JEVSOUP: SYSTEM-ONE ROUTING FOR TRAINING-FREE LORA COMPOSITION

Xiuying Wang[1,†], *Jiahua Cheng*[1,†], *Shuotian Li*[2,†], *Yufan Cheng*[3], *Bowen Deng*[1], *Zhexuan Bai*[1], *Yichen Li*[4,*]

[1]Beijing University of Posts and Telecommunications [2]University of Malaya
[3]Georgian College [4]Huazhong University of Science and Technology

ABSTRACT

Building adaptable AI systems requires effective coordination of specialized capabilities across diverse tasks. Low-rank adaptation (LoRA) enables modular expertise, but existing routing approaches may require auxiliary data, additional training, or autoregressive decoding. We propose JevSoup, a training-free framework separating System One expert routing from System Two execution. Using only the input and expert descriptions, Jev selects two experts through structured probabilities. JevSoup retains the leading expert's update, projects the second onto the orthogonal complement of the first update's row space, and combines them with equal weights. Across 14 PorTAL tasks and three Qwen3 scales, JevSoup achieves absolute gains of up to 1.19% in task-macro and 1.21% in sample-micro accuracy over the strongest evaluated external baselines. Our code is available at `https://github.com/Leowang980/JevSoup`.

***Index Terms*—** System One models, LoRA, expert routing, training-free adaptation

1. INTRODUCTION

System One and System Two describe complementary aspects of cognition: judgments that arise with little conscious effort and reasoning that requires sustained attention [1]. This distinction offers an analogy for how AI assistants coordinate specialized capabilities. A customer-support assistant, for example, may need billing, product-policy, or troubleshooting expertise depending on the request [2]. Low-rank adaptation (LoRA) [3] provides a practical way to make such expertise modular, representing specialized capabilities as lightweight adapters to a shared language model. As these adapters accumulate into an expert library, using them effectively requires deciding which experts are relevant to each input and how their contributions should be combined.

To make effective use of these expert libraries, prior work has explored a range of routing and composition strategies. MoLE [4] and related gated mixtures [5] train expert selection networks, while LoraRetriever [6] learns an instruction-tuned retriever. Without router training, methods can still use auxiliary data to select components for averaging, as in Model Soups [7] and AdapterSoup [8], build representations for retrieval [9, 10], or optimize mixture weights without gradients, as in LoraHub [11].

However, representative examples may be unavailable, and learned routers require additional training and may need updating as the expert library evolves. LoGo [12] and Arrow [13] avoid these dependencies using adapter activations and prototypes derived from adapter weights, respectively. These signals tie routing to the backbone and leave task relevance implicit. LLMs can instead match inputs to explicit expert descriptions [14]. This approach still requires autoregressive decoding and parsing. Jev [15], a pretrained System One model, offers a structured alternative by mapping the input and expert descriptions directly to probabilities over expert identifiers. Yet selection alone does not determine how to combine the chosen adapters. Routing probabilities need not be suitable mixture weights, and direct averaging does not explicitly control overlap between parameter updates. Inspired by the use of orthogonal projection to limit interference in continual learning [16], we treat composition as a separate problem from expert selection.

To this end, we propose JevSoup, a training-free framework that separates System One expert selection from System Two task execution. Jev selects an ordered pair of experts using only the input and their capability descriptions. For composition, JevSoup retains the higher-ranked update and projects the second onto the orthogonal complement of the first update's row space. The resulting updates are combined with equal weights in a single frozen backbone. This requires neither expert training samples nor additional router or expert training.

Our main contributions are threefold:

- We formulate LoRA-based inference as System One routing and System Two execution, identifying dependence on training samples, router retraining, and autoregressive decoding as limitations of existing routing approaches.

[†]These authors contributed equally. [*]Corresponding author.

