the model pair and task family. The gain network π_θ is shared by all modules and maps dimensionless statistics ϕ_i of direction i to $g_i = g_i^0 + \tanh \pi_\theta(\phi_i)$ (Appendix B). Because L and H multiply Δ_g, their correction depends on the incoming adapter's basis and vanishes when the central update is zero. The operators thus learn to redirect adapter-specific updates. A zero source adapter need not yield a zero basis, because the regression target also contains the drift term.

The last layer of π_θ, P_a and Q_a are initialized to zero, so ΔW_θ equals the closed-form approximation when meta-training starts. Left and right multiplication do not increase rank, so $\operatorname{rank}(\Delta W_\theta) \le K$. The bound is K rather than R because learned gains may assign nonzero weights to directions excluded by g^0. The factors $(I + L)B_f \operatorname{diag}(g)$ and $A_f(I + H)$ are stored as an ordinary LoRA. Target inference requires neither the meta-model nor the basis.

3.4 META-TRAINING WITH OUTPUT-LEVEL FEEDBACK

The closed form matches module outputs without directly optimizing answer-level behavior. Our meta-training objective addresses three limitations. First, similar intermediate activations need not yield correct candidate selection. Second, selecting the correct candidate at an early token position does not ensure a complete, properly terminated response. Third, constraining module outputs on Z does not ensure that the model's output distribution is preserved on general instructions.

The parameters are fitted over the training adapters of the pair,

$$\theta^\star = \arg\max_\theta \; \mathbb{E}_{a\sim\mathcal{A}_{\text{train}}}\big[J_{\text{dec}}(a) + \eta\, J_{\text{seq}}(a) - \lambda\, J_{\text{pres}}(a)\big], \tag{5}$$

where every term is evaluated on the frozen target with the update $\mathcal{F}_\theta(a)$ attached. Let x be a task prompt whose instruction lists the admissible answers. The candidates share the answer prefix, and z_c is the target logit of candidate c at the position where they first differ. With reference answer $y^\star$,

$$J_{\text{dec}} = \mathbb{E}_x\big[\,\text{softmax}(z/\tau)_{y^\star}\big], \quad J_{\text{seq}} = \mathbb{E}_x\big[\log p_\theta(y^\star_{1:T} \mid x)\big], \quad J_{\text{pres}} = \mathbb{E}_u\big[\text{KL}\big(p_{\text{ref}} \,\|\, p_\theta\big)(u)\big]. \tag{6}$$

J_{dec} is the expected $0/1$ reward of a policy selecting among the listed candidates at the teacher-forced decision position. The expectation over candidates is summed exactly, giving the REINFORCE gradient without sampling noise. This term saturates on confident predictions and emphasizes uncertain candidate predictions. Candidate selection does not measure exact-match accuracy under free generation, so J_{seq} rewards the complete reference answer, including its end-of-turn token. J_{pres} measures KL divergence from the unmodified target model on its own continuations of general instructions u, as in RLHF. All gradients reach θ through the frozen target, which provides differentiable output-level feedback. Data sizes and optimization settings are listed in Appendix B.

3.5 A BEHAVIORAL INTERFACE ACROSS ARCHITECTURES

When source and target differ in tokenizer, width, depth or module types, we do not assume module correspondence and cannot directly construct D or Ω. PILOT-X instead uses a behavioral interface with the same correction operators and meta-training objective.

Both constructions are instances of one local problem. Expanding Equation (2) shows that it is equivalent to

$$\min_{\text{rank}(\Delta)\le R} \; \tfrac{1}{2}\,\text{Tr}\left(\Delta M \Delta^\top\right) - \langle G, \Delta\rangle, \tag{7}$$

in which M measures the update's effect on task and general inputs, and G specifies a direction for changing module weights. In Section 3.2, $G = YX_t^\top$ is constructed from source module outputs. In PILOT-X, the adapted source answers unlabeled task prompts in its own chat template. The answers are retokenized for the target, and one teacher-forced backward sweep over these prompt-answer pairs gives $G = -\partial\mathcal{L}/\partial W_t$ for every module. Source-generated text provides the cross-model interface. With the same M, the whitened SVD of $GM^{-1/2}$ yields the basis (B_f, A_f).

The gradient G does not determine an appropriate transfer step size, and M is a chosen metric rather than a Hessian. Thus, the closed-form optimality of Section 3.2 does not extend to the behavioral objective. The rank-R part of each module is normalized to 1% of $\|W_t\|_F$. A global step size is selected by the likelihood of held-out source answers using forward passes. Basis construction requires no ground-truth task labels or iterative optimization, but it uses one backward sweep over the calibration prompts per adapter. Tables distinguish this regime from same-architecture transfer. The operators of Section 3.3 are meta-trained once per model pair using Section 3.4. This basis captures source behavior through generated answers, and we select between rank-4 and rank-16 configurations by held-out-adapter accuracy on the validation set (Appendix B).

Reuse across sources. The learned parameters are defined in target-module coordinates. The operators L and H act on each module's output and input coordinates, respectively, and the gain network processes each basis direction independently. Their parameter shapes are independent of the source model and the number of basis directions, which varies across interfaces and adapter ranks. The behavioral basis, for example, includes no drift directions. An existing meta-model can therefore initialize meta-training for a new source with the same target when the adapted target modules and operator rank agree,

$$\theta^{(0)}_{s_2\to t} = \theta^\star_{s_1\to t}, \tag{8}$$

after which Equation (5) is optimized on the training adapters of s_2 and the result is frozen and used as in Section 3.3. Within a pair the frozen rule is reused for new adapters, and across sources the rule is reused as an initialization. Whether the initialization is useful is an empirical question, which Section 4.6 answers for four targets.

4 EXPERIMENTS

4.1 SETUP

Task, adapters and metrics. We evaluate seven commonsense tasks from LLM-Adapters (Hu et al., 2023) on Qwen3.5 (Qwen Team, 2026), Llama-3.2 (Grattafiori et al., 2024) and Gemma-4 (Gemma Team, 2026). We extract generated answers, score them by exact match and average the accuracies of the seven tasks, which have 19,149 test items in total. Source LoRAs with r=16 and α=32 cover all linear projections and are trained on 4,096 labeled examples for one epoch with seeds 42, 43 and 44. Each operator is trained on two source adapters and transfers the held-out third. Every transferred update in the comparisons is exported as a rank-16 LoRA, the rank of the source adapters and of the LoRA reference, by a truncated SVD weighted with the target activations on unlabeled task prompts and general instructions (Appendix C). A separate labeled validation set supports monitoring and configuration selection (Appendix B). Target capability preservation is measured by IFEval (Zhou et al., 2023), 0-shot MMLU (Hendrycks et al., 2021), and per-token KL from the unmodified target model on its own continuations of held-out general prompts, which we call gen-KL.

Supervised references and transfer baselines. Supervised references train target adapters on labeled data. They are LoRA (Hu et al., 2022) on 4,096 examples, DoRA (Liu et al., 2024), PiSSA (Meng et al., 2024), LoRA+ (Hayou et al., 2024), NEFTune (Jain et al., 2024) and LoRA-Dash (Si et al., 2025) (Table 6). Baselines without target gradients include direct transfer $W_t + \Delta W_s$, as reused in PortLLM (Khan et al., 2025), LoRA-X-style (Farhadzadeh et al., 2025a) projection onto the target's leading weight subspace, weight interpolation, Cross-LoRA's subspace alignment (Xia et al., 2025), ProLoRA's subspace and null-space projection (Farhadzadeh et al., 2025b), and our adaptation of Command-V's activation converters (Wang et al., 2025) to LoRA weights. Both settings include Trans-LoRA (Wang et al., 2024), TiTok (Jung & Kim, 2026) and unfiltered sequence-level answer distillation (Hinton et al., 2015; Kim & Rush, 2016) from the adapted source. These methods train a LoRA per adapter on source-generated answers, without labels at transfer time.

4.2 TRANSFER QUALITY AND TARGET RETENTION

We evaluate PILOT on held-out adapters (Table 1). PILOT is meta-trained on 4,096 labeled examples, the size of the LoRA training set, and rows marked § use 16,384. The closed-form initialization alone comes within a point of LoRA accuracy on both Qwen targets with little IFEval loss, but falls behind on Llama (Tables 6 and 12). The meta-model improves on it for every pair, most on Llama with 3.7 points, and PILOT exceeds LoRA in mean accuracy on all three targets and on 12 of the 21 combinations of target and task while keeping IFEval close to the unmodified target model. A larger meta-training set adds a little accuracy. Even when source and target are identical and the closed form nearly matches the adapter, the meta-model improves accuracy in every fold (Table 8).

Methods that fit the target to task data trade retention for accuracy. Supervised references, direct transfer on Qwen and weight interpolation lose a large share of the target's IFEval score. The per-adapter distillation methods (unfiltered distillation, TiTok and Trans-LoRA) retain more, but they are less accurate than PILOT and retain less than either of our stages. The remaining baselines without training preserve the target but recover little task accuracy, except ProLoRA, which recovers part of it. Only our two stages combine LoRA-level accuracy with the target's IFEval on all three targets. Appendix F examines target-side training with the same retention objective. On unrelated tasks, both targets preserve baseline math accuracy. Code accuracy is kept on Llama and slightly reduced on Qwen3.5-2B, still above supervised LoRA with its answer prefix removed (Table 7).

Table 1: Same-architecture transfer. 7-task Acc. averages the seven commonsense tasks. IFEval, MMLU and the KL to the unmodified target measure retention. Means over three seeds, 7-task Acc. ± standard deviation, in %. Best in bold, second best underlined, unmodified target excluded.

	Qwen3.5-2B-Base → Instruct				Qwen3.5-4B-Base → Instruct				Llama-3.2-3B-Base → Instruct			
Method	7-task Acc.	IFEval	MMLU	KL	7-task Acc.	IFEval	MMLU	KL	7-task Acc.	IFEval	MMLU	KL
Unmodified target	67.82	65.43	59.86	0	83.17	82.07	72.62	0	44.24	69.50	61.57	0
Supervised adaptation on the target												
LoRA	78.63±0.38	35.80	57.65	0.17	87.66±0.22	30.04	73.37	0.40	76.31±0.17	40.85	60.22	0.32
DoRA	78.68±0.20	36.17	57.74	0.19	87.39±0.22	32.47	73.45	0.37	76.40±0.17	40.60	60.28	0.32
PiSSA	77.06±0.39	19.22	54.99	1.28	86.69±0.26	21.32	69.89	1.36	77.37±0.28	22.24	54.63	0.55
LoRA+	78.60±0.74	27.05	57.13	0.44	87.89±0.34	27.42	73.05	0.73	77.60±0.18	17.74	54.66	1.11
NEFTune	78.77±0.32	32.16	57.29	0.30	87.42±0.09	30.01	73.33	0.43	76.43±0.40	43.38	60.45	0.30
LoRA-Dash	78.64±0.41	35.55	57.47	0.20	87.64±0.13	30.56	73.70	0.40	76.33±0.19	40.17	60.19	0.32
Transfer with per-adapter training on the target												
Unfiltered distillation	78.27±0.32	58.04	59.55	0.03	87.35±0.22	71.23	74.72	0.04	73.86±0.81	65.50	60.47	0.02
TiTok	76.99±0.74	57.49	58.44	0.03	87.11±0.22	72.27	74.65	0.04	73.79±0.75	65.31	60.18	0.03
Trans-LoRA	76.78±0.20	44.73	59.16	0.08	85.48±0.18	58.47	74.03	0.07	71.57±0.12	47.26	59.93	0.17
Transfer without per-adapter training on the target												
Direct transfer	77.80±0.24	31.67	57.87	0.22	80.99±11.46	39.74	74.33	0.19	74.10±0.19	64.94	61.45	0.02
LoRA-X-style projection	71.31±0.11	**67.90**	59.42	0.0015	84.50±0.23	80.65	72.85	0.0013	55.52±2.95	69.44	61.81	0.0009
Cross-LoRA	67.76±0.11	65.74	59.70	**0.0005**	83.17±0.03	**81.58**	72.62	**0.0004**	44.25±0.24	**70.06**	61.58	**0.0003**
ProLoRA	75.77±0.26	61.18	58.72	0.02	86.31±0.37	58.90	73.38	0.01	68.85±1.25	68.27	61.87	0.0036
Command-V, adapted	68.27±0.25	67.04	59.27	0.0063	83.76±0.14	80.84	73.00	0.0045	49.22±6.35	68.02	61.07	0.03
Weight interpolation	79.32±0.15	19.65	59.70	0.36	88.04±0.23	29.02	**74.99**	0.42	73.71±0.90	22.49	58.55	0.44
PILOT	79.02±0.24	65.13	59.63	0.0017	88.49±0.29	80.96	74.07	0.0009	77.95±0.21	68.15	61.54	0.0012
PILOT, 16,384 examples[§]	**79.80**±0.04	66.73	**60.32**	0.0011	**88.65**±0.54	81.45	73.90	0.0007	**78.84**±0.53	68.33	**61.95**	0.0009

