PARAMETRIC MEMORY DECODING FOR ZERO- SHOT ROUTING IN LORA-BASED EXTERNAL PARAMETRIC MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

With the rise of parametric memory, LoRA-based External Parametric Memory (EPM) has emerged as a modular solution, but existing routing methods often introduce additional training, deployment, and maintenance overhead. This raises a natural question: can a LoRA-based EPM bank be routed without maintaining an additional routing component? However, existing zero-shot LoRA routing methods still face two problems under the EPM setting: (1) their evaluations are scattered across different task settings rather than organized around EPM access, and (2) their routing signals lack a unified perspective to guide systematic improvement. To address these problems, we organize ***PMD-Bench***, covering document-level, domain-level knowledge, and task-skill, and propose ***Parametric Memory Decoding*** (PMD), the first framework designed to systematically improve zero-shot LoRA routing by reframing it as decoding activations over external parametric memory. Based on PMD, we further instantiate ***PMDRouter***, which scores each LoRA by its response magnitude from a single base-model prefill. Experiments on PMD-Bench show that PMDRouter achieves the strongest internal-signal performance across multiple zero-shot routing settings, demonstrating the feasibility of zero-shot LoRA routing and the potential of PMD as a general framework for improving zero-shot routing methods. Sources: Github.

1 INTRODUCTION

With the development of large language models, Large language model memory has attracted increasing attention (Zhang et al., 2026d; Wei et al., 2025; Zhang et al., 2025; Shazeer et al., 2017; Fedus et al., 2022; Yang et al., 2026b). A recent form of model memory is parametric memory, which enables implicit and generalizable access to stored knowledge but is hard to update or modularize in monolithic models, motivating *external parametric memory* (EPM) with independent parameter modules attached to a frozen backbone (Back et al., 2026; Mallen et al., 2023; Lewis et al., 2020). Since EPM consists of multiple lightweight parameter modules, access becomes a routing problem: selecting which LoRA memory unit to activate for each query (Bini et al., 2025). As illustrated in Figure 1(a,b), existing routing methods can be broadly divided into three categories by Lora routing methods, among which routing networks and retrieval-augmented routing rely on either learned routers or external representation spaces for adapter selection (Chen et al., 2024; Gao et al., 2025; Zhao et al., 2024b; Wang et al., 2025). While effective, these components add additional training, deployment, and maintenance overhead, motivating our question: Can a LoRA memory bank be accessed by a zero-shot policy?

As illustrated by the zero-shot paradigm in Figure 1 (c), this intrinsic access setting attempts to select LoRA modules that explore this direction by designing routing signals from the query, the backbone, and observable LoRA-side information. Representative methods such as Arrow, SEQR, HiLoRA, SpectR, and LAG derive routing signals from task vectors, adapter-side statistics, or internal model states (Wang et al., 2022; Han et al., 2025; Cao et al., 2025). These methods show that LoRA routing can be performed without an additional trained router, but two issues remain under the EPM setting: **Evaluation:** fragmented memory forms and access assumptions lead to inconsistent characterization across memory granularities; **Method:** routing signals are often constructed by searching many query-side, LoRA-side, and scoring choices, yet remain poorly separable for similar memory units.

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.

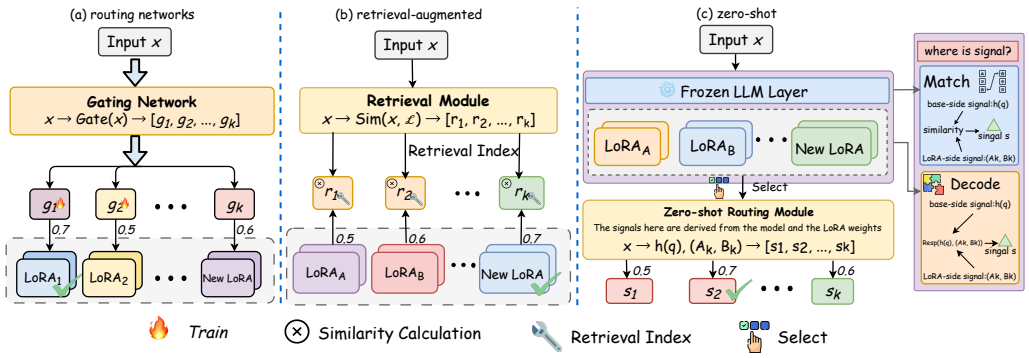

Figure 1: **Three LoRA routing paradigms**: (a) learned routing networks, (b) retrieval-augmented routing, and (c) zero-shot LoRA routing, where scores are computed from the query-side signal $h(q)$ and observable LoRA weights (A_k, B_k). PMD defines the routable response object as $\rho_k(q) = \mathrm{Resp}(h(q), (A_k, B_k))$, and decodes it as $s_k = D(\rho_k(q))$.

These two issues point to two concrete problems in current zero-shot LoRA routing: **Evaluation.** Existing evaluations are not organized around a unified protocol of external parametric memory access. Many LoRA routing benchmarks are built around task-skill adapter banks, while others focus on document- or knowledge-oriented retrieval settings (Zhao et al., 2024a). At the same time, terms such as training-free, data-free, and zero-shot are often used inconsistently (Xia et al., 2025; Lu et al., 2025; Lazier et al., 2025; Fleshman & Van Durme, 2025c; Zhang et al., 2026c;b); here, zero-shot LoRA routing excludes routing-specific training data, retrieval indexes, and external representation spaces. **Method.** Existing zero-shot routers also lack a compact principle for constructing routing signals. In practice, they often search over many query-side representations, LoRA-side descriptors, scoring rules, layers, modules, and normalization choices (Dhasade et al., 2026; Fleshman & Van Durme, 2025b; Han et al., 2025). While effective in some settings, these signals can be difficult to separate when candidate LoRA memories are trained on similar domains, related tasks, or overlapping documents.

To address these problems, we first organize ***PMD-Bench***, a multi-granularity benchmark for EPM access that covers PAPERQA for document-level memory, NQ-DOMAINLORA for domain-level knowledge memory, and TASK-LORA for task-skill memory. We then propose ***Parametric Memory Decoding*** (PMD), the first framework designed to systematically improve zero-shot LoRA routing by reframing it as decoding activations over external parametric memory. Based on PMD, we instantiate ***PMDRouter***, which takes semantic activations from a single base-model prefill and scores each LoRA by its query-conditioned response magnitude. Experiments on PMD-Bench show that PMDRouter achieves the strongest internal-signal performance across multiple zero-shot routing settings and stable gains across diverse memory granularities. Overall, these results demonstrate the feasibility of zero-shot LoRA routing and suggest that PMD can serve as a general framework for improving zero-shot routing methods.

Our contributions are threefold: (1) We organize **PMD-Bench**, the multi-granularity benchmark for external parametric memory routing. It covers document-level, domain-level knowledge, and task-skill memory with 35 memory-unit partitions in total (Section 3.2). (2) We introduce **Parametric Memory Decoding** (PMD), which formulates zero-shot LoRA routing as decoding over external parametric memory rather than query-to-adapter matching. PMD reduces the estimated routing-design complexity by about $40\times$ while improving routing accuracy (Sections 4.2, 5.5). (3) We propose **PMDRouter**, a parameter-free linear-response router that scores LoRAs by response energy from a single backbone prefill. PMDRouter achieves the strongest internal-signal zero-shot performance in most settings on PMD-Bench, with gains up to 63.8 points on Task-LoRA (Sections 4, 5.2).

2 RELATED WORK

External Parametric Memory Model parameters can serve as memory for factual knowledge, task skills, behavioral patterns, and domain preferences (Zhang et al., 2024; Wadhwa et al., 2024; Wang et al., 2024; Qiao et al., 2026; Sprechmann et al., 2018; Heinzerling & Inui, 2021; Valadarsky

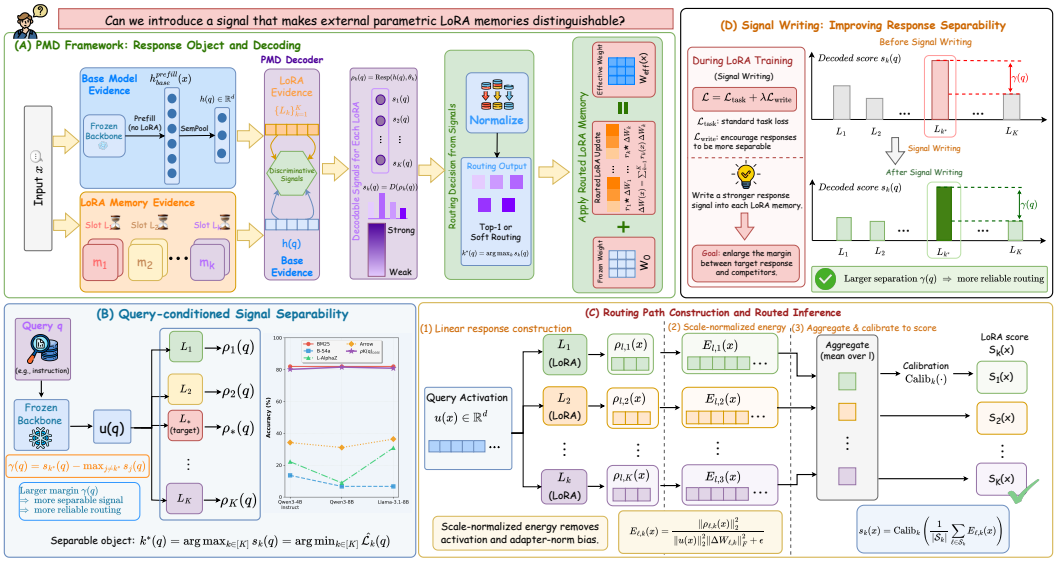

Figure 2: Overview of PMD and PMDRouter. (A) PMD Framework: Response Object and Decoding; (B) Query-conditioned Signal Separability; (C) Routing Path Construction and Routed Inference; (D) Signal Writing: Improving Response Separability.

et al., 2017; Zhang et al., 2026a; Li et al., 2026b; 2024b). PEFT methods such as adapters and LoRA further externalize this memory into compact, modular, and composable parameter units.