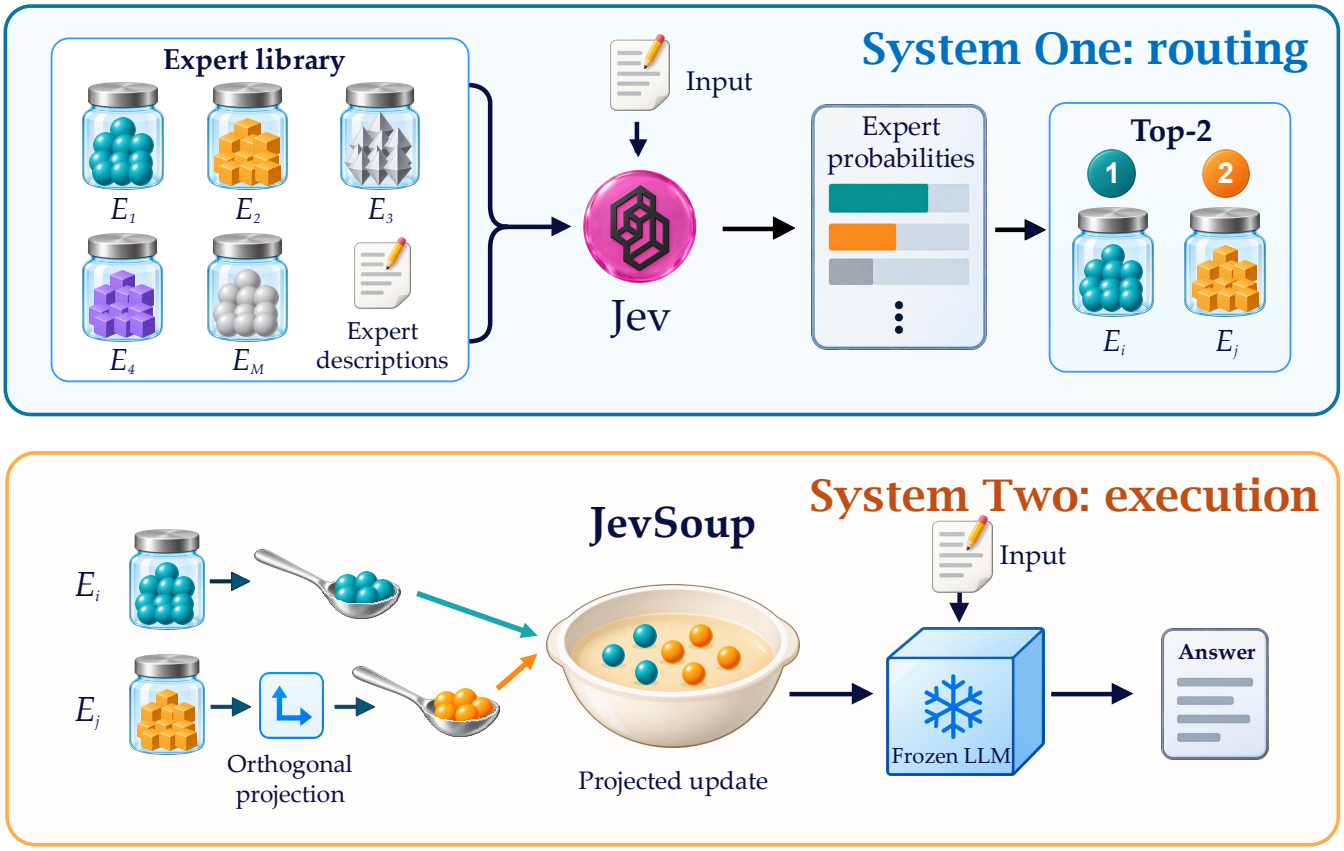

Fig. 1. Overview of JevSoup. Top (System One): Jev uses the input and expert descriptions to rank the experts and select an ordered Top-2 pair. Bottom (System Two): The second LoRA update is orthogonally projected relative to the first, and the two complete updates are equally combined in the frozen language model to produce the answer. Probability bars are schematic.