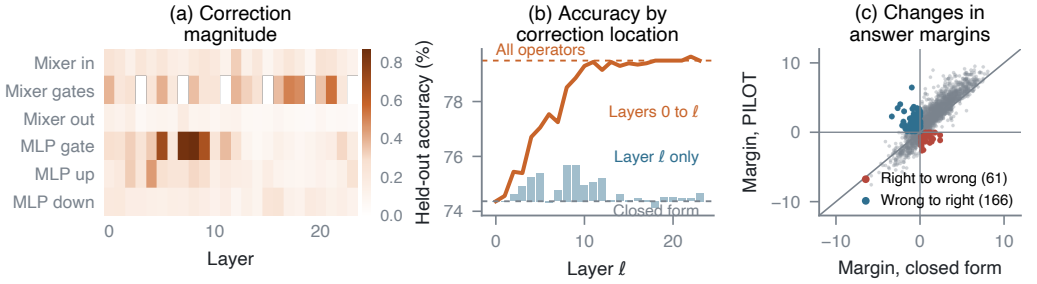

Figure 2: The learned correction on Qwen3.5-2B. (a) Relative size $\|\Delta W_\theta - \Delta W_{\theta_0}\|_F / \|\Delta W_{\theta_0}\|_F$ per layer and module group, averaged over the three folds. White cells indicate module types absent from the corresponding layer. (b) Held-out-adapter validation accuracy with correction applied in layers 0 to ℓ, shown as a line, or only in layer ℓ, shown as bars. (c) Reference-answer margin over the highest-scoring alternative for each validation example, before and after correction.

4.3 GENERALIZATION TO UNSEEN ADAPTERS

All PILOT results use adapters excluded from meta-training. Two further source adapters per pair, trained after the operators were frozen and transferred without retraining or selection, gain 1.7 points over initialization on Qwen3.5-2B and 4.9 to 5.3 on Llama. Adapters trained with other examples, a longer schedule or ranks from 4 to 128 also improve over their closed-form initialization and match or exceed a target-trained LoRA of the same configuration, which loses far more IFEval (Table 9 and Figure 5). The gain shrinks as source rank grows, because the closed form already recovers most accuracy at high rank. At high rank the operator mainly restores retention.

The experiments support reuse within the evaluated task family. We apply an operator to adapters of a disjoint task group in both directions (Table 10). Accuracy on the new group's tasks rises only 0.3 to 0.6 points above the closed-form initialization, so the results do not establish transfer across task groups. Because task content and answer format change together in this split, the experiment cannot separate their effects.

4.4 WHAT THE CORRECTION CHANGES

Operator-induced direction changes account for the accuracy gain. On the held-out seed-44 adapter of Qwen3.5-2B, the gain network alone leaves initialization accuracy unchanged, and the

Table 2: Cross-architecture transfer. 7-task Acc. averages the seven commonsense tasks, and Source LoRA is the incoming adapter on its own base. IFEval, MMLU and the KL to the unmodified target measure retention. Means over three seeds, 7-task Acc. ± standard deviation, in %. Best in bold, second best underlined, unmodified target excluded.

	Llama-3.2-3B → Qwen3.5-4B				Qwen3.5-4B → Llama-3.2-3B				Qwen3.5-4B → Gemma-4-E2B			
Method	7-task Acc.	IFEval	MMLU	KL	7-task Acc.	IFEval	MMLU	KL	7-task Acc.	IFEval	MMLU	KL
Unmodified target	83.17	82.07	72.62	0	44.24	69.50	61.57	0	47.29	77.82	59.44	0
Source LoRA, own base	70.49±0.79				87.89±0.07				87.89±0.07			
Supervised adaptation on the target												
LoRA	87.66±0.22	30.04	73.37	0.40	76.31±0.17	40.85	60.22	0.32	72.20±0.37	68.58	57.19	0.04
Transfer with per-adapter training on the target												
Unfiltered distillation	80.39±1.20	72.83	71.60	0.03	78.70±0.08	65.43	60.05	0.02	**75.55**±0.64	76.71	57.65	0.04
TiTok	80.88±1.04	71.60	72.52	0.03	71.16±1.22	66.61	58.79	0.02	64.06±2.12	74.86	57.19	0.04
Trans-LoRA	81.16±0.35	53.17	71.50	0.09	71.78±0.91	58.78	60.93	0.10	63.53±1.25	74.74	59.23	0.03
Transfer without per-adapter training on the target												
Cross-LoRA	83.17±0.03	81.21	72.61	**0.0004**	44.13±0.05	**69.99**	**61.55**	**0.0003**	47.39±0.16	**77.63**	59.34	**0.0005**
Command-V, adapted	82.62±0.08	**81.27**	71.45	0.02	49.82±1.71	69.69	61.36	0.0027	41.44±0.67	76.59	**59.43**	0.01
PILOT-X initialization	80.54±0.95	70.18	71.69	0.02	66.44±0.05	65.50	60.02	0.03	33.03±0.10	77.57	59.20	0.0034
PILOT-X	87.93±0.40	77.82	73.93	0.0038	77.84±0.43	66.97	60.62	0.0032	72.97±0.60	76.65	59.28	0.0046
PILOT-X, 16,384 examples[§]	**88.70**±0.33	79.30	**74.56**	0.0031	**80.35**±0.35	66.97	61.10	0.0022	74.54±0.60	75.54	59.34	0.0046

rank-4 operators produce the gain (Table 15). Component-wise interventions locate the contribution (Figure 2, Table 11). The output-side operators and the MLP modules each recover nearly all of the validation gain, the input-side operators little, and correcting the first eleven of 24 layers suffices. Llama-3.2-3B shows the same pattern.

Candidate-selection accuracy and end-of-turn probabilities change differently. The correction addresses distinct initialization failures. On Qwen3.5-2B it improves candidate-selection accuracy. On Llama it improves both this accuracy and the end-of-turn token's log-probability. On Qwen3.5-4B candidate-selection accuracy does not improve, but the end-of-turn log-probability rises from -14.7 to near zero and the test accuracy of generated answers still rises by 1.1 points. The source of this last gain remains unresolved. Without separate objective-term ablations, the increase in end-of-turn probability cannot be attributed to the sequence term alone.

4.5 TRANSFER WITHOUT MODULE CORRESPONDENCE

Behavioral initialization closes about two thirds of the Llama accuracy gap between the unmodified target model and supervised LoRA (Table 2), but matching weaker source answers reduces Qwen3.5-4B accuracy, and on Gemma the initialization breaks the answer format, which the correction restores. Correction improves all three targets, and a larger meta-training set, used in the rows marked [§], improves them further. Without an incoming adapter, Qwen accuracy falls and Llama accuracy is near zero, indicating that the incoming source adapter is necessary for these accuracy gains (Table 13).

The PILOT-X operator uses rank 16 and longer training on two targets, selected on validation examples over rank 4 in Section 3.3 (Appendix B, Table 13). It improves accuracy for every fold and direction and exceeds the LoRA reference on all targets. Configuration selection requires labeled validation data, with test items excluded. PILOT-X is more accurate than TiTok and Trans-LoRA on every target. Unfiltered answer distillation is less accurate on Qwen3.5-4B but more accurate on Gemma, and on Llama PILOT-X leads only with the larger meta-training set. PILOT-X loses at most 4.3 IFEval points.

4.6 META-TRAINING BUDGET AND INITIALIZATION ACROSS SOURCES

The main rows of Tables 1 and 2 are meta-trained with 4,096 labeled examples and the rows marked [§] with 16,384, both for 512 to 768 steps. The larger set adds under a point within and up to 2.5 points across architectures. A quarter of the examples with a third to half of the steps costs at most 1.9 points, and 256 examples cost 2.3 to 4.1. At 256 steps, validation accuracy saturates beyond 4,096 examples, and tripling the steps changes test accuracy by at most a point (Appendix H).

Figure 3 evaluates new sources for four fixed targets using behavioral bases. Initial weights come from Table 1 or, for Gemma, Table 2. Meta-training starts from these weights (Equation (8)) or from

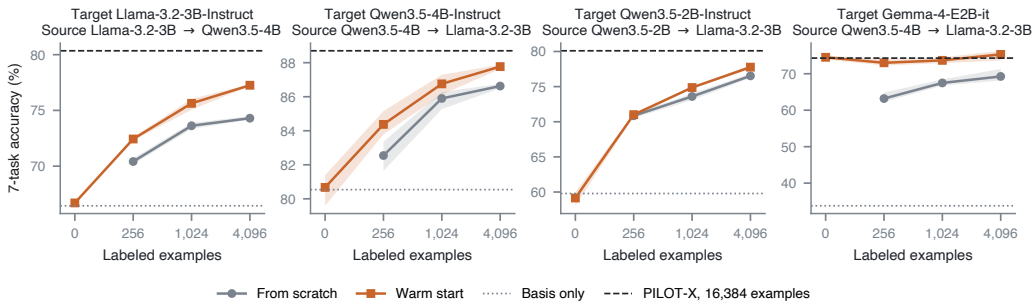

Figure 3: Meta-model transfer to a new source model. Held-out-adapter 7-task accuracy after 256 steps of meta-training from scratch or from the existing meta-model, averaged over three seeds.

scratch for 256 steps on 256, 1,024 or 4,096 labeled examples, matched by fold. Warm starts are more accurate than training from scratch for every target and budget and in 35 of 36 fold pairs.

The initialization benefit depends on prior meta-training. Before adaptation, gains appear only when the basis type matches prior training. On Gemma, whose meta-model was trained on behavioral bases, the loaded meta-model is already near the reference, and adaptation changes it by under two points. On the other three targets, whose meta-models use closed-form bases, the advantage appears only after adaptation (Table 14). Further seeds, 64 examples and the reverse direction keep the warm-start advantage. IFEval does not consistently improve, since warm starts match or exceed training from scratch in 23 of 45 comparisons and are mostly lower on Gemma. The shared target-only LoRA of Appendix F, which uses no incoming adapter, gains as much from warm starts on Qwen3.5-2B and Llama-3.2-3B (Appendix H). This control does not isolate transfer-specific knowledge, because the advantage reflects supervised learning of the target and task family. For a fixed target and task family, a meta-model trained with one source is a useful initialization for another. Example counts cover adaptation of a fully trained meta-model.

4.7 Ablation and parameter sensitivity

Table 15 examines the learned correction, the KL weight λ and the operator rank r on a held-out Qwen3.5-2B adapter. Removing the correction reduces accuracy by 2.3 points, and the gain network alone recovers none of it (Section 4.4). Varying λ from 1 to 20 barely changes accuracy, while a stronger retention penalty halves gen-KL. Operator ranks 1, 4 and 16 all improve on the closed-form initialization, with diminishing returns.

Shared-correction controls in Appendix F show that a target-only LoRA and an additive correction trained under the same conditions attain higher task accuracy, with higher gen-KL, when labeled training data cover the adapters' tasks. In the held-out task-group comparison, the closed-form transfer and both corrections outperform the target-only LoRA.