LoRA Routing Existing LoRA routing methods access a LoRA bank through different mechanisms, but under different assumptions. Some rely on trained routers, gating networks, or layer-/token-level expert allocation (Chen et al., 2024; Buehler & Buehler, 2024; Araujo et al., 2024; Gao et al., 2025; Li et al., 2025a; Zhuang et al., 2025; Sawada et al., 2020; Cai et al., 2021; Muqeeth et al., 2024; Huang et al., 2026); others use task representations or external embedding spaces to retrieve or assign LoRA modules (Kong et al., 2024; Page-Caccia et al., 2023; Jin et al., 2026; Shukla et al., 2026; Wu et al., 2024; He et al., 2023; Yang et al., 2025; Li et al., 2024a; Prabhakar et al., 2025; Cao et al., 2026b). While effective, these methods mix memory content with extra routing capacity, and therefore do not isolate whether the memory bank itself is accessible from internal query signals alone. To isolate whether the memory bank is intrinsically decodable, one must focus on the zero-shot, zero-shot-based setting considered here, rather than related directions such as data-free LoRA transfer, query-adaptive fusion, LoRA-augmented generation, and LoRA reuse (Chen et al., 2026; Ponti et al., 2023; Tian et al., 2025; Zhang & Li, 2024; Li et al., 2025c; Zhang & Zhou, 2025; Su et al., 2025; Li et al., 2025b; Cao et al., 2026a; Li et al., 2026a; Ji et al., 2026a; Cui et al., 2026; Ji et al., 2026b).

zero-shot LoRA routing Zero-shot LoRA routing focuses on constructing adapter-selection signals from the query, the frozen backbone, and observable LoRA-side information. Representative methods include Arrow, SEQR, HiLoRA, SPECTR, and LAG, which use task vectors, adapter statistics, spectral descriptors, activation or textual proxies, and other model-internal signals (Wang et al., 2022; Fleshman & Van Durme, 2025b; Han et al., 2025; Fleshman & Van Durme, 2025c;a; Cao et al., 2025; Jeong et al., 2026; Kappiyath et al.; Yang et al., 2026a). While effective, these methods mostly follow static or proxy query adapter matching, which can be sensitive to design choices and poorly separable for semantically close LoRA memories.

3 Zero-shot LoRA Routing Reformulation

3.1 Problem Reformulation

We study zero-shot LoRA routing for LoRA-based external parametric memory. Given a frozen backbone f_{base} and a bank of post-attached LoRA adapters $\mathcal{M} = \{L_k\}_{k=1}^{K}$, each adapter L_k acts as

a parameterized memory unit that may encode domain knowledge, task skills, or their hybrid form. Given a query q, the router must select which adapter to activate. In this setting, the routing rule is constructed without learning a separate router: for each query, the query-dependent routing score is computed from frozen-backbone activations and observable LoRA weights, under a fixed scoring rule rather than an external retrieval representation.

Formally, let the k-th LoRA module be parameterized by low-rank factors $\theta_k = (A_k, B_k)$, with update $\Delta W_k = B_k A_k$. The query-side signal is obtained from the frozen backbone through a fixed parameter-free aggregation $h(q) = \mathcal{A}(\{h_\ell^{\text{base}}(t)\}_{t=1}^{T})$, where $\mathcal{A}$ may denote last-token pooling, mean pooling, or another fixed aggregation over the backbone trajectory. We then define a parameter-free access score $\mathcal{R}(h(q), A_k, B_k)$ and select the accessed memory slot by

$$k^*(q) = \arg\max_{k \in [K]} \mathcal{R}\big(h(q), A_k, B_k\big). \tag{1}$$

3.2 PMD-BENCH

To instantiate external parametric memory access in representative scenarios, we introduce PMD-Bench, a multi-granularity benchmark suite that measures whether LoRA-based memory banks are addressable across coarse-grained domain knowledge, fine-grained document knowledge, and task-oriented capability access, instantiated by NQ-DomainLoRA, PaperQA, and Task-LoRA, respectively. Training data are converted into a unified two-turn `messages` format for LoRA training, while evaluation data are kept as independent QA or task instances for access selection and downstream scoring. Detailed dataset statistics and construction procedures are provided in Appendix A. Together, these benchmarks provide a compact but diverse suite for studying zero-shot access to external parametric memory across knowledge, document, and skill memory settings.

4 METHODOLOGY

4.1 FRAMEWORK OVERVIEW

Figure 2 summarizes our methodology. Under the zero-shot formulation in Section 3.1, PMD views backbone-and-LoRA-only routing as decoding query-conditioned responses rather than matching queries to static adapter descriptors. Each memory unit L_k induces a response object $\rho_k(q)$, which is mapped to an access score and judged by its separability from competing responses. PMDRouter instantiates PMD with a low-rank linear response and a calibrated energy decoder, requiring only one adapter-free backbone prefill and no trained gate, external retrieval, or per-LoRA trial generation. The same view also motivates an optional training-side signal-writing objective that strengthens the response signal used at inference time.

4.2 PMD FRAMEWORK

Section 3.1 defines the admissible information for zero-shot LoRA routing: frozen backbone activations and observable LoRA weights. PMD specifies how to turn this information into a routing signal. PMD is a framework for constructing zero-shot LoRA routing signals by decomposing routing into response construction and response decoding:

$$s_k(q) = D\big(\mathrm{Resp}\big(h(q), \theta_k\big)\big), \qquad k^*(q) = \arg\max_{k \in [K]} s_k(q), \tag{2}$$

where $h(q)$ is the query representation produced by the frozen backbone, $\theta_k = (A_k, B_k)$ denotes the observable parameters of the k-th LoRA memory unit, $\mathrm{Resp}(\cdot, \cdot)$ constructs the query-conditioned response induced by this LoRA, and D decodes this response into a scalar routing score.

Separable response object. This formulation makes the routing object explicit. The router does not compare the query with a static adapter descriptor; instead, for each candidate LoRA L_k, it computes an inference-time response signal:

$$\rho_k(q) = \mathrm{Resp}\big(h(q), \theta_k\big). \tag{3}$$

The response $\rho_k(q)$ can be instantiated as a hidden-state perturbation, response energy, loss-related surrogate, or any other signal computed only from frozen backbone activations and LoRA weights.

Thus, the routable object is the query-conditioned signal induced by the adapter, rather than the adapter identity or a static descriptor. Given the response-derived score $s_k(q) = D(\rho_k(q))$, routing succeeds when the target memory produces a score that is separated from competing memories. Let $k^\star$ denote the target memory for query q. We measure this separability by the PMD response margin $\gamma(q) = s_{k^\star}(q) - \max_{j \neq k^\star} s_j(q)$. A larger $\gamma(q)$ means that the target LoRA produces a more distinguishable response under the current query, making it easier to construct an effective decoder for routing.

Decoding pathway. Under this view, a concrete PMD router is determined by two design choices: the response constructor Resp, which decides what inference-time signal is read from each LoRA, and the decoder D, which maps this signal to an access score. The resulting decoding pathway is

$$q \to h(q) \to \{\rho_k(q)\}_{k=1}^K \to \{s_k(q)\}_{k=1}^K \to r(q), \tag{4}$$

where $r(q)$ is the final access pattern over the LoRA bank. In hard routing, $r(q)$ selects one memory unit; in soft routing, it gives a distribution over candidate units.

4.3 PMDROUTER

PMDRouter instantiates the PMD solver in Section 4.2 with a low-rank linear response operator and a scale-normalized energy decoder. This gives a parameter-free routing rule that requires only one adapter-free backbone prefill and the frozen LoRA weights. Concretely, it realizes the abstract response object $\rho_k(x)$ by applying each candidate LoRA update to a query-conditioned backbone activation, and then maps this response to an access score through scale-normalized response energy. Directly probing each candidate LoRA with an estimated loss provides a strong separability signal, but it requires per-LoRA evaluation. PMDRouter serves as an efficient approximation to this loss-based signal by replacing per-LoRA loss probing with response-energy decoding; Appendix D.3 implements and discusses several additional PMD routing paths under this framework.