- We propose JevSoup, a training-free framework that combines Jev-based structured selection with an asymmetric composition rule: retain the higher-ranked update, orthogonalize the second against its row space, and fuse them with equal weights.
- Across 14 tasks and three model scales, JevSoup achieves absolute gains of up to 1.19% in task-macro and 1.21% in sample-micro accuracy over the strongest evaluated external baselines.

2. METHODOLOGY

2.1. Problem Definition

Let f_{θ_0} be a frozen language model. We consider a library of M compatible LoRA experts, $\mathcal{E} = \{E_1, E_2, \ldots, E_M\}$. All experts have frozen LoRA parameters and adapt the same weight matrices of the backbone. Each expert E_i is associated with an expert card c_i, containing its identifier and a natural-language description of its capabilities.

For an adapted weight matrix W_0, expert E_i contributes the complete LoRA [3] update

$$\Delta W_i = s_i B_i A_i, \qquad s_i = \alpha_i / r_i, \tag{1}$$

where $A_i \in \mathbb{R}^{r_i \times d_{\text{in}}}$ and $B_i \in \mathbb{R}^{d_{\text{out}} \times r_i}$ are frozen factors, and r_i and α_i denote the rank and scaling hyperparameter. We omit matrix indices; the same definitions apply independently at each adapted matrix.

Given an input x and the expert library $\mathcal{E}$, our goal is to select relevant experts and compose their updates into an input-dependent update $\Delta W(x)$, yielding adapted weights $W_0 + \Delta W(x)$ for task execution. In our multiple-choice evaluation, x contains the prompt and all candidate answers. Gold answers and task labels are unavailable to routing.

2.2. JevSoup Framework

JevSoup implements this selection–composition pipeline in two stages. System One routing identifies an ordered pair of experts from their capability descriptions. System Two execution composes their updates and uses them in a single frozen backbone. The framework is illustrated in Fig. 1.

System One routing. Expert selection can be expressed as matching the input to the capabilities described in the expert cards. We use Jev [15], a pretrained System One model, to make this decision through a typed choice interface rather than generate a textual expert name. Jev is instructed to select the most suitable expert, not solve the task. Conditioned on the fixed cards, the router R_ϕ returns a probability vector over expert identifiers:

$$\mathbf{p}(x) = R_\phi(x), \qquad p_i(x) \geq 0, \quad \sum_{i=1}^{M} p_i(x) = 1. \quad (2)$$

After validating the response and normalizing its probabilities, we select the ordered Top-2 pair (a, b), with $p_a(x) \geq p_b(x)$, breaking ties by expert identifier. This decision is made once per input and shared across all adapted matrices and candidate-answer evaluations.

System Two execution. In continual learning, Orthogonal Gradient Descent (OGD) [16] addresses interference by projecting new-task loss gradients away from the span of stored gradients of past-task model outputs. Inspired by this principle, we retain one expert update and remove the component of the other that overlaps its row space. Unlike OGD, this operation acts on frozen LoRA updates without computing gradients or performing training.

The routing order determines which update to retain: the higher-ranked expert a serves as the anchor, and only expert b is projected. We omit the input dependence below. Let $k_a = \mathrm{rank}(\Delta W_a)$ and $Q_a \in \mathbb{R}^{d_{\mathrm{in}} \times k_a}$ have orthonormal columns spanning the row space of the complete update ΔW_a. Then $Q_a Q_a^\top$ projects onto input-space directions used by this update. Let I denote the $d_{\mathrm{in}} \times d_{\mathrm{in}}$ identity matrix. With projection strength $\lambda \in [0, 1]$, the second update becomes

$$\widetilde{\Delta W}_b(\lambda) = \Delta W_b\big(I - \lambda Q_a Q_a^\top\big), \quad (3)$$

We then combine the anchor and projected update with equal weights:

$$\Delta W(\lambda) = \tfrac{1}{2}\big(\Delta W_a + \widetilde{\Delta W}_b(\lambda)\big). \quad (4)$$

Equal weighting separates expert ranking from mixture magnitude without learning additional coefficients. Each update retains its original LoRA scaling. We do not restore the projected update's norm or renormalize the branch weights, so projection can change both direction and magnitude. The composed update is applied through two low-rank branches within one backbone, not by averaging the A and B factors independently or ensembling separate model predictions.