5 Conclusion and limitations

We introduced PILOT, which transfers LoRA adapters by correcting an adapter-specific basis with reusable low-rank operators. The operators are meta-trained through the frozen target to improve task responses while constraining changes to general behavior. Once trained, the rule transfers same-architecture adapters without ground-truth task labels or gradients. PILOT-X extends the framework across architectures using source-generated answers and one backward sweep per adapter. Across six model pairs, held-out adapters exceed the supervised LoRA reference in mean task accuracy while largely retaining instruction following. Reuse covers the tested adapter ranks and training configurations, and the learned rule provides an initialization for meta-training with new sources at a fixed target and task family. The generality of the learned transfer rule beyond the evaluated settings remains to be established.

AI USE STATEMENT

We used a large language model to assist with writing and editing the text of the paper and with writing and debugging software code. The research ideas, the method and the experimental design are the authors' own. We reviewed all AI-assisted text and code, every number in the tables and figures is produced by scripts from stored evaluation outputs, and we take responsibility for the full content of this work.

ETHICS STATEMENT

This work involves no human subjects, crowdsourcing or personal data. All experiments use publicly released model weights and public benchmarks, namely the LLM-Adapters commonsense tasks, IFEval and MMLU, under their respective licenses. Transfer carries an adapter's behavior to a new base model, including any undesirable behavior the adapter encodes, and it could be applied to adapters trained for harmful purposes. The retention penalty limits changes to the target's behavior on general instructions, but we did not evaluate safety or refusal behavior, so transferred adapters should be checked with the same safety evaluations as the target model before deployment.

REPRODUCIBILITY STATEMENT

Sections 3.2 to 3.4 specify the initialization, the operator and the objective. Appendix B lists data sizes, prompt sources and hyperparameters. Commonsense evaluations extract generated answers and score them by exact match on the full public test sets, IFEval uses the official scorer, and MMLU is 0-shot. All numbers are three-seed means of leave-one-adapter-out transfers unless a table states otherwise. Code, meta-models and transferred adapters will be released upon publication.

REFERENCES

Kevin Clark, Paul Vicol, Kevin Swersky, and David J. Fleet. Directly fine-tuning diffusion models on differentiable rewards. In *International Conference on Learning Representations*, 2024.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. AlphaEdit: Null-space constrained knowledge editing for language models. In *International Conference on Learning Representations*, 2025.

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

Farzad Farhadzadeh, Debasmit Das, Shubhankar Borse, and Fatih Porikli. Zero-shot adaptation of parameter-efficient fine-tuning in diffusion models. In *International Conference on Machine Learning*, 2025b.

Gemma Team. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026. URL `https://arxiv.org/abs/2607.02770`.

Aaron Grattafiori et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

David Ha, Andrew Dai, and Quoc V. Le. HyperNetworks. In *International Conference on Learning Representations*, 2017.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. LoRA+: Efficient low rank adaptation of large models. In *International Conference on Machine Learning*, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

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

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Ka-Wei Lee. LLM-Adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 5254–5276, 2023. URL https://aclanthology.org/2023.emnlp-main.319/.

Neel Jain, Ping-yeh Chiang, Yuxin Wen, John Kirchenbauer, Hong-Min Chu, Gowthami Somepalli, Brian R. Bartoldson, Bhavya Kailkhura, Avi Schwarzschild, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. NEFTune: Noisy embeddings improve instruction finetuning. In *International Conference on Learning Representations*, 2024.

Xisen Jin, Xiang Ren, Daniel Preoţiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. In *International Conference on Learning Representations*, 2023.

Chanjoo Jung and Jaehyung Kim. TiTok: Transfer token-level knowledge via contrastive excess to transplant LoRA. In *International Conference on Learning Representations*, 2026.

Rana Muhammad Shahroz Khan, Pingzhi Li, Sukwon Yun, Zhenyu Wang, Shahriar Nirjon, Chau-Wai Wong, and Tianlong Chen. PortLLM: Personalizing evolving large language models with training-free and portable model patches. In *International Conference on Learning Representations*, 2025.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1317–1327. Association for Computational Linguistics, 2016. doi: 10.18653/v1/D16-1139.

Pin-Jie Lin, Rishab Balasubramanian, Fengyuan Liu, Nikhil Kandpal, and Tu Vu. Efficient model development through fine-tuning transfer. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 2617–2636, 2025. URL https://aclanthology.org/2025.emnlp-main.131/.

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

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

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, 2022.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. In *International Conference on Learning Representations*, 2023.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022.

Mihir Prabhudesai, Anirudh Goyal, Deepak Pathak, and Katerina Fragkiadaki. Aligning text-to-image diffusion models with reward backpropagation. *arXiv preprint arXiv:2310.03739v2*, 2023. Later versions withdrawn by the authors; subsumed by arXiv:2407.08737.

Qwen Team. Qwen3.5: Towards native multimodal agents. Blog post, February 2026. URL https://qwen.ai/blog?id=qwen3.5. Accessed 2026-09-18.

Chongjie Si, Zhiyi Shi, Shifan Zhang, Xiaokang Yang, Hanspeter Pfister, and Wei Shen. Unleashing the power of task-specific directions in parameter efficient fine-tuning. In *International Conference on Learning Representations*, 2025.

Pengwei Tang, Xiaolin Hu, Yong Liu, Lizhong Ding, Dongjie Zhang, Xing Wu, and Debing Zhang. Put the space of LoRA initialization to the extreme to preserve pre-trained knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

Barry Wang, Avi Schwarzschild, Alexander Robey, Ali Payani, Charles Fleming, Mingjie Sun, and Daphne Ippolito. Command-V: Pasting LLM behaviors via activation profiles. *arXiv preprint arXiv:2506.19140*, 2025.

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

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3–4):229–256, 1992. doi: 10.1007/BF00992696.

Feifan Xia, Mingyang Liao, Yuyang Fang, Defang Li, Yantong Xie, Weikang Li, Yang Li, Deguo Xia, and Jizhou Huang. Cross-LoRA: A data-free LoRA transfer framework across heterogeneous LLMs. *arXiv preprint arXiv:2508.05232*, 2025.

Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. ImageReward: Learning and evaluating human preferences for text-to-image generation. In *Advances in Neural Information Processing Systems*, 2023.

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*, 2024.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

A Additional related work

Adapters on a fixed base. LoRA (Hu et al., 2022) learns low-rank updates on a frozen base, and later work changes its parametrization, initialization or optimization (Liu et al., 2024; Meng et al., 2024; Hayou et al., 2024; Si et al., 2025; Jain et al., 2024). These methods adapt a given base with labels and gradients, and we use them as supervised references on the target.

B Implementation details

Model pairs. We transfer adapters from Qwen3.5-2B-Base to Qwen3.5-2B-Instruct, from Qwen3.5-4B-Base to Qwen3.5-4B-Instruct, and from Llama-3.2-3B-Base to Llama-3.2-3B-Instruct. For PILOT-X we transfer the same source adapters across families, namely Llama-3.2-3B to Qwen3.5-4B-Instruct, Qwen3.5-4B to Llama-3.2-3B-Instruct, and Qwen3.5-4B to Gemma-4-E2B-it.

Transfer protocol. The closed-form basis takes X_t from 1,024 unlabeled task prompts and Z from 1,024 held-out databricks-dolly-15k prompts in the chat template of the target. We set μ=10 and R=128, and β=0.25 except for the 4B pair, which uses β=0.5. These values were chosen on a labeled development split of 1,094 items before meta-training. For each pair we meta-train on two of the three source adapters and transfer the third, and we rotate over the three seeds. Each reported adapter is therefore new to the meta-model that transfers it. The operator rank is 4 in the same-architecture experiments. For PILOT-X it is selected by validation accuracy, which is higher at rank 16 in every fold of every direction. Meta-training examples come from the commonsense pool and are disjoint from the training examples of all adapters, from the development split and from the test sets. Within a pair, the held-out adapter differs from the training adapters only by its training seed.

Data. Source adapters use the fixed 4,096-example commonsense training set of the evaluation protocol. Meta-training examples are drawn from the LLM-Adapters commonsense-170k pool after removing every instruction that appears in the source training set or in the development split of 1,094 items. The meta-models of the main rows of Tables 1 and 2 use the first 4,096 of 16,384 labeled meta-training examples, and those marked § use all 16,384. The analyses of Sections 4.3, 4.4, 4.6 and 4.7 and of the appendix use the 16,384-example meta-models unless stated otherwise. A disjoint *validation set* of 4,096 labeled examples is used to track training and select configurations. This validation set is separate from the 1,094-example development split. Meta-training alternates between the two training adapters of a fold. Test sets are the public test splits. No decision was based on them. The operator rank of PILOT-X was selected using validation accuracy. General instructions come from databricks-dolly-15k in the chat template of the target. We use 1,024 of them for the closed-form basis, 512 together with the target's own 64-token continuations for the KL term, and 256 held-out prompts for gen-KL.

Closed-form basis. We set β=0.25 for 2B and Llama and 0.5 for 4B, μ=10 and R=128. We scale the ridge coefficient by the mean diagonal entry of the combined task and preservation Gram matrix, $\gamma=10^{-3}\cdot\operatorname{mean}\operatorname{diag}(X_tX_t^\top+\tilde{\mu}ZZ^\top)$. Further, $R_{\max}$=256 and K_b=64. The adapted source continues each task prompt for 32 tokens. We solve for the transferred updates in layer order, processing attention modules before MLP modules and using activations computed after transferring the preceding modules. The basis of a 2B model takes about 15 minutes on one 80 GB GPU and that of a 4B model about 35 minutes.

Meta-model. The gain network π_θ is an MLP with two hidden layers of width 64, 20 inputs, one output and SiLU activations, and it is shared by all modules. For direction i it reads twenty dimensionless statistics. Twelve describe the direction, namely the relative index, the spectral decay $\log\sigma_i/\sigma_1$, the cumulative energy, the energy $a_iX_tX_t^\top a_i^\top$ on task activations and $\tilde{\mu}\,a_iZZ^\top a_i^\top$ on general activations, where a_i is row i of A_f, both energies on a log scale and their log ratio, its shares of the effect and drift terms, whether it is a drift direction, and its initial gain. Eight describe the module, namely depth, the drift ratio, the fraction of kept directions, the log aspect ratio and a one-hot module type. Its last layer is initialized to zero and the gain is $g=g^0+\tanh(\cdot)$. For the operators, P_a and Q_a are initialized to zero, $P_b,Q_b\sim\mathcal{N}(0,1/d)$ and r=4. The objective uses τ=0.25, η=0.1 and

λ=5, and the KL term is computed on 8 general sequences per step. We optimize with AdamW on 32 task examples per step in micro-batches of 8, learning rates 10^{-3} for the gain network and 3×10^{-4} for the operators, a cosine schedule with 8 warm-up steps and gradient clipping at 1. Training runs for 768 steps on 2B and 512 on 4B and Llama. The two 4B meta-models for seeds 42 and 43 were resumed from their step-256 checkpoint with a restarted schedule at learning rate 5×10^{-4} after an out-of-memory failure. The rank-16 configuration of PILOT-X trains for 768 steps on the Llama and 4B targets and for 512 steps on Gemma, so the selection by validation accuracy is between two configurations, rank 4 with 512 steps and rank 16 with its step count, and not between ranks alone. An independent repetition of the Qwen3.5-4B to Llama direction on bases rebuilt on another machine, with 512 steps for both ranks, gives 78.25 at rank 4 and 79.49 at rank 16. All computation runs in bfloat16 autocast on frozen fp32 target weights, and the target base weights remain frozen.

PILOT-X basis. The adapted source answers 1,024 unlabeled task prompts, and gradients are accumulated over all of them in one sweep. The step size is searched in $\{0, 0.25, 0.5, 1, 2, 4, 8\}$ on 128 held-out source answers and scales the initial gains, $g^0 = \alpha \cdot \mathbf{1}[i \leq R]$. For the two directions between Llama and Qwen the step size was absorbed into B_f, so that g^0 remains an indicator vector. The main directions and the drift directions of a basis are not orthogonal to each other.

PILOT-X operator. We compared two configurations by validation accuracy. The first is the configuration of the same-architecture experiments, operator rank 4 and 512 steps. The second has operator rank 16 and trains for 768 steps on the Qwen3.5-4B and Llama targets and for 512 steps on Gemma. The second achieves higher validation accuracy in every fold and direction and is the PILOT-X of Table 2, and the first is the row r=4 of Table 13. For the same-architecture pairs we keep rank 4 throughout, because we did not run this selection there.