Linear-response instantiation of Resp. Let f_{base} be a frozen transformer with L layers, and let $\mathcal{M} = \{L_k\}_{k=1}^K$ be a bank of LoRA memory units. For each adapted layer $\ell \in \mathcal{S}_k$, the k-th LoRA provides a low-rank update $\Delta W_{\ell,k} = B_{\ell,k} A_{\ell,k}$, where $A_{\ell,k} \in \mathbb{R}^{r_\ell \times d}$ and $B_{\ell,k} \in \mathbb{R}^{d \times r_\ell}$. Given an input x, PMDRouter performs a single adapter-free prefill and extracts a routing activation $u(x) = \mathrm{SemPool}(h_{\ell_\star}^{\mathrm{base}}(x)) \in \mathbb{R}^d$, where $\mathrm{SemPool}$ is a fixed pooling rule such as last-token pooling or span-level mean pooling.

Following the PMD definition in Eq. (3), we instantiate the response object as the LoRA-induced first-order change on this activation:

$$\rho_{\ell,k}(x) = \mathrm{Resp}_{\mathrm{lin}}\big(u(x), \theta_{\ell,k}\big) = \Delta W_{\ell,k} u(x) = B_{\ell,k} A_{\ell,k} u(x). \tag{5}$$

This response can be motivated from a local loss- or output-response view. For routing, however, we only need a score that preserves the relative ordering among candidate memories. Therefore, candidate-independent constants and higher-order terms in the local expansion are omitted, since they do not affect the ranking induced by the response score. This response is the basic object whose relative, scale-normalized strength will be decoded for routing.

Scale-normalized energy decoder. Given the responses $\{\rho_{\ell,k}(x)\}_{\ell \in \mathcal{S}_k}$, PMDRouter decodes each response by a scale-normalized addressable energy:

$$E_{\ell,k}(x) = \frac{\|\rho_{\ell,k}(x)\|_2^2}{\|u(x)\|_2^2 \, \|\Delta W_{\ell,k}\|_F^2 + \epsilon} = \frac{\|B_{\ell,k} A_{\ell,k} u(x)\|_2^2}{\|u(x)\|_2^2 \, \|B_{\ell,k} A_{\ell,k}\|_F^2 + \epsilon}. \tag{6}$$

This score measures the relative response strength of the k-th memory unit after removing the scale effects of the input activation and LoRA update norm.

We aggregate the normalized energies across layers by mean pooling and obtain the final score $\widetilde{E}_k(x) = \mathrm{Calib}_k(|\mathcal{S}_k|^{-1} \sum_{\ell \in \mathcal{S}_k} E_{\ell,k}(x))$. In our clean implementation, the calibration statistics are estimated from the training split and then fixed during evaluation; no test labels or trainable routing modules are used.

Decoding and routed inference. The calibrated addressable energy $\widetilde{E}_k(x)$ is used as the PMD access score for the k-th LoRA memory unit, where the response object is the LoRA-induced linear response $\rho_{\ell,k}(x) = B_{\ell,k}A_{\ell,k}u(x)$ and the decoder is the scale-normalized, calibrated response energy. In the hard-routing setting used in our main experiments, PMDRouter computes this score for all candidate LoRAs, selects the highest-scoring memory unit, and then runs the model with only the selected adapter activated. Thus, routing and generation are separated: the backbone is first used once without adapters to construct $u(x)$, and the selected LoRA is used only in the final inference pass.

Concretely, PMDRouter instantiates the PMD score in Eq. (2) as

$$s_k^{\mathrm{PMDRouter}}(x) = \widetilde{E}_k(x), \qquad \hat{k}(x) = \arg\max_{k\in[K]} s_k^{\mathrm{PMDRouter}}(x). \tag{7}$$

This shows that PMDRouter realizes PMD by using query-conditioned LoRA responses as the routable object and calibrated addressable energy as the decoder.

4.4 Training-Side Signal Writing

PMD also suggests a training-side mechanism for improving future memory access. Since routing reliability depends on whether the target response is separated from competing responses, a LoRA bank becomes easier to access when the target unit exposes a stronger and more distinguishable response. When adapter training is controllable, we therefore augment the original task objective with a signal-writing loss:

$$\mathcal{L} = \mathcal{L}_{\mathrm{task}} + \lambda\mathcal{L}_{\mathrm{write}}, \qquad \mathcal{L}_{\mathrm{write}} = \ell\big(g(\rho_k(x)), z_k\big). \tag{8}$$

Here, $\rho_k(x)$ denotes the query-conditioned response object in Eq. (3), z_k is the response code assigned to memory unit k, $g(\cdot)$ extracts a response feature, and $\ell(\cdot,\cdot)$ aligns it with the designated code. In PMDRouter, $\rho_k(x)$ can be instantiated by the low-rank linear responses in Eq. (5), and $g(\cdot)$ can be chosen to match the response-energy decoder in Eq. (6). This objective does not train an additional router or add inference-time parameters; it shapes LoRA parameters so that the same PMD decoder reads a stronger response signal, thereby increasing the separation between the target score $s_{k^\star}(q)$ and the strongest competing score $\max_{j\neq k^\star} s_j(q)$.

5 Experiments

5.1 Experimental Setup

Baselines. We compare PMDRouter against four baseline families defined by the source of routing signal, covering Backbone-only, LoRA-only, joint Backbone-LoRA, and non-parametric retrieval settings under the same zero-shot constraints. (1) Backbone-only signal baselines use only the frozen base model's hidden states or logits. (2) LoRA-only signal baselines use only statistics derived from each LoRA module itself. (3) Joint Backbone-LoRA signal baselines combine backbone-side query representations with LoRA-side module information, yielding stronger zero-shot baselines without training an additional router; this family covers representative zero-shot and training-free routing methods such as Arrow (Ostapenko et al., 2024), SpectR (Fleshman & Van Durme, 2025c), and LAG (Fleshman & Van Durme, 2025a). We also include (4) Text retrieval baselines with dense retrieval and BM25 to compare against non-zero-shot retrieval-based methods. The full names of these methods are provided in Appendix C.1.

Benchmark and evaluation sets. We evaluate all methods on the PMD-BENCH defined in Section 3.2: NQ-DomainLoRA, PaperQA, and Task-LoRA, which respectively represent domain-level knowledge memory, local document memory, and task-skill memory. For each benchmark, we train a bank of LoRA adapters on its training split (see Appendix B for training details) and evaluate zero-shot routing on the held-out split, where the router selects the most appropriate adapter for each input. We report routing accuracy (Acc). Detailed dataset construction and statistics are provided in Appendix A.

Method	PaperQA			NQ-DomainLoRA			TASK-Lora		
	Llama-3.1-8B	Qwen3-4B-Instruct	Qwen3-8B	Llama-3.1-8B	Qwen3-4B-Instruct	Qwen3-8B	Llama-3.1-8B	Qwen3-4B-Instruct	Qwen3-8B
Text retrieval									
embedding	0.762	0.762	0.762	0.397	0.397	0.397	0.884	0.884	0.884
bm25	0.820	0.820	0.820	0.897	0.897	0.897	0.735	0.735	0.735
Backbone–only Signals									
B-LogitVar	0.120	0.082	0.078	0.119	0.125	0.091	0.263	0.288	0.170
B-CtrL2	0.078	0.142	0.067	0.099	0.144	0.103	0.287	0.365	0.182
B-S4a	0.067	0.136	0.067	0.091	0.124	0.091	0.126	0.256	0.178
LoRA–only Signals									
L-AlphaZ	0.309	0.222	0.089	0.186	0.211	0.120	0.162	0.406	0.285
L-HidL2-L4Z	0.269	0.171	0.080	0.159	0.122	0.097	0.179	0.377	0.180
L-HidL2-LastMed	0.122	0.073	0.069	0.147	0.212	0.084	0.282	0.378	0.178
Joint Backbone–LoRA Signals									
J-Global	0.062	0.096	0.138	0.116	0.035	0.049	0.357	0.301	0.091
J-MLPGlobal	0.073	0.120	0.071	0.083	0.083	0.083	0.125	0.300	0.102
J-WAllZ	0.080	0.080	0.064	0.090	0.151	0.118	0.256	0.219	0.066
Arrow	0.364	0.342	0.311	0.087	0.130	0.162	0.158	0.139	0.108
SPECTR	0.187	0.100	0.127	0.601	0.570	0.474	0.358	0.302	0.017
LAG	0.440	0.598	0.636	0.733	0.793	0.754	0.315	0.326	0.217
PMDRouter(Ours)	0.600	0.613	0.658	0.769	0.781	0.790	0.663	0.897	0.923

Table 1: Main results on PMD-BENCH. We report top-1 routing accuracy (**Acc**) across three memory granularities and three backbones. Shaded cells indicate the best zero-shot router in each column, excluding text-retrieval baselines.