To implement this composition without forming dense update matrices, we apply the projection directly to the second expert's low-rank factor:

$$\widetilde{A}_b(\lambda) = A_b - \lambda(A_b Q_a) Q_a^\top, \quad (5)$$

with B_b and the original scale unchanged. We obtain Q_a from the leading k_a right singular vectors of $R_a A_a$, where R_a is the triangular factor of the reduced QR decomposition of B_a.

3. EXPERIMENTS

3.1. Experiment Setup

Benchmark. PorTAL [17] provides task-specialized LoRA experts covering language understanding, question answering, and commonsense reasoning. We evaluate on 19,477 multiple-choice examples across its 14 tasks and report task-macro and sample-micro accuracy. Our evaluation set is drawn from PorTAL's released validation split, with development prompts excluded. The main comparison, ablations, and sensitivity analysis use the same evaluation examples.

Baseline. Base is the unadapted backbone. **Adaptive Minds (AM)** [14] is adapted to select two experts with the unadapted Qwen3 backbone; a keyword fallback fills missing IDs, and their complete LoRA updates are equally weighted. **LoGo** [12] ranks and weights experts by last-block query-projection update norms. Our **AdapterSoup** [8] variant retrieves with Qwen3-Embedding-0.6B [18] using 100 training examples per expert after excluding evaluation overlaps, then averages LoRA factors. **Arrow** [13] routes with singular-vector prototypes and combines LoRA factors with softmax weights. All routing methods use Top-2. All baselines share our backbone, expert library, and scoring protocol; these are task adaptations, not replications of the original absolute scores.

Configuration. We use Qwen3-1.7B, 4B, and 8B [19] with 14 frozen PorTAL experts per backbone. LoRA adapts query/value projections with rank 8 and $\alpha = 16$. We construct description-only expert cards from task definitions, with no training examples, and use the same cards for Jev and AM. We use Jev version `jev-1.13.0`[1] with card ordering fixed by seed 42. Unless otherwise specified, we set the projection strength λ to 1. Inference uses BF16, seed 42, a 768-token prompt limit, and one candidate per forward pass. Answers are scored by character-normalized continuation log-probability. Current experiments run on an NVIDIA RTX PRO 6000 GPU.

3.2. Main Results

Among completed results in Table 1, JevSoup has the highest task-macro and sample-micro accuracy at all three scales. Relative to Base, it gains 12.86, 11.25, and 9.14 macro points at 1.7B, 4B, and 8B. AM Top-2 is the strongest external method in macro accuracy at all three scales; JevSoup exceeds it by 1.13, 1.19, and 0.21 points, respectively. Its micro accuracy is also higher than each external baseline at all three scales. The Jev-only control is examined alongside AM Top-2 in Table 2 to separate routing from orthogonal composition.

[1] `https://docs.typesafe.ai/models`

Table 1. Main comparison on PorTAL (accuracy, %). Macro and Micro denote task- and sample-averaged accuracy. Best and second-best values in each column are bold and underlined, respectively. At each model scale, all routing methods share the same backbone, expert library, and evaluation protocol.

	Qwen3-1.7B		Qwen3-4B		Qwen3-8B	
Method	Macro	Micro	Macro	Micro	Macro	Micro
Base	58.05	61.79	64.05	69.76	68.15	74.41
Adaptive Minds [14]	<u>69.79</u>	<u>67.15</u>	<u>74.11</u>	72.49	<u>77.07</u>	<u>78.54</u>
LoGo [12]	68.74	64.96	71.43	69.75	75.94	77.01
AdapterSoup [8]	69.18	66.85	73.61	<u>72.60</u>	76.86	77.87
Arrow [13]	69.12	65.81	72.65	71.27	76.45	77.11
JevSoup	**70.91**	**67.40**	**75.30**	**73.81**	**77.29**	**78.73**

Table 2. 4B ablation (accuracy, %). Random Top-2 reports mean ± standard deviation over three routing seeds. Prob. denotes routing probabilities renormalized over the selected pair. Proj. indicates whether full projection ($\lambda = 1$) is applied to the second expert's update.