Cost. The one-time stage needs the bases of the training adapters and about half a GPU hour of meta-training for the 2B pair. For each new adapter in the same-architecture setting, constructing the basis takes about 15 minutes of forward passes for a 2B model and 35 minutes for a 4B model on one 80 GB GPU, and applying the frozen meta-model and merging the factors takes seconds. On these model sizes this is not less GPU time than training a LoRA on the target, which takes 5 to 8 minutes. For same-architecture transfer, reuse requires no ground-truth task labels, gradients or optimizer at transfer time for each new adapter. In our runs, it also preserves instruction following better than direct target-side training. An existing meta-model can be adapted to a new source for the same target with 1,024 labeled examples and 256 steps (Appendix H). For adapters whose tasks are covered by the labels of the one-time stage, a shared target LoRA trained with the same objective is the simpler choice (Appendix F).

Supervised references. Unless noted, all target-side references use the source training configuration, namely rank 16, α=32, all linear projections, one epoch at learning rate 10^{-4} and batch 32, on the 4,096-example training set. LoRA+ uses its recommended learning-rate ratio of 16 between the B and A factors, with a base learning rate of 2×10^{-5} for one epoch on 2B and 5×10^{-5} for two epochs on 4B. NEFTune uses noise scale 5. LoRA-Dash follows Si et al. (2025) except for the length of the warm-up phase, which we shorten from their 100 steps to 32 because a full run has 128 steps. After the warm-up, the eight singular directions (u_i, v_i) of each base weight with the largest relative change $|u_i^\top \Delta W v_i|/\sigma_i$ are fixed as task-specific directions. Each receives one learnable scale that starts at zero and is trained jointly with the LoRA factors for the remaining 96 steps, and the result is merged into the saved adapter of rank $16 + 8$.

Training-free transfer baselines. Cross-LoRA (Xia et al., 2025) takes rank-320 truncated SVDs of the source and target base weights, fits the Frobenius-optimal linear maps between the two left bases and the two right bases, and applies these maps to the LoRA factors. Its projection formula is written for equal shapes, so we use the form that is also defined for different shapes, $\Delta W_t = U_t U_s^\top \Delta W_s V_s V_t^\top$. Modules are paired by parameter name as in its algorithm, and a source module without a target module of that name is skipped. ProLoRA (Farhadzadeh et al., 2025b) splits ΔW_s into the part inside the row and column subspaces of W_s and the part inside their complements and projects each part onto the corresponding spaces of W_t, for module pairs whose subspace similarity is at least 0.8. The paper does not state the subspace rank for full-rank weights, and we keep the singular directions that capture 90% of $\|W\|_F^2$, the rule of our LoRA-X-style baseline. Its

projections are written for equal shapes, and for different shapes it first finds a common subspace as in LoRA-X, so we run it on the same-architecture pairs only. Command-V (Wang et al., 2025) transfers residual interventions and converts activations at inference time, so it does not apply to LoRA weights directly. Our adaptation keeps its three components, namely last-token activation profiles on 1,030 general prompts, the layer pairing $l_R = \lfloor l_D |L_R|/|L_D| \rfloor$ and pseudo-inverse converters. It fits one converter on the inputs and one on the outputs of every linear layer adapted with LoRA and folds them into a static update $\Delta W_t = C_{\text{out}}^{\top} \Delta W_s C_{\text{in}}^{\top}$. With fewer prompts than hidden dimensions the untruncated pseudo-inverse is ill-conditioned, and on Llama targets applying the transferred update reduces accuracy to near zero. We therefore truncate singular values below a relative cut-off, which we choose on the development split over thresholds from 10^{-1} to 10^{-5}, also testing the untruncated pseudo-inverse. The value 10^{-3} is best on average and is used for every pair.

Trans-LoRA. We follow Wang et al. (2024). A discriminator, the source base with a LoRA, is trained once per source model to separate the real task inputs from inputs that the source model synthesizes from five seed examples per task. At transfer time the target model synthesizes inputs from the same seeds, the discriminator keeps those it accepts as real until their number equals the size of the training set, the adapted source labels them, and a target LoRA is trained on the result with the distillation settings of our unfiltered answer-distillation baseline. Our adapter covers a mixture of tasks, so seeds and synthesis follow the task proportions of the training set. The discriminators reach 94 to 95% held-out accuracy and accept 6 to 13% of the synthesized inputs, so collecting 4,096 accepted inputs for one pool requires between 36,864 and 88,064 samples from the target model. All six pools reach the full size.

TiTok. We follow Jung & Kim (2026). The excess score of an answer token is $\log p_{\text{expert}}(y_i) - \log p_{\text{amateur}}(y_i)$ under the source model with and without its adapter. We keep the M=4,096 samples with the highest mean excess out of $2M$ and the top 70% of tokens per sample, align the scores to the target tokenizer by character spans, and train a new LoRA with r=8, α=8 and dropout 0.05 for two epochs at 5×10^{-5} with batch 4 and weight decay 10^{-2}. Our implementation differs in two respects. The queries are our unlabeled task prompts and not prompts synthesized by the source expert, and the end-of-turn token is always supervised. Unfiltered answer distillation is the same pipeline without the selection of samples and tokens. The adapted source answers the first M unlabeled prompts and the target LoRA is trained on the full answers with cross-entropy, which is sequence-level knowledge distillation (Kim & Rush, 2016). Gemma-4 shares key-value states across layers through the cache. Every forward pass on it, in all methods, therefore runs with the cache enabled and without activation checkpointing, and the supervised Gemma references were trained in the same way.

Evaluation. The updates produced by PILOT and PILOT-X are reduced to rank 16 as described in Appendix C, exported as LoRA factors and loaded with PEFT. For each benchmark, all methods use the same prompts, generation settings, answer extraction and decoding.

C TRANSFER QUALITY AND TARGET RETENTION

Rows of Table 1. Table 6 lists the seven task accuracies behind every entry of the accuracy columns. The supervised target-side references use labels and gradients on the 4,096-example training set. LoRA+ follows its recommended training configuration. Unfiltered answer distillation and TiTok train a new LoRA for every adapter on answers generated by the adapted source model, with target gradients but without ground-truth task labels. Weight interpolation moves the target weights toward the source base. Our results evaluate adapters excluded from the operator's meta-training. IFEval is strict prompt-level accuracy and MMLU is 0-shot accuracy on 14,042 questions. KL is the mean per-token $\mathrm{KL}(p_{\text{target}} \,\|\, p_{\text{transferred}})$ on 256 held-out general instructions continued by the unmodified target model.

Rank of the transferred adapters. The stored update ΔW_θ of Equation (4) has rank at most K and about 128 nonzero directions. For every comparison we replace it, module by module, with the rank-16 update closest in the metric M of Equation (2), the truncated SVD of $\Delta W_\theta M^{1/2}$ multiplied back by $M^{-1/2}$, where M is collected on the target with the transferred adapter attached. This needs forward passes only. Table 3 reports every tested rank and plain truncated SVD. Rank 128 matches

the stored adapter within 0.1 points, and activation-aware reduction to rank 16 changes accuracy by at most 0.6 points, slightly less than plain SVD.

Table 3: Accuracy of our transferred adapters at every tested rank. Each adapter is reduced to rank k by activation-aware truncation, the default of all other tables, or by plain truncated SVD. Stored is the adapter as exported, of rank up to 320 with about 128 nonzero directions. 7-task accuracy in %, means of three folds.

		Qwen3.5-2B-Instruct			Qwen3.5-4B-Instruct			Llama-3.2-3B-Instruct		
Rank	Reduction	PILOT	PILOT 16,384	Closed form	PILOT	PILOT 16,384	Closed form	PILOT	PILOT 16,384	Closed form
16	activation-aware	79.02	79.80	77.81	88.49	88.65	87.50	77.95	78.84	74.27
16	plain SVD	78.82	79.64	77.55	88.50	88.49	87.42	77.78	78.75	73.67
32	activation-aware	79.11	79.92	78.02	88.48	88.66	87.60	77.97	78.89	74.33
32	plain SVD	79.04	79.89	77.98	88.45	88.65	87.48	77.91	78.80	74.06
64	activation-aware	79.30	79.98	78.19	88.52	88.72	87.63	77.98	78.85	74.39
64	plain SVD	79.21	79.93	78.17	88.56	88.79	87.47	78.01	78.90	74.45
128	plain SVD	79.39	80.14	78.36	88.54	88.80	87.59	78.12	78.92	74.47
Stored		79.35	80.08	78.38	88.57	88.74	87.61	78.08	78.88	74.48
LoRA	rank 16		78.63			87.66			76.31	

Results at the stored rank. Table 4 repeats our rows of Tables 1 and 2 with the stored update in place of its rank-16 copy, and Table 5 does the same for Figure 3. The rows of Table 2 for Qwen3.5-4B and Llama-3.2-3B in both directions come from meta-models retrained on our cluster, the same models whose rank-16 copies appear in the main text.

Table 4: Our rows of Tables 1 and 2 at the stored rank of the exported update, up to 320 with about 128 nonzero directions, instead of rank 16. Columns and notation as in Table 1. Means of three seeds.

	Qwen3.5-2B-Base → Instruct				Qwen3.5-4B-Base → Instruct				Llama-3.2-3B-Base → Instruct			
Method	7-task Acc.	IFEval	MMLU	KL	7-task Acc.	IFEval	MMLU	KL	7-task Acc.	IFEval	MMLU	KL
PILOT	79.35	65.37	59.63	0.0021	88.57	80.71	74.34	0.0011	78.08	67.34	61.40	0.0015
PILOT, 16,384 examples	80.08	65.56	60.23	0.0014	88.74	80.90	74.15	0.0010	78.88	67.78	61.84	0.0012
Closed-form initialization	78.38	66.97	59.05	0.0016	87.61	81.33	73.77	0.0012	74.48	65.87	61.25	0.0022
	Llama-3.2-3B → Qwen3.5-4B				Qwen3.5-4B → Llama-3.2-3B				Qwen3.5-4B → Gemma-4-E2B			
PILOT-X initialization	80.27	71.66	71.71	0.02	66.61	65.56	59.96	0.03	32.67	77.51	59.22	0.0035
PILOT-X	87.93	78.07	73.95	0.0038	77.86	67.47	60.49	0.0032	72.95	76.89	59.25	0.0046
PILOT-X, 16,384 examples	88.70	79.36	74.53	0.0031	80.35	67.47	61.13	0.0022	74.61	75.42	59.37	0.0047

Table 5: Figure 3 at the stored rank. 7-task accuracy of the held-out adapter of the new source after 256 steps of meta-training from scratch and from the existing meta-model, means over the folds where both exist.

Target	256 examples		1,024 examples		4,096 examples	
	Scratch	Warm	Scratch	Warm	Scratch	Warm
Llama-3.2-3B-Instruct	70.38	72.40	73.58	75.57	74.41	77.20
Qwen3.5-4B-Instruct	82.54	84.35	85.92	86.72	86.60	87.73
Qwen3.5-2B-Instruct	70.97	71.07	73.63	74.76	76.56	77.73
Gemma-4-E2B-it	63.13	72.87	67.42	73.64	69.28	75.33

Meta-training dynamics. Figure 4 follows the three terms of the objective on the validation set during meta-training of the 2B and Llama pairs.

Off-task behavior. Table 7 evaluates commonsense adapters on math and code problems that they were not trained for, on Qwen3.5-2B and Llama-3.2-3B. Several adapters trained on or transferred to the target answer GSM8K problems without generating reasoning steps and lose almost all of their accuracy on this benchmark, and they prepend the commonsense answer phrase to generated code, which fails the strict MBPP protocol. The closed-form approximation and PILOT leave math accuracy at the level of the unmodified target model on both targets. On code, PILOT does not prepend the answer phrase. On Llama it keeps the unmodified target model's pass rate, 42.86 against 42.54, while the supervised LoRA keeps 20.85 even with the answer phrase removed. On Qwen3.5-2B

Table 6: Per-task results of same-architecture transfer from Base to Instruct models with all supervised references, accuracy in %. *Supervised on the target with 4,096 examples. †Held-out adapters. §Meta-trained with 16,384 labeled examples. Every other PILOT row is meta-trained with 4,096, as many as the LoRA reference. ♯Trains a new LoRA per adapter on answers generated by the adapted source, with target gradients but no ground-truth task labels. ‡Moves the target weights toward the source base. ♭Recommended LoRA+ configuration. ΔLoRA and ΔTarget compare the task average with the LoRA reference and the unmodified target model. Means over three source seeds. PILOT rows are shaded. °Our adaptation of Command-V to weight-space LoRA (Appendix B). Best in bold and second best underlined, without the unmodified target model. Our rows are the transferred adapters reduced to rank 16 (Appendix C).