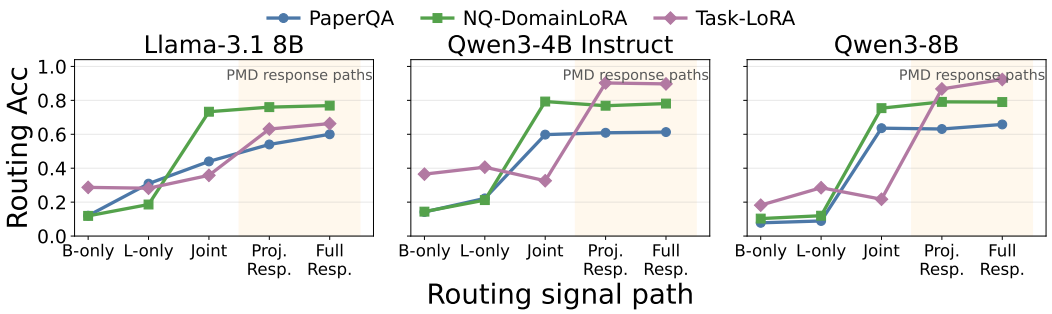

Figure 3: Routing accuracy across signal paths. Moving from single-side cues to joint and PMD response paths generally improves routing accuracy across backbones and benchmarks.

5.2 MAIN RESULT

PMDRouter leads the zero-shot baselines in most PMD-Bench settings (Table 1). It performs consistently on PaperQA and shows its largest gains on Task- LoRA. On NQ-DomainLoRA, it leads on Llama-3.1-8B and Qwen3- 8B, but trails LAG on Qwen3-4B-Instruct (0.781 vs. 0.793).

BM25 remains strong on PaperQA and NQ-DomainLoRA, while PMDRouter matches or exceeds retrieval baselines on Task-LoRA with the two Qwen backbones. Thus, backbone and LoRA signals support effective zero-shot routing, although retrieval remains competitive for knowledge-oriented memories. Judge scores, per-category results, and failure cases appear in Appendices D.4, D, and G.

5.3 RESULT ANALYSIS

We analyze PMDRouter from two complementary perspectives: whether response-based signal paths improve routing, and whether the PMD response margin explains routing behavior. As shown in Figure 3, routing accuracy generally improves as the routing signal moves from single-side cues to joint and response-based paths. Backbone-only and LoRA-only signals are consistently weak, while projection response and full response paths achieve stronger performance across backbones and benchmarks. This supports the PMD view that routing should be constructed from query-conditioned interactions between the frozen-backbone signal and LoRA weights, rather than from either side

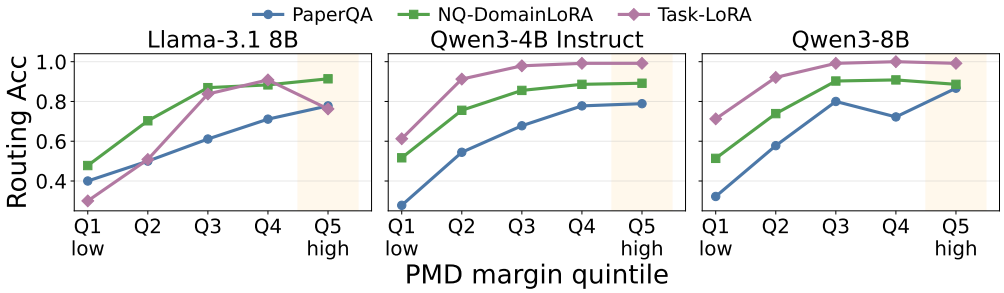

Figure 4: PMD response margin predicts routing success. Queries are grouped by normalized top-1/top-2 response margin; higher-margin groups generally achieve higher routing accuracy across backbones and benchmarks.

alone. We further examine the separability of the resulting PMD response signal. For each query, we compute the normalized top-1/top-2 PMD response margin and group examples into five quintiles. As shown in Figure 4, routing accuracy generally increases from low-margin to high-margin groups across backbones and benchmarks. This confirms that response separability is a key driver of memory addressability: when the target memory response is more clearly separated from competing responses, routing is more reliable. The margin trend is strongest on PaperQA and Task-LoRA, which helps explain the large gains of PMDRouter in Table 1. NQ-DomainLoRA also shows a clear increasing trend, although its high-margin groups tend to saturate, suggesting that domain-level knowledge routing is additionally affected by backbone-specific geometry and overlap among memory units.

5.4 Improving Memory Addressability During Training

Variant	Acc	NormGap
QMean-NoAux	0.629	0.466
QMean-Raw	0.689	0.554
QMean-Log	0.676	0.527
Last-Raw	0.631	0.479
Ortho	0.629	0.463
Mono	0.627	0.465
Base	0.613	0.463

Table 2: PaperQA routing after LoRA signal-writing training (Qwen3-4B-Instruct). Acc and normalized top-1/top-2 gap (NormGap) use the same inference-time router. **Color:** best.

On PaperQA with Qwen3-4B-Instruct, we compare training variants under the same QMean-All-LogC router, which uses question-mean pooling, all-module responses, and log-centered scores (Section 4.4; objectives in Appendix D.2). QMean-Raw achieves the highest accuracy and normalized gap (0.689/0.554), ahead of QMean-Log (0.676/0.527) and the matched QMean-NoAux control (0.629/0.466; Table 2). The unshaped Base bank scores 0.613/0.463, but its effective training batch differs from the QMean variants, so this comparison does not isolate the auxiliary loss.

BM25 remains strong on PaperQA and NQ-DomainLoRA, while PMDRouter matches or exceeds retrieval baselines on Task-LoRA with the two Qwen backbones. Thus, backbone and LoRA signals support effective zero-shot routing, although retrieval remains competitive for knowledge-oriented memories. Judge scores, per-category results, and failure cases appear in Appendices D.4, D, and G.

5.5 Problem Complexity Analysis

Beyond final routing accuracy, we analyze whether PMD also simplifies the structure of the zero-shot LoRA routing problem. Here, complexity refers not to runtime cost, but to the size and coupling of the design space required to construct a routing rule. As summarized in Table 3, the direct matching view requires choices over query representations, memory descriptors, and decision rules, leading to a large and highly coupled search space. In contrast, PMD fixes the routing object as the query-conditioned response and leaves the solver to choose how this response is decoded. Under the direct matching view, the router must jointly specify query-side signals, memory-side signals, and matching decisions, with additional coupling across these axes. This yields an estimated search space of ~ 56–4.2×10^5,

Formulation	Design space			Complexity			Best Acc.
	Qry.	Mem.	Dec.	Cpl.	S.S.	Comp.	
Direct matching view	$\sim$9	$\sim$11	$\sim$56	$\sim$21	~ 56–4.2×10^5	$\sim$1.00	0.636
PMD view	$\sim$1	$\sim$1	$\sim$3	$\sim$1	$\sim$3	$\sim$0.025	0.669

Table 3: Design-space complexity of direct matching and PMD. Abbreviations: Qry. denotes query representation, Mem. memory descriptor, Dec. decision rule, Cpl. coupling degree, S.S. search space, and Comp. normalized complexity. Lower values indicate simpler routing search; Best Acc. reports the top accuracy.

coupling degree of ~ 21, and normalized complexity of ~ 1.00. Under the PMD view, the query and memory axes collapse into a response-centered formulation: the main remaining choices are reduced to the response instantiation and the decoder. This reduces the estimated search space to ~ 3, coupling degree to ~ 1, and normalized complexity to ~ 0.025. Despite this smaller design space, PMD also supports stronger routing performance on PaperQA, improving the best accuracy from 0.636 under the direct matching view to 0.669. This suggests that PMD is useful not only as a stronger routing formulation, but also as a way to rewrite zero-shot LoRA routing into a more compact and less coupled decoding problem.

5.6 Ablation Study

To clarify PMDRouter's gains, we compare four inference-time decoder settings on PaperQA: *Backbone-only*, *LoRA-only*, *Projection-only*, and the full decoder. As shown in Figure 5, *Backbone-only* and *LoRA-only* are consistently weaker across all three backbones, showing that neither query-side nor adapter-side signals alone are sufficient for reliable LoRA selection. By contrast, *Projection-only* improves substantially on all three backbones and is the strongest variant across all backbones, suggesting that query-conditioned low-rank interaction already captures most of the useful routing signal. The full decoder is slightly worse than *Projection-only* across all three backbones, indicating that full write-back does not necessarily yield stronger routing separability. A plausible explanation is that $As(x)$ is closer to memory addressing, whereas $BAs(x)$ introduces the adapter-specific metric defined by B, which may bring generation-oriented scale biases. Overall, this ablation shows that zero-shot LoRA routing relies on decodable query-conditioned interactions, with the low-rank addressing projection providing the most stable signal.