Selection	Weights	Proj.	Macro	Micro
Random Top-2	Equal	No	72.43 ± 0.34	71.77 ± 0.16
Jev Top-1	Single	–	73.24	71.83
Jev Top-2	Prob.	No	74.45	72.33
Jev Top-2	Prob.	Yes	<u>74.86</u>	72.41
AM Top-2	Equal	No	74.11	72.49
Jev Top-2	Equal	No	74.64	<u>73.49</u>
JevSoup	Equal	Yes	**75.30**	**73.81**

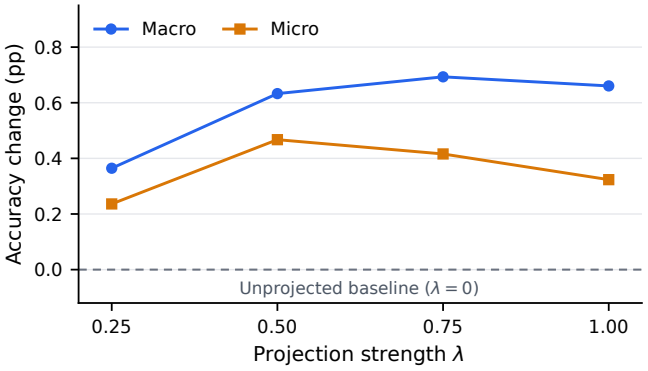

Fig. 2. 4B sensitivity to projection strength. Point-estimate accuracy changes relative to the unprojected control ($\lambda = 0$, dashed line), with fixed routing and equal weights.

3.3. Ablation Study

Table 2 examines seven 4B configurations: Random Top-2, AM Top-2, Jev Top-1, and four Jev Top-2 variants with equal or probability weights, each with or without projection. AM Top-2 and unprojected Jev Top-2 use the same equal-weight composition, isolating expert selection. The equal-weight projected variant is JevSoup. All use the same backbone, expert pool, and scoring protocol with model seed 42; Random Top-2 averages over routing seeds 42–44.

Under equal weights and without projection, Jev Top-2 exceeds Random Top-2 by 2.21 macro and 1.72 micro points, and AM Top-2 by 0.53 and 1.00 points. These comparisons support Jev-based selection under matched Top-2 fusion. Jev Top-2 also exceeds Jev Top-1 by 1.40 macro and 1.66 micro points. Equal weights outperform probability weights both with and without projection, especially on micro accuracy. Overall, JevSoup achieves the highest macro (75.30%) and micro (73.81%) accuracy, 0.66 and 0.32 points above the unprojected equal-weight Jev Top-2 variant. Together, these results support the combination of Jev-based Top-2 selection, equal weighting, and ordered projection in JevSoup.

3.4. Sensitivity Analysis

Figure 2 varies $\lambda \in \{0.25, 0.5, 0.75, 1\}$ with fixed routing and equal weights. All four settings yield positive point-estimate gains over $\lambda = 0$. Macro accuracy spans 75.01–75.34% and micro accuracy 73.72–73.95%, varying by 0.33 and 0.23 percentage points, respectively, on this evaluation set. These small variations suggest that JevSoup is relatively insensitive to projection strength over the tested range. We retain $\lambda = 1$ for the main results.

4. CONCLUSION

We presented JevSoup, a training-free framework that separates System One expert routing from System Two task execution. Jev selects an ordered pair using only the input and expert descriptions; composition retains the first update, projects the second onto the orthogonal complement of the first update's row space, and combines them with equal weights. Across 14 PorTAL tasks and three Qwen3 model scales, JevSoup achieves the highest task-macro and sample-micro accuracy among the evaluated methods.

5. ACKNOWLEDGMENT

No funding was received for this study. The authors have no relevant financial or nonfinancial interests to disclose.