	Method	PIQA	SIQA	HellaS.	WinoG.	ARC-e	ARC-c	OBQA	Avg.	ΔLoRA	ΔTarget
Qwen3.5-2B-Inst (target)	Unmodified target	71.71	63.10	54.42	54.78	89.44	75.51	65.80	67.82	−10.81	0.00
	LoRA*	80.29	71.48	83.08	65.64	92.92	81.54	75.47	78.63	0.00	+10.81
	DoRA*	80.23	71.36	82.96	65.83	92.95	81.66	75.80	78.68	+0.05	+10.86
	PiSSA*	77.80	69.87	80.82	65.80	91.41	78.61	75.13	77.06	−1.57	+9.24
	LoRA+*♭	80.30	70.64	83.46	66.56	92.59	80.66	76.00	78.60	−0.03	+10.78
	NEFTune*	81.25	71.60	82.90	65.46	92.93	81.26	76.00	78.77	+0.14	+10.95
	LoRA-Dash*	80.31	71.34	83.11	65.77	92.95	81.43	75.60	78.64	+0.01	+10.82
	Transferred from Qwen3.5-2B-Base without ground-truth task labels at transfer time										
	Direct transfer	80.31	70.83	80.88	65.38	92.45	80.60	74.14	77.80	−0.83	+9.98
	LoRA-X-style projection	74.56	66.02	63.94	55.67	91.29	77.36	70.33	71.31	−7.32	+3.49
	Cross-LoRA	71.35	63.12	54.74	54.59	89.52	75.17	65.87	67.76	−10.87	−0.06
	ProLoRA	78.35	69.11	74.73	60.59	92.52	80.29	74.80	75.77	−2.86	+7.95
	Weight interpolation‡	**81.29**	71.82	83.61	65.54	93.28	82.54	77.20	79.32	+0.69	+11.50
	Command-V, adapted°	72.62	64.09	56.73	54.33	89.67	75.45	65.00	68.27	−10.36	+0.45
	Unfiltered distillation♯	80.58	70.66	82.29	65.27	92.72	80.69	75.67	78.27	−0.36	+10.45
	TiTok♯ (Jung & Kim, 2026)	78.29	68.94	83.23	63.22	91.86	80.43	72.93	76.99	−1.64	+9.17
	Trans-LoRA♯	80.14	70.25	79.28	60.35	92.55	80.92	74.00	76.78	−1.85	+8.96
	Closed-form initialization	80.54	70.78	81.22	62.46	92.90	81.65	75.13	77.81	−0.82	+9.99
	PILOT†	79.38	70.73	82.61	**67.85**	92.80	**82.71**	77.07	79.02	+0.39	+11.20
	PILOT, 16,384 examples†§	81.16	**72.07**	**84.53**	67.25	**93.52**	82.68	**77.40**	**79.80**	+1.17	+11.98
Qwen3.5-4B-Inst (target)	Unmodified target	85.26	74.67	84.89	66.30	96.72	90.78	83.60	83.17	−4.49	0.00
	LoRA*	88.61	78.45	92.94	75.01	97.53	91.95	89.13	87.66	0.00	+4.49
	DoRA*	88.36	78.08	92.82	74.69	97.66	91.70	88.40	87.39	−0.27	+4.22
	PiSSA*	86.31	76.73	91.46	77.19	96.56	90.70	87.87	86.69	−0.97	+3.52
	LoRA+*♭	87.36	77.53	92.56	78.79	97.50	91.98	89.53	87.89	+0.23	+4.72
	NEFTune*	88.14	78.34	92.83	75.19	97.54	92.24	87.67	87.42	−0.24	+4.25
	LoRA-Dash*	88.34	77.91	92.69	76.03	97.60	91.72	89.20	87.64	−0.02	+4.47
	Transferred from Qwen3.5-4B-Base without ground-truth task labels at transfer time										
	Direct transfer	85.76	67.72	91.77	70.40	87.33	82.31	81.60	80.99	−6.67	−2.18
	LoRA-X-style projection	86.52	76.19	86.89	68.85	96.92	91.24	84.87	84.50	−3.16	+1.33
	Cross-LoRA	85.36	74.58	84.77	66.38	96.68	90.81	83.60	83.17	−4.49	0.00
	ProLoRA	87.14	77.87	90.36	72.22	97.39	91.86	87.33	86.31	−1.35	+3.14
	Weight interpolation‡	**89.34**	78.15	93.21	76.32	97.60	92.24	89.40	88.04	+0.38	+4.87
	Command-V, adapted°	85.66	75.27	85.87	67.51	96.78	90.99	84.27	83.76	−3.90	+0.59
	Unfiltered distillation♯	88.74	77.97	92.36	74.56	97.38	92.27	88.20	87.35	−0.31	+4.18
	TiTok♯ (Jung & Kim, 2026)	88.03	76.24	92.82	75.01	97.28	92.21	88.20	87.11	−0.55	+3.94
	Trans-LoRA♯	86.78	76.46	90.15	69.46	97.25	92.01	86.27	85.48	−2.18	+2.31
	Closed-form initialization	88.59	77.62	92.81	74.69	97.52	92.18	89.07	87.50	−0.16	+4.33
	PILOT†	87.99	**78.63**	92.97	**79.77**	97.63	**92.95**	89.47	88.49	+0.83	+5.32
	PILOT, 16,384 examples†§	88.56	78.11	**93.50**	79.74	**97.72**	92.43	**90.47**	**88.65**	+0.99	+5.48
Llama-3.2-3B-Inst (target)	Unmodified target	64.47	33.32	29.02	47.12	52.69	36.86	46.20	44.24	−32.07	0.00
	LoRA*	81.83	71.82	84.09	62.17	87.73	75.45	71.07	76.31	0.00	+32.07
	DoRA*	81.92	72.18	84.24	62.53	87.53	75.26	71.13	76.40	+0.09	+32.16
	PiSSA*	80.52	**73.90**	85.03	69.90	87.07	71.22	73.93	77.37	+1.06	+33.13
	LoRA+*♭	80.90	73.25	**85.30**	70.22	87.00	72.72	73.80	77.60	+1.29	+33.36
	NEFTune*	81.97	71.67	83.67	63.01	87.44	74.94	72.33	76.43	+0.12	+32.19
	LoRA-Dash*	81.79	71.97	84.10	62.51	87.73	75.51	70.73	76.33	+0.02	+32.09
	Transferred from Llama-3.2-3B-Base without ground-truth task labels at transfer time										
	Direct transfer	80.58	70.49	75.67	57.83	86.95	74.38	72.80	74.10	−2.21	+29.86
	LoRA-X-style projection	72.00	47.39	38.40	52.64	64.25	53.47	60.47	55.52	−20.79	+11.28
	Cross-LoRA	64.45	33.50	29.13	47.09	52.64	36.63	46.33	44.25	−32.06	+0.01
	ProLoRA	77.46	62.64	61.83	55.38	83.92	71.27	69.47	68.85	−7.46	+24.61
	Weight interpolation‡	80.65	68.85	80.98	57.93	85.79	71.36	70.40	73.71	−2.60	+29.47
	Command-V, adapted°	53.81	41.47	44.81	47.25	61.63	48.49	47.07	49.22	−27.09	+4.98
	Unfiltered distillation♯	80.69	67.14	81.13	59.88	86.87	71.53	69.80	73.86	−2.45	+29.62
	TiTok♯ (Jung & Kim, 2026)	80.40	67.20	82.29	58.99	86.50	72.12	69.00	73.79	−2.52	+29.55
	Trans-LoRA♯	80.56	68.13	70.93	54.59	86.50	71.19	69.07	71.57	−4.74	+27.33
	Closed-form initialization	80.97	69.87	77.19	58.27	86.91	73.18	73.47	74.27	−2.04	+30.03
	PILOT†	81.50	72.41	83.29	70.14	86.99	74.86	**76.47**	77.95	+1.64	+33.71
	PILOT, 16,384 examples†§	**82.23**	73.17	84.90	**71.72**	**88.19**	**75.65**	76.00	**78.84**	+2.53	+34.60

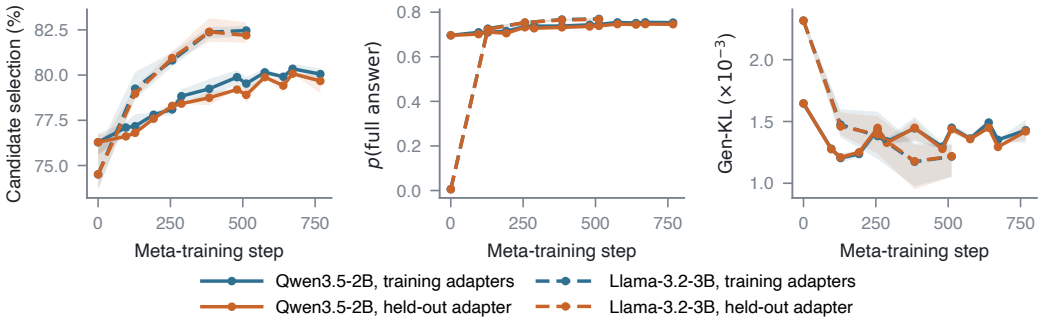

Figure 4: Meta-training on Qwen3.5-2B, shown as solid lines, and Llama-3.2-3B, shown as dashed lines, measured on 2,048 validation examples. Lines are means over the three folds and bands show the range. The left panel shows candidate-selection accuracy, the middle panel the probability of the complete reference answer including the end-of-turn token, and the right panel the KL to the unmodified target model on held-out general instructions.

it scores 38.84, from 35.2 to 41.3 across seeds, below the 42.2 of the unmodified target model and above the 34 to 36 that the supervised adapters reach with the answer phrase removed. The closed-form initialization alone varies strongly across seeds on this target, from 4.8 to 35.6. At the stored rank, PILOT scored 28.4 on Qwen3.5-2B, from 21.9 to 32.4, so reduction to rank 16 improves code retention on this target.

Table 7: Off-task behavior of commonsense adapters on two targets. The prompts contain only the problem and the answer format, without the instruction wrapper of the commonsense training examples. GSM8K and AQuA report answer accuracy and MBPP reports the pass rate on three assertions. The final column removes a leading "the correct answer is" from generated code before execution. The number of source seeds behind each mean is given in parentheses. Notation as in Table 1.

Method	GSM8K	AQuA	MBPP strict	MBPP, answer prefix removed
Qwen3.5-2B-Instruct				
Unmodified target	57.71 (1)	66.54 (1)	42.22 (1)	42.22 (1)
LoRA*	0.00 (3)	32.41 (3)	0.00 (3)	35.87 (3)
DoRA*	0.07 (1)	33.07 (1)	0.00 (1)	35.24 (1)
Direct transfer	0.00 (1)	31.10 (1)	0.00 (1)	30.79 (1)
Weight interpolation‡	0.00 (1)	31.10 (1)	0.00 (1)	32.06 (1)
Closed-form init. only	58.21 (3)	68.50 (3)	22.86 (3)	23.07 (3)
PILOT†	57.81 (3)	69.03 (3)	38.84 (3)	38.84 (3)
Llama-3.2-3B-Instruct				
Unmodified target	75.97 (1)	56.69 (1)	42.54 (1)	42.54 (1)
LoRA*	0.37 (3)	21.92 (3)	0.00 (3)	20.85 (3)
Closed-form init. only	75.15 (3)	56.43 (3)	40.00 (3)	40.00 (3)
PILOT†	75.92 (3)	55.91 (3)	42.86 (3)	42.86 (3)

Source and target identical. As a check of what the meta-model adds when there is nothing to transfer, we run the same-architecture protocol with each base model as both source and target, so that $X_s = X_t$ and the drift term vanishes. The source adapters, folds, objective and step counts are those of Table 1, and a separate meta-model is trained for each base on the 16,384 labeled examples of the rows marked § (Table 8). For seed 44 the closed form, reduced to rank 16, is 0.3, 0.1 and 0.8 points below the source adapter on the three bases, and within 0.2 points at the stored rank. PILOT is above the source LoRA of the same seed in all nine folds, by 0.6 to 1.1 points on the two Qwen bases and by 2.1 to 3.1 points on Llama-3.2-3B, which is about three quarters of what the correction adds on the instruction-tuned target for Qwen3.5-4B, about half for Llama-3.2-3B and about two fifths for Qwen3.5-2B. IFEval is 0.4 to 4.3 points above the closed form, MMLU is at most 1.6 points above it, and the generation KL stays below 0.002. Part of what the meta-model adds is therefore a

correction of the adapter for the task family that does not depend on a change of base, in line with the target-only control of Appendix H, and the remainder compensates the change of base.