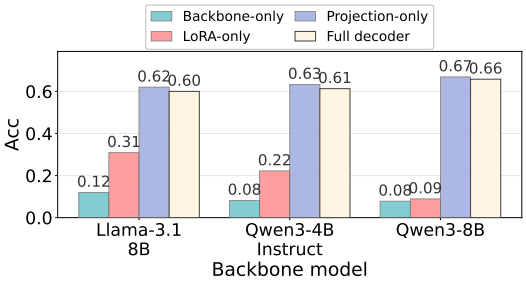

Figure 5: Ablation Study. Accuracy of different decoder adaptation settings across backbone models: Backbone only, LoRA only, Projection only, and Full decoder.

6 Conclusion

We study zero-shot LoRA routing for LoRA-based external parametric memory without an additional routing component. We organize PMD-Bench to evaluate this setting across document-level, domain-level, and task-skill memory, and propose PMD, which reframes routing as decoding activations over external parametric memory. Based on PMD, PMDRouter scores each LoRA by its response magnitude from a single adapter-free backbone prefill. Experiments show that it achieves the strongest internal-signal performance across multiple zero-shot settings, while retrieval remains competitive in some knowledge-oriented cases. Overall, these results demonstrate the feasibility of zero-shot LoRA routing and suggest that PMD can guide future improvements in zero-shot routing methods.

AI USE STATEMENT

Generative AI tools were used in this work to assist with manuscript preparation, language polishing, and aspects of experimental design, including refining the presentation of research ideas, checking the codebase, and improving the clarity and consistency of the manuscript. These tools served only a supporting role and did not independently determine the research questions, core methodology, experimental conclusions, or scientific claims of this work. All AI-assisted content and suggestions were carefully reviewed, verified, and revised by the authors.

REFERENCES

Vladimir Araujo, Marie Francine Moens, and Tinne Tuytelaars. Learning to route for dynamic adapter composition in continual learning with language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 687–696, 2024.

Seungju Back, Dongwoo Lee, Naun Kang, Taehee Lee, S. K. Hong, Youngjune Gwon, and Sungjin Ahn. Understanding lora as knowledge memory: An empirical analysis, 2026. URL https://arxiv.org/abs/2603.01097.

Maged Badawi, Mohammedyahia Abushanab, Sheethal Bhat, and Andreas Maier. Review of zero-shot and few-shot ai algorithms in the medical domain. *arXiv preprint arXiv:2406.16143*, 2024.

Haoyue Bai, Haoyu Wang, Shengyu Chen, Zhengzhang Chen, Lu-An Tang, Wei Cheng, Yanjie Fu, and Haifeng Chen. Learning to route: A rule-driven agent framework for hybrid-source retrieval-augmented generation. In *Proceedings of the ACM Web Conference 2026*, pp. 4338–4349, 2026.

Massimo Bini, Ondrej Bohdal, Umberto Michieli, Zeynep Akata, Mete Ozay, and Taha Ceritli. Memlora: Distilling expert adapters for on-device memory systems. *arXiv preprint arXiv:2512.04763*, 2025.

Eric L Buehler and Markus J Buehler. X-lora: Mixture of low-rank adapter experts, a flexible framework for large language models with applications in protein mechanics and molecular design. *APL Machine Learning*, 2(2), 2024.

Shaofeng Cai, Yao Shu, and Wei Wang. Dynamic routing networks. In *proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 3588–3597, 2021.

Jiaqi Cao, Jiarui Wang, Rubin Wei, Qipeng Guo, Kai Chen, Bowen Zhou, and Zhouhan Lin. Memory decoder: A pretrained, plug-and-play memory for large language models. *arXiv preprint arXiv:2508.09874*, 2025.

Jie Cao, Zhenxuan Fan, Zhuonan Wang, Tianwei Lin, Ziyuan Zhao, Rolan Yan, Wenqiao Zhang, Feifei Shao, Hongwei Wang, Jun Xiao, et al. Comol: Efficient mixture of lora experts via dynamic core space merging. *arXiv preprint arXiv:2603.00573*, 2026a.

Jinghan Cao, Yu Ma, Xinjin Li, Qingyang Ren, and Xiangyun Chen. Task-specific efficiency analysis: When small language models outperform large language models, 2026b. URL https://arxiv.org/abs/2603.21389.

Mei-Yen Chen, Thi Thu Uyen Hoang, Michael Hahn, and M. Saquib Sarfraz. When does LoRA reuse work? theoretical limits and mechanisms for recycling LoRAs without data access. *Transactions on Machine Learning Research*, 2026.

Shaoxiang Chen, Zequn Jie, and Lin Ma. Llava-mole: Sparse mixture of lora experts for mitigating data conflicts in instruction finetuning mllms. *arXiv preprint arXiv:2401.16160*, 2024.

Mingxuan Cui, Jingpu Yang, Fengxian Ji, Qian Jiang, Zhecheng Shi, Jiaming Wang, Zirui Song, Fajri Koto, and Xiuying Chen. Textalign: Preference alignment for text rendering with hierarchical rewards. *arXiv preprint arXiv:2605.19320*, 2026.

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.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

William Fleshman and Benjamin Van Durme. Lora-augmented generation (lag) for knowledge-intensive language tasks. *arXiv preprint arXiv:2507.05346*, 2025a.

William Fleshman and Benjamin Van Durme. Seqr: Secure and efficient qr-based lora routing. *arXiv preprint arXiv:2509.18093*, 2025b.

William Fleshman and Benjamin Van Durme. Spectr: Dynamically composing lm experts with spectral routing. *arXiv preprint arXiv:2504.03454*, 2025c.

Chongyang Gao, Kezhen Chen, Jinmeng Rao, Ruibo Liu, Baochen Sun, Yawen Zhang, Daiyi Peng, Xiaoyuan Guo, and VS Subrahmanian. Mola: Moe lora with layer-wise expert allocation. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 5097–5112, 2025.

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.

Qiang He, Yu Wang, Xingwei Wang, Weiqiang Xu, Fuliang Li, Kaiqi Yang, and Lianbo Ma. Routing optimization with deep reinforcement learning in knowledge defined networking. *IEEE Transactions on Mobile Computing*, 23(2):1444–1455, 2023.

Benjamin Heinzerling and Kentaro Inui. Language models as knowledge bases: On entity representations, storage capacity, and paraphrased queries. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 1772–1791, 2021.

Yitong Huang, Ziqi Yang, Zihui Wang, Jianzhong Qi, Rongshan Yu, Xiaoliang Fan, and Cheng Wang. Hybrid routing for a mixture of lora experts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 31211–31219, 2026.

Wooseong Jeong, Wonyoung Lee, and Kuk-Jin Yoon. Preference-aligned lora merging: Preserving subspace coverage and addressing directional anisotropy. *arXiv preprint arXiv:2603.26299*, 2026.

Fengxian Ji, Jingpu Yang, Zirui Song, Lang Gao, Junhong Liang, Zhenhao Chen, Jinghui Zhang, and Xiuying Chen. Servimage: An image generation and editing benchmark from real-world commercial imaging services. *ACL26 Main*, 2026a.

Fengxian Ji, Jingpu Yang, Zirui Song, Yuanxi Wang, Zhexuan Cui, Yuke Li, Qian Jiang, and Xiuying Chen. Finestate-bench: Benchmarking state-conditioned grounding for fine-grained gui state setting. *ACL26 Findings*, 2026b.

Pengfei Jin, Peng Shu, Sifan Song, Sekeun Kim, Qing Xiao, Cheng Chen, Tianming Liu, Xiang Li, and Quanzheng Li. Beyond adapter retrieval: Latent geometry-preserving composition via sparse task projection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 22417–22425, 2026.

Adarsh Kappiyath, Abhra Chaudhuri, Jiaxi Li, AJAY KUMAR JAISWAL, Xilu Wang, Li Shen, Souvik Kundu, Xiatian Zhu, and Lu Yin. Maple: Masked adapter prototype learning for ood generalization.

Rui Kong, Qiyang Li, Xinyu Fang, Qingtian Feng, Qingfeng He, Yazhu Dong, Weijun Wang, Yuanchun Li, Linghe Kong, and Yunxin Liu. Lora-switch: Boosting the efficiency of dynamic llm adapters via system-algorithm co-design. *arXiv preprint arXiv:2405.17741*, 2024.

Lara Magdalena Lazier, Aritra Dhar, Vasilije Stambolic, and Lukas Cavigelli. Ac-lora:(almost) training-free access control-aware multi-modal llms. *arXiv preprint arXiv:2505.11557*, 2025.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim ocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474, 2020.

Dengchun Li, Yingzi Ma, Naizheng Wang, Zhengmao Ye, Zhiyuan Cheng, Yinghao Tang, Yan Zhang, Lei Duan, Jie Zuo, Cal Yang, et al. Mixlora: Enhancing large language models fine-tuning with lora-based mixture of experts. *arXiv preprint arXiv:2404.15159*, 2024a.

Dengchun Li, Naizheng Wang, Zihao Zhang, Haoyang Yin, Lei Duan, Meng Xiao, and Mingjie Tang. Dynmole: Boosting mixture of lora experts fine-tuning with a hybrid routing mechanism. *arXiv preprint arXiv:2504.00661*, 2025a.