6. COMPLIANCE WITH ETHICAL STANDARDS

No ethical approval was required for this study.

7. REFERENCES

[1] Daniel Kahneman, Patrick Egan, et al., *Thinking, fast and slow*, vol. 1, Farrar, straus and Giroux New York, 2011.

[2] Quan Shi, Alexandra Zytek, Pedram Razavi, Karthik Narasimhan, and Victor Barres, "τ-Knowledge: Evaluating conversational agents over unstructured knowledge," *arXiv preprint arXiv:2603.04370*, 2026.

[3] 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," *arXiv preprint arXiv:2106.09685*, 2021.

[4] Xun Wu, Shaohan Huang, and Furu Wei, "Mixture of lora experts," *arXiv preprint arXiv:2404.13628*, 2024.

[5] Jingwei Xu, Junyu Lai, and Yunpeng Huang, "Meteora: Multiple-tasks embedded lora for large language models," in *International Conference on Learning Representations*, 2025, vol. 2025, pp. 55049–55076.

[6] Ziyu Zhao, Leilei Gan, Guoyin Wang, Wangchunshu Zhou, Hongxia Yang, Kun Kuang, and Fei Wu, "Loraretriever: Input-aware lora retrieval and composition for mixed tasks in the wild," in *Findings of the Association for Computational Linguistics: ACL 2024*, 2024, pp. 4447–4462.

[7] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al., "Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time," in *International conference on machine learning*. Pmlr, 2022, pp. 23965–23998.

[8] Alexandra Chronopoulou, Matthew E Peters, Alexander Fraser, and Jesse Dodge, "AdapterSoup: Weight averaging to improve generalization of pretrained language models," in *Findings of the Association for Computational Linguistics: EACL 2023*, 2023, pp. 2054–2063.

[9] Ziyi Han, Huanyu Wang, Zeyu Zhang, Xiangxiang Dai, Xutong Liu, and John Lui, "Hilora: Adaptive hierarchical lora routing for training-free domain generalization," *arXiv preprint arXiv:2510.12266*, 2025.

[10] Akash Dhasade, Anne-Marie Kermarrec, Igor Pavlovic, Diana Petrescu, Rafael Pires, Mathis Randl, and Martijn de Vos, "Effective lora adapter routing using task representations," *arXiv preprint arXiv:2601.21795*, 2026.

[11] Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin, "Lorahub: Efficient cross-task generalization via dynamic lora composition," *arXiv preprint arXiv:2307.13269*, 2023.

[12] Seungeon Lee, Soumi Das, Manish Gupta, and Krishna P Gummadi, "LoRA on the go: Instance-level dynamic LoRA selection and merging," in *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2026, pp. 39583–39601.

[13] Oleksiy Ostapenko, Zhan Su, Edoardo Maria Ponti, Laurent Charlin, Nicolas Le Roux, Matheus Pereira, Lucas Caccia, and Alessandro Sordoni, "Towards modular LLMs by building and reusing a library of LoRAs," *arXiv preprint arXiv:2405.11157*, 2024.

[14] Pavan C Shekar and Aswanth Krishnan, "Adaptive minds: Empowering agents with LoRA-as-tools," *arXiv preprint arXiv:2510.15416*, 2025.

[15] Diogo Almeida, "Introducing System One Models & Jev," TypeSafe AI blog. `https://typesafe.ai/blog/introducing-system-one-models-and-jev`, Sept. 2026, Published September 15; accessed September 23, 2026.

[16] Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li, "Orthogonal gradient descent for continual learning," in *International conference on artificial intelligence and statistics*. PMLR, 2020, pp. 3762–3773.

[17] Benjamin Geist, "PorTAL: Portable task adapters for LLMs," Software, version 0.2.0. `https://github.com/ramp-public/portallib`, Accessed September 23, 2026.

[18] Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, et al., "Qwen3 embedding: Advancing text embedding and reranking through foundation models," *arXiv preprint arXiv:2506.05176*, 2025.

[19] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al., "Qwen3 technical report," *arXiv preprint arXiv:2505.09388*, 2025.