Table 8: Source and target identical. Adapters trained on a base model are transferred to that same base model. 7-task accuracy of the source LoRA on its own base, of the closed form, and of PILOT with a meta-model trained on this pair, over three folds with held-out adapters of seeds 42 to 44, the gain of PILOT over the source LoRA of the same seed, and the retention metrics of PILOT, with the closed form in brackets. The closed form is the seed-44 fold. The last column repeats the gain of PILOT over the closed form when the same source adapters are transferred to the instruction-tuned model with the 16,384-example meta-model, the rows marked § in Table 1.

Base model	Source LoRA	Closed form	PILOT	Gain over source LoRA	IFEval	MMLU	KL	Gain on the Instruct target
Qwen3.5-2B-Base	79.29	78.84	80.04	+0.75	47.5 (44.7)	61.3	0.0008	+1.99
Qwen3.5-4B-Base	87.89	87.81	88.77	+0.88	65.6 (64.5)	73.9	0.0005	+1.15
Llama-3.2-3B-Base	70.49	70.64	73.01	+2.51	19.6 (17.0)	48.9	0.0015	+4.57

D REUSE AFTER FREEZING

Adapters trained after the meta-models were frozen. We trained two more source adapters per pair with seeds 45 and 46 after the meta-models had been frozen and transferred them with the meta-model of the seed-44 fold. They score 79.18 and 79.67 on 2B, where the closed-form initialization alone gives 77.52 and 78.00, and 78.72 and 78.79 on Llama, where it gives 73.80 and 73.48. These values are close to the three-fold means of 79.80 on 2B and 78.84 on Llama. The source adapters of this check were trained again with the same configuration for the rank-16 evaluation.

Adapters with other ranks and training configurations. The operators are trained on adapters produced with one training configuration, rank 16 and one epoch at 10^{-4} on the standard training set. To test whether they carry over to adapters of other sizes and training configurations, we trained additional source adapters on Qwen3.5-2B-Base after the operators had been frozen. One uses a different draw of 4,096 labeled examples from the same pool, of which 187 have instructions that also appear in the meta-training set. Five configurations use rank 4, 8, 32, 64 or 128 in place of 16, each with three seeds. One trains for three epochs at 2×10^{-4}. Each was transferred with the operator of the seed-44 fold and by the closed form alone, without any selection, and Table 9 compares both with a LoRA trained on the target with the same training configuration. The closed-form construction represents each adapter as a basis of K directions per module, so the operators do not take the source adapter's rank as an explicit input, and Figure 5 follows the rank of the source adapter from 4 to 128.

Table 9: Source adapters trained with other training configurations on Qwen3.5-2B-Base, transferred to Qwen3.5-2B-Instruct with the operator of one fold, frozen before these adapters were trained, and by the closed form alone, against a LoRA trained on the target with the same training configuration. Rank rows are means of three seeds, the other rows one seed, accuracy in %.

	7-task Acc.			IFEval		
Source configuration	Target LoRA	Closed form	PILOT	Target LoRA	Closed form	PILOT
Different 4,096 examples	78.69	76.64	78.72	38.45	67.84	68.21
Rank 4	77.01	76.24	79.04	44.98	68.64	67.53
Rank 8	78.17	77.05	79.51	39.00	68.76	66.73
Rank 32	79.54	78.15	79.95	29.39	64.63	65.06
Rank 64	79.60	78.38	79.55	24.09	62.54	64.82
Rank 128	78.43	78.78	79.87	20.46	55.64	61.06
3 epochs, lr 2×10^{-4}	79.49	79.09	80.48	29.39	59.33	62.29

Composition of the task groups. Group A of Table 10 contains BoolQ, PIQA, SIQA and HellaSwag, and group B contains WinoGrande, ARC and OBQA. WinoGrande makes up 89% of the examples in group B, which limits the cross-group test.

Transfer of an unrelated adapter. A math adapter transferred by the closed form scores 59.62 on the commonsense suite. Applying the commonsense-trained operators to it gives 62.05. Both values are 6 to 8 points below the unmodified target model's score of 67.82, indicating that the high accuracies in Table 1 depend on the task information supplied by the source adapter (Table 15).

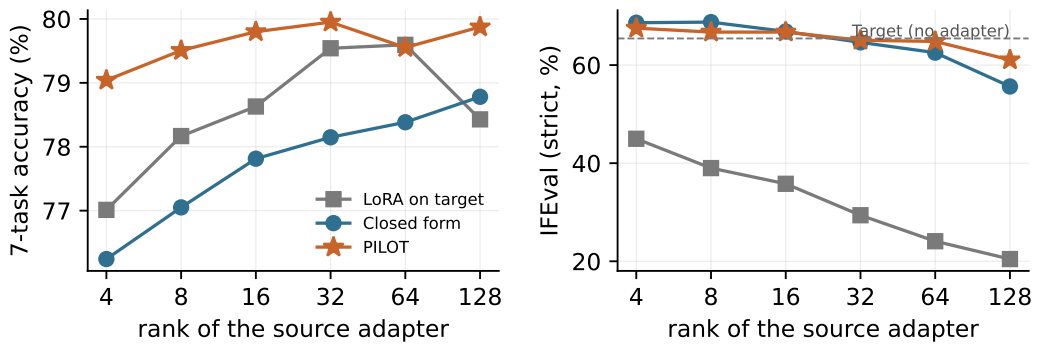

Figure 5: Source adapters of rank 4 to 128 on Qwen3.5-2B, transferred with the frozen operator of one fold and by the closed form alone, against a LoRA trained on the target with the same rank. Each rank reports the mean over three seeds, and rank 16 is the main configuration. The dashed line is the IFEval score of the unmodified target model.

Table 10: Cross-task-group test on Qwen3.5-2B. Source adapters are trained on one answer-format group of the commonsense pool and transferred with a meta-model trained only on adapters and labels from the other group. Group A contains BoolQ, PIQA, SIQA and HellaSwag, and group B contains WinoGrande, ARC and OBQA. BoolQ is not among the seven evaluated tasks, so group-A accuracy averages PIQA, SIQA and HellaSwag. The upper block has two group-B adapters with seeds 42 and 43, scored on the group-B tasks and on the group-A tasks. The lower block has two group-A adapters transferred with a group-B meta-model. Slash-separated values report scores for two adapters.

Group-B adapters, transferred with	Group-B tasks	Group-A tasks	IFEval
Closed-form init only	79.21/79.42	70.20/67.58	68.39/66.91
Group-A meta-model, unseen task group	79.71/79.76	73.55/71.84	68.21/67.84
All-task meta-model of Table 1, seed 42	80.77	71.37	70.06
Target LoRA on the group-B examples, seed 42*	80.07	70.68	43.07
Group-A adapters, transferred with	**Group-B tasks**	**Group-A tasks**	**IFEval**
Closed-form init only	77.10/76.75	77.98/78.04	67.47/66.17
Group-B meta-model, unseen task group	78.24/77.74	78.62/78.47	67.10/65.62

Reproducibility of the learned operators. We measure how much two meta-models agree by the mean squared cosine of the principal angles between the column spaces of their output-side factors P_a, averaged over the 138 modules of Qwen3.5-2B with $d_{\text{out}} \geq 2048$. Two random rank-4 subspaces give 0.001. The three folds, which share the random initialization of P_b and Q_b and differ in their training adapters, have a mean subspace similarity of 0.225. A second meta-model of the seed-44 fold trained from a different initialization on the same data has a subspace similarity of 0.106 with the first. The write directions are thus reproducible far above chance, but they are not unique. The input-side read directions Q_b have a mean subspace similarity of 0.887 across folds and 0.009 across initializations, which suggests that their subspaces are strongly influenced by initialization. Held-out accuracy on the validation set is 80.1 for the first and 79.0 for the second initialization.

E What the correction changes

Table 11 lists the interventions behind Figure 2, and Figure 6 repeats the analysis on Llama-3.2-3B. In panel (a) of both figures, the Mixer rows group the token-mixer projections into query, key and value inputs, the gating inputs of the linear-attention layers and the output projection.

Interventions on Llama-3.2-3B. Figure 6 repeats the analysis of Figure 2 for the seed-44 fold of Llama-3.2-3B. The results follow the same pattern as those for Qwen3.5-2B. On the validation set, the full correction raises accuracy from 74.95 to 82.13, the output-side operators alone reach 81.49 and the input-side operators alone 75.44, the MLP modules alone reach 81.25, and the first third of

Table 11: Interventions on the meta-model of the seed-44 fold of Qwen3.5-2B, applied to its held-out adapter and measured on the 2,048 validation examples with teacher forcing. The correction is applied to the listed subset of modules, and all other modules keep the closed-form approximation. Candidate-selection accuracy counts an example as correct when the logit of the reference candidate is strictly larger than that of every other candidate. The margin is the mean logit difference between the reference and the highest-scoring alternative candidate.

Modules receiving correction	Accuracy	Margin	$\log p$(full answer)
Closed form without correction	74.37	1.52	-0.529
Gain network only	74.46	1.53	-0.528
Operators only	79.10	2.26	-0.508
Full correction	79.49	2.26	-0.509
Output-side operators $I+P_aP_b$ only	78.96	2.04	-0.498
Input-side operators $I+Q_aQ_b$ only	75.73	1.72	-0.522
MLP modules only, 72	79.30	2.09	-0.494
Token-mixer modules only, 114	76.90	1.78	-0.508
Layers 0 to 7 only	77.25	1.76	-0.497
Layers 8 to 15 only	77.69	2.00	-0.516
Layers 16 to 23 only	74.66	1.55	-0.531
All but layers 0 to 7	77.73	2.03	-0.520
All but layers 8 to 15	77.20	1.78	-0.500
All but layers 16 to 23	79.39	2.24	-0.505

the layers alone reaches 80.96. End-of-turn probability behaves differently from candidate-selection accuracy. The mean log-probability of the end-of-turn token rises from -4.74 under the closed form to above -0.01 with the full correction, and the same happens when the correction is applied only to the middle or final third of the layers. Several layer groups therefore increase the end-of-turn probability, while the accuracy gain is localized in the early MLPs.

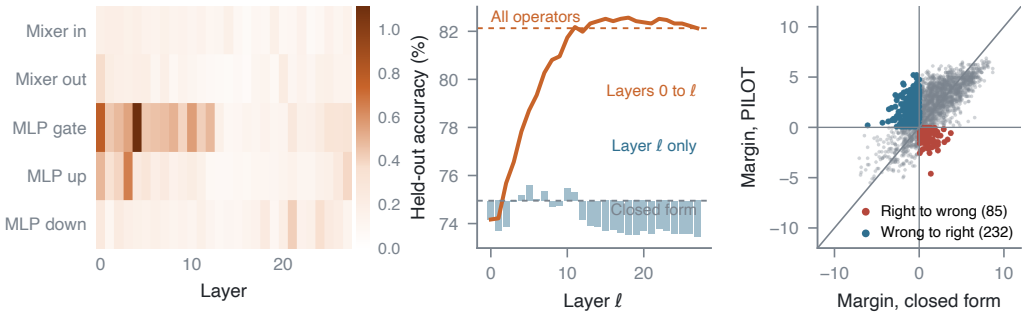

Figure 6: Anatomy of the learned correction on Llama-3.2-3B, with the same panels as Figure 2.