Guanyu Li, Zhiheng Xi, Zhihao Zhang, Boyang Hong, Tao Gui, Qi Zhang, and Xuan-Jing Huang. Loracoe: Improving large language model via composition-based lora expert. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 31278–31292, 2025b.

Qiyang Li, Rui Kong, Yuchen Li, Hengyi Cai, Shuaiqiang Wang, Linghe Kong, Guihai Chen, and Dawei Yin. Adafuse: Accelerating dynamic adapter inference via token-level pre-gating and fused kernel optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 31680–31688, 2026a.

Xinjin Li, Yu Ma, Yangchen Huang, Xingqi Wang, Yuzhen Lin, and Chenxi Zhang. Synergized data efficiency and compression (sec) optimization for large language models. In *2024 4th International Conference on Electronic Information Engineering and Computer Science (EIECS)*, pp. 586–591, 2024b. doi: 10.1109/EIECS63941.2024.10800533.

Xinjin Li, Yu Ma, Kaisen Ye, Jinghan Cao, Minghao Zhou, and Yeyang Zhou. Hy-Facial: hybrid feature extraction by dimensionality reduction methods for enhanced facial expression classification. In Wolfgang Osten and Eden Mamut (eds.), *Eighteenth International Conference on Machine Vision (ICMV 2025)*, volume 14114, pp. 141140R. International Society for Optics and Photonics, SPIE, 2026b. doi: 10.1117/12.3096291. URL https://doi.org/10.1117/12.3096291.

Zhiwen Li, Zhongjie Duan, Die Chen, Cen Chen, Daoyuan Chen, Yaliang Li, and Yingda Chen. Autolora: Automatic lora retrieval and fine-grained gated fusion for text-to-image generation. *arXiv preprint arXiv:2508.02107*, 2025c.

Haodong Lu, Chongyang Zhao, Jason Xue, Lina Yao, Kristen Moore, and Dong Gong. Little by little: Continual learning via incremental mixture of rank-1 associative memory experts. *arXiv preprint arXiv:2506.21035*, 2025.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 9802–9822, 2023.

Mohammed Muqeeth, Haokun Liu, Yufan Liu, and Colin Raffel. Learning to route among specialized experts for zero-shot generalization. *arXiv preprint arXiv:2402.05859*, 2024.

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.

Lucas Page-Caccia, Edoardo Maria Ponti, Zhan Su, Matheus Pereira, Nicolas Le Roux, and Alessandro Sordoni. Multi-head adapter routing for cross-task generalization. *Advances in Neural Information Processing Systems*, 36:56916–56931, 2023.

Edoardo Maria Ponti, Alessandro Sordoni, Yoshua Bengio, and Siva Reddy. Combining parameter-efficient modules for task-level generalisation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 687–702, 2023.

Akshara Prabhakar, Yuanzhi Li, Karthik Narasimhan, Sham Kakade, Eran Malach, and Samy Jelassi. Lora soups: Merging loras for practical skill composition tasks. In *Proceedings of the 31st International Conference on Computational Linguistics: Industry Track*, pp. 644–655, 2025.

Jingyang Qiao, Weicheng Meng, Yu Cheng, Zhihang Lin, Zhizhong Zhang, Xin Tan, Jingyu Gong, Kun Shao, and Yuan Xie. Memory intelligence agent. *arXiv preprint arXiv:2604.04503*, 2026.

Kaku Sawada, Daisuke Kotani, and Yasuo Okabe. Network routing optimization based on machine learning using graph networks robust against topology change. In *2020 International conference on information networking (ICOIN)*, pp. 608–615. IEEE, 2020.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Shreya Shukla, Aditya Sriram, Milinda Kuppur Narayanaswamy, and Hiteshi Jain. qa-flora: Data-free query-adaptive fusion of loras for llms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 32992–33000, 2026.

Pablo Sprechmann, Siddhant M Jayakumar, Jack W Rae, Alexander Pritzel, Adria Puigdomenech Badia, Benigno Uria, Oriol Vinyals, Demis Hassabis, Razvan Pascanu, and Charles Blundell. Memory-based parameter adaptation. *arXiv preprint arXiv:1802.10542*, 2018.

Zhan Su, Fengran Mo, Guojun Liang, Jinghan Zhang, Bingbing Wen, Prayag Tiwari, and Jian-Yun Nie. Tensorized clustered lora merging for multi-task interference. *arXiv preprint arXiv:2508.03999*, 2025.

Yimin Tian, Bolin Zhang, Zhiying Tu, and Dianhui Chu. Adapters selector: Cross-domains and multi-tasks lora modules integration usage method. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 593–605, 2025.

Asaf Valadarsky, Michael Schapira, Dafna Shahaf, and Aviv Tamar. Learning to route. In *Proceedings of the 16th ACM workshop on hot topics in networks*, pp. 185–191, 2017.

Hitesh Wadhwa, Rahul Seetharaman, Somyaa Aggarwal, Reshmi Ghosh, Samyadeep Basu, Soundararajan Srinivasan, Wenlong Zhao, Shreyas Chaudhari, and Ehsan Aghazadeh. From rags to rich parameters: Probing how language models utilize external knowledge over parametric information for factual queries. *arXiv preprint arXiv:2406.12824*, 2024.

Hanqing Wang, Yixia Li, Shuo Wang, Guanhua Chen, and Yun Chen. Milora: Harnessing minor singular components for parameter-efficient llm finetuning. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4823–4836, 2025.

Peng Wang, Zexi Li, Ningyu Zhang, Ziwen Xu, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. Wise: Rethinking the knowledge memory for lifelong model editing of large language models. *Advances in Neural Information Processing Systems*, 37:53764–53797, 2024.

Thomas Wang, Adam Roberts, Daniel Hesslow, Teven Le Scao, Hyung Won Chung, Iz Beltagy, Julien Launay, and Colin Raffel. What language model architecture and pretraining objective works best for zero-shot generalization? In *International Conference on Machine Learning*, pp. 22964–22984. PMLR, 2022.

Rubin Wei, Jiaqi Cao, Jiarui Wang, Jushi Kai, Qipeng Guo, Bowen Zhou, and Zhouhan Lin. Mlp memory: A retriever-pretrained memory for large language models. *arXiv preprint arXiv:2508.01832*, 2025.

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

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.

Jie-Si Yang, Zhuoqi Zeng, and Zijian Shen. Neural-symbolic dual-indexing architectures for scalable retrieval-augmented generation. *IEEE Access*, 13:210507–210519, 2025.

Jingpu Yang, Fengxian Ji, Zhengzhao Lai, Zhexuan Cui, Guangxian Ouyang, Qian Jiang, Fan Zhang, Min Peng, Qianqian Xie, Preslav Nakov, et al. Labguard: Grounding natural-language laboratory rules into runtime guards for embodied laboratory agents. *arXiv preprint arXiv:2606.31045*, 2026a.

Jingpu Yang, Fengxian Ji, Zhengzhao Lai, Juanfan Wu, Mingxuan Cui, and Yufeng Wang. Zero-parameter geometric gating for temporally stable low-altitude uav video semantic segmentation. *arXiv preprint arXiv:2606.09162*, 2026b.

Fan Zhang, Zhen Li, Sijia Peng, and Yu Chen. When alpha disappears: A one-switch benchmark for decision-time leakage in financial backtests. *arXiv preprint arXiv:2605.23959*, 2026a.

Fan Zhang, Jiabin Luo, Zheng Zhang, Shuanghong Huang, Zhipeng Liu, and Yu Chen. Beyond visual realism: Toward reliable financial time series generation. *arXiv preprint arXiv:2601.12990*, 2026b.

Fan Zhang, Mingzi Song, Rania Elbadry, Yankai Chen, Shaobo Wang, Yixi Zhou, Xunwen Zheng, Yueru He, Yuyang Dai, Georgi Georgiev, et al. Finreporting: An agentic workflow for localized reporting of cross-jurisdiction financial disclosures. *arXiv preprint arXiv:2604.05966*, 2026c.

Hao Zhang, Yuyang Zhang, Xiaoguang Li, Wenxuan Shi, Haonan Xu, Huanshuo Liu, Yasheng Wang, Lifeng Shang, Qun Liu, Yong Liu, et al. Evaluating the external and parametric knowledge fusion of large language models. *arXiv preprint arXiv:2405.19010*, 2024.

Haobo Zhang and Jiayu Zhou. Unraveling lora interference: Orthogonal subspaces for robust model merging. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 26459–26472, 2025.

Xin Zhang, Kailai Yang, Chenyue Li, Hao Li, Qiyu Wei, Jun'ichi Tsujii, and Sophia Ananiadou. Memadapter: Fast alignment across agent memory paradigms via generative subgraph retrieval. *arXiv preprint arXiv:2602.08369*, 2026d.

Yuxuan Zhang and Ruizhe Li. Dlp-lora: Efficient task-specific lora fusion with a dynamic, lightweight plugin for large language models. *arXiv preprint arXiv:2410.01497*, 2024.