Interventions on Qwen3.5-4B. The 4B pair behaves differently. On the validation set, teacher-forced candidate-selection accuracy does not improve with the correction, with 86.04 for the closed form and 85.50 for the full correction. The end-of-turn probability increases instead. The mean log-probability of the end-of-turn token rises from -14.68 to above -0.01, and the output-side operators or the last third of the layers alone achieve the same. The gain of 1.1 points in test accuracy under free generation on this pair (Table 1) is therefore not explained by candidate-selection accuracy at the first answer token, and we have not isolated its source.

F SHARED-CORRECTION CONTROLS

The one-time stage uses task labels and a retention loss, so its gains could arise from a task correction shared across adapters rather than from the transfer mechanism. We test this with two controls that use the data, objective, KL weight and number of steps of PILOT (Table 12). The first adds one low-rank residual UV per module to the closed-form initialization of every adapter, with about as

Table 12: Shared corrections trained once with the data, objective, KL weight and steps of PILOT. The target-only LoRA does not use a source-adapter input. In the left and middle blocks, labeled data cover the adapter tasks, and the columns report accuracy, IFEval and gen-KL. In the right block, the labels come from task group A only, the correction is applied to a group-B adapter, and accuracy is reported on the tasks of each group. Best in bold and second best underlined.

	Qwen3.5-2B			Llama-3.2-3B			A labels, B adapter	
Method	Acc.	IFEval	KL	Acc.	IFEval	KL	B tasks	A tasks
Closed-form initialization	77.81	**66.85**	**0.0011**	74.27	65.68	<u>0.0013</u>	79.21	70.20
Shared target-only LoRA, r=8	**80.98**	63.77	0.0072	**80.67**	**68.58**	0.0023	76.72	**81.15**
Closed form + additive UV, r=8	<u>80.74</u>	64.20	<u>0.0062</u>	<u>80.23</u>	66.73	0.0024	<u>79.32</u>	<u>77.87</u>
Closed form + operators, PILOT	79.80	<u>66.73</u>	**0.0011**	78.84	<u>68.33</u>	**0.0009**	**79.71**	73.55

many trainable parameters as the operators. The second trains the same residual without any basis, which is a target LoRA that is trained once, does not depend on a source adapter and is reused across all source adapters.

When the labels cover the tasks of the adapters, both controls are more accurate than PILOT, by 0.9 to 1.2 points on Qwen3.5-2B and 1.4 to 1.8 on Llama. The multiplicative operators change the target least, with a gen-KL that is one sixth to two fifths as large and an IFEval score that is 2.5 to 3.0 points higher on Qwen3.5-2B and about equal on Llama. Comparable accuracy gains can be obtained by shared corrections under the same supervision, so the improvement over the closed form does not establish a unique advantage of multiplicative operators. Among the tested corrections, the operators cause the smallest change in the target output distribution. The comparison changes when the supervision does not cover the adapter's tasks. With labels of task group A only, the shared target LoRA reaches 76.72 on the group-B tasks, whereas the closed-form transfer of a group-B adapter reaches 79.21 without ground-truth task labels at transfer time, and both corrections applied to this initialization maintain similar accuracy. In this comparison, the closed-form transfer preserves task accuracy that the tested target-only correction does not recover. Reuse does not save GPU time on these model sizes. In the same-architecture setting, it requires no ground-truth task labels, gradients or optimizer at transfer time for each new adapter (Appendix B).

G TRANSFER WITHOUT MODULE CORRESPONDENCE

Table 13 lists the seven task accuracies behind Table 2, the rank-4 configuration of PILOT-X, and the control using a behavioral basis constructed without a source adapter.

Table 13: Per-task results of cross-architecture transfer with PILOT-X, accuracy in %, notation as in Table 6. ♯Trains a new LoRA per adapter on answers generated by the adapted source. ◇Control without an incoming adapter. ♮Rank 16, selected on the validation set over the rank-4 configuration listed above it. §Meta-trained with 16,384 labeled examples. Every other PILOT row is meta-trained with 4,096, as many as the LoRA reference. °Our adaptation of Command-V to weight-space LoRA (Appendix B). Best in bold and second best underlined, without the unmodified target model. Our rows are reduced to rank 16 except the control without a source adapter, whose adapter is no longer available and is shown at its stored rank.

	Method	PIQA	SIQA	HellaS.	WinoG.	ARC-e	ARC-c	OBQA	Avg.	ΔLoRA	ΔTarget
Llama-3.2-3B → Qwen3.5-4B	*References on the target Qwen3.5-4B-Instruct*										
	Unmodified target	85.26	74.67	84.89	66.30	96.72	90.78	83.60	83.17	−4.49	0.00
	LoRA*	**88.61**	**78.45**	92.94	75.01	97.53	91.95	89.13	87.66	0.00	+4.49
	Source Llama-3.2-3B-Base + LoRA → target Qwen3.5-4B-Instruct										
	Cross-LoRA	85.40	74.53	84.77	66.54	96.76	90.76	83.47	83.17	−4.49	0.00
	Command-V, adapted°	84.51	73.20	84.04	66.53	96.81	91.13	82.13	82.62	−5.04	−0.55
	Unfiltered distillation♯	84.40	70.37	86.76	61.83	96.37	87.29	75.73	80.39	−7.27	−2.78
	TiTok♯ (Jung & Kim, 2026)	83.89	70.90	87.31	61.25	96.65	88.43	77.73	80.88	−6.78	−2.29
	Trans-LoRA♯	85.31	70.78	85.98	64.30	96.37	87.66	77.73	81.16	−6.50	−2.01
	PILOT-X init. only	81.39	71.51	82.53	63.11	96.26	89.36	79.60	80.54	−7.12	−2.63
	PILOT-X, no source adapter◇	83.95	74.56	85.31	64.72	95.88	88.40	82.60	82.20	−5.46	−0.97
	PILOT-X, r=4†	87.67	77.67	92.83	78.22	97.66	92.58	87.87	87.78	+0.12	+4.61
	PILOT-X†♮	87.48	77.72	92.34	78.58	97.60	92.61	89.20	87.93	+0.27	+4.76
	PILOT-X, 16,384 examples†♮§	88.14	78.20	**93.50**	**80.93**	**97.70**	**92.92**	**89.53**	**88.70**	+1.04	+5.53
Qwen3.5-4B → Llama-3.2-3B	*References on the target Llama-3.2-3B-Instruct*										
	Unmodified target	64.47	33.32	29.02	47.12	52.69	36.86	46.20	44.24	−32.07	0.00
	LoRA*	81.83	71.82	84.09	62.17	87.73	75.45	71.07	76.31	0.00	+32.07
	Source Qwen3.5-4B-Base + LoRA → target Llama-3.2-3B-Instruct										
	Cross-LoRA	64.40	33.11	29.17	46.85	52.53	36.43	46.40	44.13	−32.18	−0.11
	Command-V, adapted°	63.55	38.81	38.94	48.38	61.76	44.34	52.93	49.82	−26.49	+5.58
	Unfiltered distillation♯	82.55	73.22	85.52	69.51	88.69	75.31	76.13	78.70	+2.39	+34.46
	TiTok♯ (Jung & Kim, 2026)	78.98	64.74	86.73	53.38	83.70	69.40	61.20	71.16	−5.15	+26.92
	Trans-LoRA♯	80.01	69.13	72.57	54.59	86.56	71.30	68.27	71.78	−4.53	+27.54
	PILOT-X init. only	75.03	64.04	60.16	49.46	82.21	68.55	65.67	66.44	−9.87	+22.20
	PILOT-X, no source adapter◇	2.01	0.00	1.45	2.53	0.08	0.00	0.00	0.87	−75.44	−43.37
	PILOT-X, r=4†	**82.64**	72.77	84.33	71.46	87.47	**76.37**	73.53	78.37	+2.06	+34.13
	PILOT-X†♮	80.09	71.90	85.08	71.72	87.13	74.49	74.47	77.84	+1.53	+33.60
	PILOT-X, 16,384 examples†♮§	82.52	**74.31**	**87.78**	**75.37**	**88.85**	76.11	**77.53**	**80.35**	+4.04	+36.11
Qwen3.5-4B → Gemma-4-E2B	*References on the target Gemma-4-E2B-it*										
	Unmodified target	63.44	40.99	41.09	29.12	60.14	50.68	45.60	47.29	−24.91	0.00
	LoRA*	76.70	69.43	73.13	59.14	87.82	73.81	65.40	72.20	0.00	+24.91
	Source Qwen3.5-4B-Base + LoRA → target Gemma-4-E2B-it										
	Cross-LoRA	63.15	40.63	41.02	29.67	60.75	50.82	45.67	47.39	−24.82	+0.09
	Command-V, adapted°	61.59	30.57	41.54	30.18	47.01	44.68	34.53	41.44	−30.76	−5.85
	Unfiltered distillation♯	**79.18**	**70.85**	78.94	64.72	**89.16**	**75.65**	**70.33**	**75.55**	+3.34	+28.25
	TiTok♯ (Jung & Kim, 2026)	71.87	56.59	**79.51**	53.14	77.84	61.38	48.13	64.06	−8.14	+16.77
	Trans-LoRA♯	70.44	61.14	57.11	50.80	82.56	64.42	58.27	63.53	−8.67	+16.24
	PILOT-X init. only	49.66	33.81	26.88	47.54	27.88	21.08	24.33	33.03	−39.18	−14.27
	PILOT-X, r=4†	76.06	69.46	75.05	62.90	87.35	74.49	65.53	72.98	+0.77	+25.68
	PILOT-X†♮	76.06	68.52	75.39	64.51	87.46	72.81	66.00	72.97	+0.76	+25.67
	PILOT-X, 16,384 examples†♮§	76.97	69.60	77.18	**65.80**	88.48	75.29	68.47	74.54	+2.34	+27.25

H META-TRAINING BUDGET AND ADAPTATION TO A NEW SOURCE

Protocol and accounting. All runs of Section 4.6 keep the objective, the optimizer, the learning rates and the leave-one-adapter-out folds of Appendix B and change the number of labeled examples and steps. The labeled-example budgets are the first 64, 256, 1,024 or 4,096 labeled examples of the 16,384-example meta-training set, so an existing meta-model has seen every example in a smaller budget, and the numbers of labeled examples describe the adaptation stage and not the supervision of the whole system. We evaluate the held-out adapter by candidate-selection accuracy on 2,048 of the 4,096 validation examples, sampled with a fixed seed (Figure 4), and by generation accuracy on the seven test sets. No configuration is selected in this section. The operator rank matches the existing meta-model, 4 for the closed-form pairs and 16 for the cross-architecture meta-model of Gemma, and the run from scratch uses the same rank. The new sources in Figure 3 belong to different model families from the corresponding targets, so their bases come from the behavioral interface with the step-size selection of Appendix B, which uses 128 held-out source answers and no ground-truth task labels. The two directions between Llama-3.2-3B and Qwen3.5-4B are those of Table 2, and Llama-3.2-3B to Qwen3.5-2B and to Gemma-4-E2B are new. A warm start loads the gain network and the operators of the seed-matched fold of the existing meta-model and continues training on the training adapters of the new source. In the reverse direction, the lower block of Table 14, the same-architecture pair starts from the meta-model trained with a cross-architecture source for the same target. On Llama-3.2-3B this is the rank-4 meta-model of the Qwen3.5-4B source trained from scratch. On Qwen3.5-2B and Qwen3.5-4B it is the Llama-source meta-model of the paragraph on further checks, which had itself been initialized from the same-architecture meta-model of the same fold, so these two rows return to the original source after adaptation to the cross-architecture source, and only the Llama row is independent evidence. The held-out adapter is excluded from every training stage.

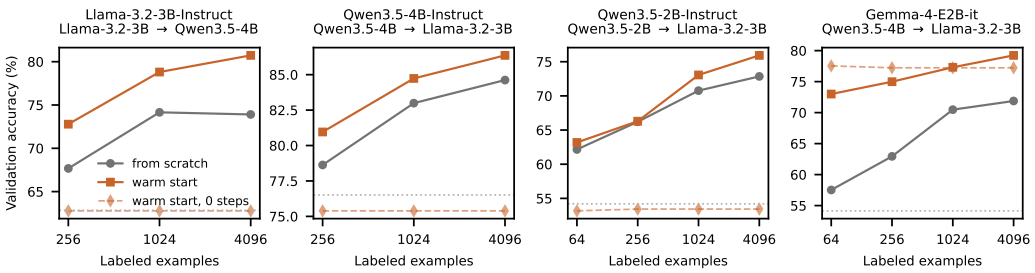

Figure 7: Validation accuracy of the held-out adapter against the number of labeled examples of the 256-step adaptation stage, from scratch and warm-started from the existing meta-model, paired by fold, together with the warm start before adaptation. The dotted line is the basis alone.

Labeled examples and optimization steps in meta-training. The same-architecture pairs were meta-trained from scratch on the seed-44 fold with 256, 1,024, 4,096 and 16,384 labeled examples for 256 steps and with 1,024 and 4,096 labeled examples for 768 steps. On the test sets, 4,096 labeled examples and 256 steps give 78.65, 87.98 and 77.31 on Qwen3.5-2B, Qwen3.5-4B and Llama-3.2-3B, against 79.77, 88.50 and 79.25 for the full budget of the same fold, and 256 labeled examples give 75.74, 86.17 and 75.19. On the validation set for the 2B pair, 256 steps give 74.85, 76.61, 77.10 and 77.39 for the four labeled-example budgets. Training for 768 steps on 4,096 labeled examples raises the validation accuracy on the three pairs from 77.10 to 77.78, 86.38 to 86.82 and 79.39 to 81.69, and changes the test accuracy to 78.76, 87.54 and 78.32, so teacher-forced candidate-selection accuracy on the validation set can improve without a matching improvement in the accuracy of generated answers.

Before adaptation. The bracketed values of Table 14 are the validation accuracy right after loading the existing meta-model, before any step, and the warm start at 0 examples in Figure 3 reports test accuracy for the same transferred adapter. On Gemma the loaded meta-model scores 74.5 on the test sets against 33.8 for the basis alone and 69.3 for the scratch run with 4,096 labeled examples, and on the validation set 256 steps on 256 or 1,024 labeled examples leave validation accuracy nearly unchanged, whereas 4,096 labeled examples raise it by 2 points. On the other three targets the loaded

Table 14: Adaptation to a new source, by target and label budget with 256 steps, paired by fold. Validation is the accuracy on the validation set from scratch → warm start after adaptation, with the warm start before adaptation at 0 steps in brackets. Test and IFEval give 7-task accuracy and IFEval from scratch → warm start. Fold counts in parentheses. The lower block adapts the same-architecture pair from the meta-model trained with a cross-architecture source. The reference is the seed-44 fold of Table 1. The reference for the upper block is PILOT-X of the new pair with 16,384 examples. Test accuracy and IFEval are measured at rank 16, validation accuracy during meta-training at the stored rank.

Target	New source / original source	Examples	Validation [0 steps]	7-task Acc.	IFEval	Ref.
Llama-3.2-3B-Instruct	Qwen3.5-4B-Base / Llama-3.2-3B-Base	256	67.68 → 72.79 [62.78] (3)	70.41 → 72.43 (3)	70.2 → 69.4 (3)	80.35
Llama-3.2-3B-Instruct	Qwen3.5-4B-Base / Llama-3.2-3B-Base	1,024	74.15 → 78.81 [62.78] (3)	73.62 → 75.63 (3)	69.0 → 69.8 (3)	80.35
Llama-3.2-3B-Instruct	Qwen3.5-4B-Base / Llama-3.2-3B-Base	4,096	73.91 → 80.73 [62.78] (3)	74.30 → 77.26 (3)	66.9 → 69.0 (3)	80.35
Qwen3.5-4B-Instruct	Llama-3.2-3B-Base / Qwen3.5-4B-Base	256	78.63 → 80.96 [75.39] (3)	82.54 → 84.37 (3)	80.0 → 79.7 (3)	88.70
Qwen3.5-4B-Instruct	Llama-3.2-3B-Base / Qwen3.5-4B-Base	1,024	82.99 → 84.73 [75.39] (3)	85.90 → 86.76 (3)	77.1 → 78.6 (3)	88.70
Qwen3.5-4B-Instruct	Llama-3.2-3B-Base / Qwen3.5-4B-Base	4,096	84.62 → 86.36 [75.39] (3)	86.63 → 87.77 (3)	78.9 → 78.7 (3)	88.70
Qwen3.5-2B-Instruct	Llama-3.2-3B-Base / Qwen3.5-2B-Base	64	62.16 → 63.18 [53.17] (1)	66.89 → 67.11 (1)	62.5 → 62.7 (1)	80.10
Qwen3.5-2B-Instruct	Llama-3.2-3B-Base / Qwen3.5-2B-Base	256	66.21 → 66.29 [53.43] (3)	70.85 → 71.01 (3)	63.6 → 63.3 (3)	80.10
Qwen3.5-2B-Instruct	Llama-3.2-3B-Base / Qwen3.5-2B-Base	1,024	70.77 → 73.06 [53.43] (3)	73.60 → 74.87 (3)	61.9 → 62.1 (3)	80.10
Qwen3.5-2B-Instruct	Llama-3.2-3B-Base / Qwen3.5-2B-Base	4,096	72.85 → 75.93 [53.43] (3)	76.53 → 77.77 (3)	61.1 → 62.8 (3)	80.10
Gemma-4-E2B-it	Llama-3.2-3B-Base / Qwen3.5-4B-Base	64	57.52 → 73.00 [77.54] (1)	61.39 → 71.71 (1)	77.6 → 77.3 (1)	74.29
Gemma-4-E2B-it	Llama-3.2-3B-Base / Qwen3.5-4B-Base	256	62.94 → 74.98 [77.23] (3)	63.20 → 73.00 (3)	77.0 → 75.8 (3)	74.29
Gemma-4-E2B-it	Llama-3.2-3B-Base / Qwen3.5-4B-Base	1,024	70.48 → 77.31 [77.23] (3)	67.48 → 73.67 (3)	77.3 → 74.9 (3)	74.29
Gemma-4-E2B-it	Llama-3.2-3B-Base / Qwen3.5-4B-Base	4,096	71.88 → 79.23 [77.23] (3)	69.26 → 75.30 (3)	76.6 → 75.5 (3)	74.29
Llama-3.2-3B-Instruct	Llama-3.2-3B-Base / Qwen3.5-4B-Base	64	73.05 → 75.20 [69.92] (1)	74.28 → 74.40 (1)	68.4 → 68.6 (1)	79.25
Llama-3.2-3B-Instruct	Llama-3.2-3B-Base / Qwen3.5-4B-Base	1,024	78.81 → 80.57 [69.92] (1)	75.72 → 77.52 (1)	66.7 → 69.9 (1)	79.25
Llama-3.2-3B-Instruct	Llama-3.2-3B-Base / Qwen3.5-4B-Base	4,096	79.39 → 82.28 [69.92] (1)	77.31 → 78.38 (1)	67.8 → 68.0 (1)	79.25
Qwen3.5-2B-Instruct	Qwen3.5-2B-Base / Llama-3.2-3B-Base	1,024	76.61 → 79.74 [46.68] (1)	77.65 → 78.78 (1)	66.0 → 64.9 (1)	79.77
Qwen3.5-2B-Instruct	Qwen3.5-2B-Base / Llama-3.2-3B-Base	4,096	77.10 → 80.18 [46.68] (1)	78.65 → 79.56 (1)	66.5 → 65.8 (1)	79.77
Qwen3.5-4B-Instruct	Qwen3.5-4B-Base / Llama-3.2-3B-Base	1,024	85.11 → 87.26 [67.63] (1)	87.13 → 88.41 (1)	80.8 → 79.9 (1)	88.50
Qwen3.5-4B-Instruct	Qwen3.5-4B-Base / Llama-3.2-3B-Base	4,096	86.38 → 87.55 [67.63] (1)	87.98 → 89.20 (1)	81.5 → 80.0 (1)	88.50

meta-model is within a point of the basis alone on the test sets and within 2 points on the validation set. The meta-models of these targets were trained on closed-form bases and are applied to behavioral bases, whose directions carry no drift component and whose statistics enter the gain network with a different distribution, and the operators require adaptation to the new basis before accuracy improves. On Gemma, therefore, most of the advantage of the warm start is present before adaptation, and on the other targets it is produced by adaptation.

Further checks. With 1,024 labeled examples and additional training seeds, the warm start improves validation accuracy over training from scratch by 3.8, 4.2 and 5.6 points on Llama-3.2-3B, by 0.9 and 2.5 on Qwen3.5-4B, by 7.6 and 4.3 on Gemma and by 1.8, 2.6 and 2.4 in the reverse direction on Llama. With 64 labeled examples the warm start improves validation accuracy by 1.0 on Qwen3.5-2B, 15.5 on Gemma and 2.2 in the reverse direction on Llama, where the test sets give 74.40 against 74.28. With the full budget of 16,384 labeled examples and 768 steps, the warm start still helps. On Llama-3.2-3B with the Qwen3.5-4B source it reaches 83.54 on the validation set against 80.81 from scratch and 80.24 on the test sets, and on Qwen3.5-4B with the Llama source 89.43 on the test sets. In the reverse direction on the seed-44 fold, the warm-started same-architecture meta-model reaches 89.20 on Qwen3.5-4B with 4,096 labeled examples, which is above the 88.50 of the full budget on that fold, with the round-trip caveat above. Over the 45 matched pairs of runs, IFEval of the warm start is at or above that of the scratch run in 23 and below it in 22, with differences between -3.1 and $+4.1$ points, and on Gemma it is below in 8 of 10.

Target-only control. To ask whether the warm start reuses a transfer rule or task knowledge of the target, we repeat the protocol with the shared target-only LoRA of Section F, which has rank 8, no source-adapter input and the same objective. At 1,024 labeled examples on the seed-44 fold, the target-only LoRA warm-started from its full-budget counterpart of the same-architecture pair reaches 79.64 on the validation set and 79.75 on the test sets on Qwen3.5-2B with the Llama source, against 75.15 and 76.58 from scratch, and 83.64 and 80.11 on Llama-3.2-3B with the Qwen3.5-4B source, against 79.69 and 75.98. These gains are at least as large as those of the meta-model on the same folds, which are 3.8 and 1.2 points on Llama and 1.9 and 1.3 on Qwen3.5-2B, and the loaded control already scores 80.86 and 82.47 on the validation set before any step, because it directly encodes the supervised task correction. Its generation KL after adaptation is 0.010 on Qwen3.5-2B against 0.004 from scratch. In these matched comparisons, a supervised target-side correction benefits at least as much from warm starting as the meta-model, so the warm-start advantage measures what was learned

Table 15: Ablation and sensitivity on one held-out Qwen3.5-2B adapter, full test sets. λ is the KL weight and r the operator rank. The last two rows evaluate an unrelated math adapter on the commonsense tasks. No variant was used to select the main configuration.

Variant	λ	r	Avg.	IFEval	gen-KL
Closed-form initialization only, $g{=}g^0$	n/a	n/a	77.48	65.43	0.0011
Gain network π_θ only, without P, Q	5	0	77.57	67.10	0.0011
PILOT, 16,384 examples	5	4	79.77	67.47	0.0010
PILOT	1	4	79.59	66.17	0.0015
PILOT	20	4	79.43	67.84	0.0008
PILOT	5	1	79.47	65.80	0.0012
PILOT	5	16	80.05	65.99	0.0011
Math adapter, closed-form init only	n/a	n/a	59.62	65.43	0.0010
+ commonsense-trained operator	5	4	62.05	63.77	0.0010

about the target and the task family under supervision and does not separate the transfer rule from task knowledge.

I ABLATION AND PARAMETER SENSITIVITY

Table 15 reports the ablation and parameter-sensitivity results discussed in Section 4.7, covering the learned correction, the KL weight λ, the operator rank r and the math-adapter control on one held-out Qwen3.5-2B adapter.