Zeyu Zhang, Quanyu Dai, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. A survey on the memory mechanism of large language model-based agents. *ACM Transactions on Information Systems*, 43(6):1–47, 2025.

Ziyu Zhao, Leilei Gan, Guoyin Wang, Yuwei Hu, Tao Shen, Hongxia Yang, Kun Kuang, and Fei Wu. Retrieval-augmented mixture of lora experts for uploadable machine learning. *arXiv preprint arXiv:2406.16989*, 2024a.

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*, pp. 4447–4462, 2024b.

Yuhang Zhou, Zihua Zhao, Haolin Li, Siyuan Du, Jiangchao Yao, Ya Zhang, and Yanfeng Wang. Exploring training on heterogeneous data with mixture of low-rank adapters. *arXiv preprint arXiv:2406.09679*, 2024.

Yuan Zhuang, Yi Shen, Yuexin Bian, Qing Su, Shihao Ji, Yuanyuan Shi, and Fei Miao. Ld-mole: Learnable dynamic routing for mixture of lora experts. *arXiv preprint arXiv:2509.25684*, 2025.

A DATASET

We construct three benchmark datasets: NQ-DomainLoRA, PaperQA, and Task-LoRA. All training data are formatted into a unified two-turn `messages` schema for LoRA training, while evaluation data are kept as question–answer items for independent scoring. Table 4 summarizes the overall scale of the three datasets and presents representative evaluation examples.

Dataset	Unit	Train	Eval/Test	Task type	Example input	Example answer
NQ-DomainLoRA	360 domain documents	7,200 messages	1,800 QA	Domain QA	How does the narrative focus shift between the animated trilogy and *Dragons: The Nine Realms*?	The trilogy focuses on Hiccup's coming of age in a mythical Viking setting, whereas *The Nine Realms* shifts the story about 1,300 years into the future.
PaperQA	15 paper introductions	795 messages	450 QA	Paper QA	What framework integrates diffusion-based trajectory generation with MCTS?	The proposed framework is Monte Carlo Tree Diffusion (MCTD).
Task-LoRA	8 task families	4,800 messages	1,200 messages	NLI	Determine the relationship between the premise and hypothesis. Return one label: entailment or not_entailment.	entailment

Table 4: Dataset statistics and representative evaluation/test examples for the three benchmark datasets.

NQ-DomainLoRA . NQ-DomainLoRA is designed to study *knowledge-oriented* LoRA routing. We start from the Natural Questions training set and group examples by Wikipedia document using title- and URL-based domain rules. We keep 12 target domains: *entertainment*, *history*, *sports*, *geography*, *music*, *politics*, *literature*, *business*, *science*, *biology*, *technology*, and *food*. For each domain, we sample 30 documents, yielding 360 documents in total. Each document is paired with a Wikipedia plain-text extract truncated to at most 6,000 characters. For training, each document yields 20 messages: 1 read/memorize example, 15 factual QA examples, 2 summaries, and 2 rewrites. For evaluation, each document yields 5 QA items emphasizing cross-sentence synthesis and relation-level reasoning. The resulting dataset contains 7,200 training messages and 1,800 evaluation QA pairs.

PaperQA . PaperQA is designed to study *local scientific knowledge memory*. We collect 15 papers from ICLR 2025, ICML 2025, and NeurIPS 2024, and extract their introduction sections from PDF or fallback sources. Each introduction is used to construct both training and evaluation data. The evaluation set contains 30 QA items per paper, organized into three difficulty levels: factual recall, contextual understanding, and logical/motivational inference. The training set contains 53 messages per paper: 1 reading example, 40 QA examples, 8 summaries, and 4 rewrites. In total, PaperQA contains 795 training messages and 450 evaluation QA pairs. The average introduction length is 5,131.67 characters, with a range from 2,165 to 14,298 characters.

Bank	Units	r	α	Dropout
PaperQA	15	16	16	0
NQ-DomainLoRA	12	16	32	0.02
Task-LoRA	8	16	32	0.05

Table 5: LoRA configuration per backbone. PaperQA adapts the attention projections `q_proj`, `k_proj`, `v_proj`, and `o_proj` and the MLP projections `gate_proj`, `up_proj`, and `down_proj`. NQ-DomainLoRA and Task-LoRA adapt only the four attention projections.

Task-LoRA . Task-LoRA is designed to study *task- and skill-oriented* LoRA routing. It covers 8 task families: BoolQ, edit/summarization, extractive QA, generative chain-of-thought, instruction following, multiple-choice QA, math chain-of-thought, and NLI. Raw data are collected from public task sources and converted into a unified `messages` format. For each task family, we shuffle with a fixed seed and take 600 samples for training and 150 for testing. The resulting dataset contains 4,800 training messages and 1,200 test messages in total.

For NQ-DomainLoRA and PaperQA, training files are stored as JSON objects whose keys are document or paper IDs and whose values are message lists; evaluation files are JSON objects mapping IDs to QA lists. For Task-LoRA, each task is stored as JSONL with fields including `id`, `task`, `messages`, and `meta`. This unified format allows all three datasets to be directly used for instruction tuning or LoRA training, while preserving evaluation annotations for separate scoring. As a reference compute estimate, training the nine LoRA-bank configurations used in our main experiments takes approximately 80 wall-clock hours on two NVIDIA RTX 6000 GPUs, and routing plus inference evaluation takes approximately 30 wall-clock hours on the same hardware.

B LoRA Training Details

We train one adapter per paper, domain, or task family, independently for each of the three backbones (Llama-3.1-8B, Qwen3-4B-Instruct, and Qwen3-8B). Each run starts from the corresponding frozen backbone and uses only the training split of its memory unit; the evaluation and test instances are reserved for routing and downstream evaluation. The resulting banks contain 15 PaperQA, 12 NQ-DomainLoRA, and 8 Task-LoRA adapters per backbone. Table 5 summarizes the adapter configurations recorded in the final checkpoints. No backbone weights or bias terms are trained.

Optimization. For the standard PaperQA and NQ-DomainLoRA banks, we minimize the assistant-token causal language modeling loss defined in Appendix D.2. Both use BF16 training, a maximum sequence length of 2,048 tokens, gradient checkpointing, AdamW with weight decay 0.01, a cosine learning-rate schedule, and training seed 42. Each adapter is trained on a single GPU; the batch sizes below are per-device sizes. These values are the effective launch settings, which override the defaults in the shared configuration files.

For PaperQA, each adapter has 53 training messages and is trained for 200 optimizer steps at learning rate 2×10^{-4}, with 20 warm-up steps. Qwen3-4B-Instruct uses batch size 8 and one gradient-accumulation step; Llama-3.1-8B and Qwen3-8B use batch size 1 and eight accumulation steps. Thus all three runs have an effective batch size of 8 examples.

For NQ-DomainLoRA, each adapter has 600 training messages and runs for eight epochs at learning rate 10^{-4}. The per-device batch sizes are 6, 7, and 5 for Llama-3.1-8B, Qwen3-4B-Instruct, and Qwen3-8B, respectively, with no gradient accumulation. These settings yield 800, 688, and 960 optimizer steps, respectively; linear warm-up occupies approximately the first 10% of each run (80, 68, and 96 steps), followed by cosine decay.

C Derivations and Analysis

C.1 Baseline Methods

This section summarizes the baseline routing methods used in our experiments. For each method, we first describe its full name and signal source in text, and then give only the final routing signal.

`embedding`: TF-IDF Cosine Similarity Retrieval. Despite the name, this baseline uses sparse TF-IDF vectors rather than dense embeddings. It ranks adapters by the cosine similarity between the query text and the adapter- or paper-side text d_a:

$$s_a(x) = \cos\big(\mathrm{tfidf}(x), \mathrm{tfidf}(d_a)\big).$$

`bm25`: BM25 Lexical Retrieval. This baseline ranks adapters by lexical term matching between the query text and the adapter- or paper-side text d_a. In our implementation, $k_1 = 1.5$ and $b = 0.75$:

$$s_a(x) = \sum_{t \in x} \mathrm{IDF}(t) \frac{f(t, d_a)(k_1 + 1)}{f(t, d_a) + k_1\big(1 - b + b|d_a|/\mathrm{avgdl}\big)}.$$

`B-LogitVar`: Backbone-only Logit Delta Variance. This method uses the top-token logit perturbation obtained by projecting the LoRA-induced hidden perturbation through the LM head. Let $\delta h_a = \sum_m \Delta W_{a,L,m} h^0_{L,m}(x)$ and $\delta z_{a,t} = \langle W_t, \delta h_a \rangle$. The final signal is the weighted variance of the logit perturbation over the selected token set T:

$$s_a(x) = \sum_{t \in T} \bar{p}_t \left(\delta z_{a,t} - \sum_{u \in T} \bar{p}_u \delta z_{a,u} \right)^2.$$

`B-CtrL2`: Backbone-only Centered Logit L2. This method uses the same logit perturbation as `B-LogitVar`, but scores each adapter by the distance of its perturbation from the adapter-wise center:

$$s_a(x) = \left\| \delta z_a - \frac{1}{|\mathcal{A}|} \sum_{b \in \mathcal{A}} \delta z_b \right\|_2.$$

`B-S4a`: Backbone-only Positive Surprisal Alignment. This method aligns the positive part of the centered logit perturbation with the base-model surprisal:

$$s_a(x) = \sum_{t \in T} \left[\delta z_{a,t} - \frac{1}{|\mathcal{A}|} \sum_{b \in \mathcal{A}} \delta z_{b,t} \right]_+ (-\log p_t).$$

`L-AlphaZ`: LoRA-only Alpha Norm with per-LoRA Z-score. This method uses the low-rank coordinate response of the LoRA A matrices to the backbone hidden states. Let μ_a and σ_a denote the mean and standard deviation computed over all queries for the same LoRA a:

$$r_a(x) = \left(\sum_m \left\| A_{a,L,m} h^0_{L,m}(x) \right\|_2^2 \right)^{1/2}, \qquad s_a(x) = \frac{r_a(x) - \mu_a}{\sigma_a}.$$

`L-HidL2-L4Z`: LoRA-only Hidden Centered L2 over Last-4 Layers with per-LoRA Z-score. This method uses centered hidden perturbations over the last four layers and takes the maximum layer score. Let $\delta h_{a,\ell} = \sum_m \Delta W_{a,\ell,m} h^0_{\ell,m}(x)$, and let μ_a, σ_a denote per-LoRA normalization statistics:

$$r_a(x) = \max_{\ell \in \mathrm{last4}} \left\| \delta h_{a,\ell} - \frac{1}{|\mathcal{A}|} \sum_{b \in \mathcal{A}} \delta h_{b,\ell} \right\|_2, \qquad s_a(x) = \frac{r_a(x) - \mu_a}{\sigma_a}.$$

`L-HidL2-LastMed`: LoRA-only Last-layer Hidden Centered L2 with Median Center. This method uses the last-layer LoRA hidden perturbation centered by the adapter-wise median:

$$s_a(x) = \left\| \delta h_{a,L} - \mathrm{median}_{b \in \mathcal{A}} \big(\delta h_{b,L} \big) \right\|_2.$$

`J-Global`: Joint Backbone-LoRA Global Geometry Score. This method measures the global cosine alignment between the LoRA perturbation and the base-model negative gradient. By default, it uses the last four layers, all prompt tokens, and the modules q_proj, v_proj, o_proj, up_proj, gate_proj, and down_proj:

$$s_a(x) = \frac{\sum_{\ell,m,t} \left\langle \Delta W_{a,\ell,m} h^a_{\ell,m,t}(x), g_{\ell,m,t}(x,\hat{y}) \right\rangle}{\sqrt{\sum_{\ell,m,t} \left\| \Delta W_{a,\ell,m} h^a_{\ell,m,t}(x) \right\|_2^2} \sqrt{\sum_{\ell,m,t} \left\| g_{\ell,m,t}(x,\hat{y}) \right\|_2^2}}.$$

`J-MLPGlobal`: Joint Backbone-LoRA MLP Global Geometry over All Prompt Tokens. This method is the MLP-only version of `J-Global`. It uses the same global cosine score, but restricts the module set to $m \in \{\text{up_proj}, \text{gate_proj}, \text{down_proj}\}$:

$$s_a(x) = \frac{\sum_{\ell,m,t} \left\langle \Delta W_{a,\ell,m} h^a_{\ell,m,t}(x), g_{\ell,m,t}(x,\hat{y}) \right\rangle}{\sqrt{\sum_{\ell,m,t} \left\| \Delta W_{a,\ell,m} h^a_{\ell,m,t}(x) \right\|_2^2} \sqrt{\sum_{\ell,m,t} \left\| g_{\ell,m,t}(x,\hat{y}) \right\|_2^2}}.$$

`J-WAllZ`: Joint Weighted All-layer Geometry with Query-wise Z-score. This method uses the weighted all-layer dot product between the q_proj LoRA perturbation and the negative gradient, followed by query-wise z-score normalization across adapters. Let μ_x and σ_x denote query-wise statistics over candidate adapters:

$$r_a(x) = \sum_{\ell,t} w_\ell \left\langle \Delta W_{a,\ell,q} h^a_{\ell,q,t}(x), g_{\ell,q,t}(x,\hat{y}) \right\rangle, \qquad s_a(x) = \frac{r_a(x) - \mu_x}{\sigma_x}.$$

`Arrow`: Zero-shot PEFT Routing by LoRA Parameter Geometry. Arrow scores each adapter by matching the dominant right singular direction of each LoRA update with the query hidden state:

$$v_{a,\ell,m} = v_1(\Delta W_{a,\ell,m}), \qquad s_a(x) = \text{Agg}_{\ell,m} \left| \left\langle v_{a,\ell,m}, h^0_{\ell,m}(x) \right\rangle \right|.$$

`SPECTR`: Spectral Routing and Merging for Multi-LoRA Inference. SPECTR uses the response norm of spectral LoRA factors on token hidden states. Let $\Delta W_{a,m} = B^\star_{a,m} A^\star_{a,m}$. The token- and module-level traces are used for top-k routing and aggregation:

$$r_{a,m,t}(x) = \left\| A^\star_{a,m} h^0_{m,t}(x) \right\|_2, \qquad s_a(x) = \text{Agg}_{m,t}\, r_{a,m,t}(x).$$

`LAG`: Arrow Retrieval plus SPECTR Reranking for Per-token Multi-LoRA Routing. LAG first uses an Arrow-style direction score to retrieve candidate adapters and then applies a SPECTR-style norm for reranking. Let $\mathcal{C}_{m,t} = \text{TopK}_a \left| \langle a^\star_{a,m,1}, h^0_{m,t}(x) \rangle \right|$ denote the retrieved candidate set:

$$s_{a,m,t}(x) = \left\| A^\star_{a,m} h^0_{m,t}(x) \right\|_2, \qquad a \in \mathcal{C}_{m,t}.$$

The final adapter is selected by the reranking score, and selected scores are aggregated across tokens and modules.

C.2 Existing Routers as Instances of the PMD Framework

The previous section specifies the implemented baseline scores. Here, we provide a complementary interpretation of representative zero-shot routers under the PMD framework. Rather than redefining their full implementations, we identify the response object $\rho_k(q)$ and decoder D that each method implicitly uses. This clarifies the position of PMDRouter: prior routers can be viewed as using restricted, spectral, or proxy response objects, while PMDRouter uses the full query-conditioned linear response and decodes it with scale-normalized response energy.

Notation. Let $h(q) \in \mathbb{R}^d$ denote the query-conditioned backbone activation. Each LoRA L_k has low-rank update $\Delta W_k = B_k A_k$, and we write its singular value decomposition as $\Delta W_k = \sum_{i=1}^r \sigma_{k,i} u_{k,i} v_{k,i}^\top$. A PMD router specifies a response object $\rho_k(q) = \text{Resp}(h(q), \theta_k)$ and a decoder $s_k(q) = D(\rho_k(q))$.

Arrow. Arrow can be interpreted as a rank-1 PMD path. It keeps only the dominant right singular direction of the LoRA update and reads the corresponding projection coefficient from the query activation:

$$\rho_k^{\mathrm{Arrow}}(q) = \sigma_{k,1} u_{k,1} \langle v_{k,1}, h(q) \rangle.$$

Thus, Arrow uses a truncated response object and a decoder that retains only the top-singular projection coefficient.

SpectR. SpectR can be interpreted as a spectral PMD path. Instead of keeping only the top component, it aggregates multiple singular components with a fixed spectral weighting:

$$\rho_k^{\mathrm{SpectR}}(q) = \sum_{i=1}^{r} f(\sigma_{k,i}) u_{k,i} \langle v_{k,i}, h(q) \rangle.$$

When f is the identity, this recovers the full linear response $B_k A_k h(q)$. Thus, SpectR can be viewed as a spectral response constructor with a norm-style decoder.

LAG and SEQR. LAG and SEQR correspond to proxy-based PMD paths. LAG constructs a static adapter-side proxy z_k and scores it by $s_k^{\mathrm{LAG}}(q) = \mathrm{sim}(\mathrm{emb}(q), z_k)$, so its response object is the query-independent proxy $\rho_k^{\mathrm{LAG}}(q) = z_k$. SEQR makes this proxy partially query-conditioned by using $\rho_k^{\mathrm{SEQR}}(q) = \tilde{z}_k(c(q))$ and scoring it through an external similarity function. Both methods introduce query dependence through proxy matching rather than by directly applying the LoRA operator $B_k A_k$ to the backbone activation $h(q)$.

Position of PMDROUTER. PMDROUTER uses the full query-conditioned linear response:

$$\rho_k^{\mathrm{PMDRouter}}(q) = B_k A_k h(q) = \sum_{i=1}^{r} \sigma_{k,i} u_{k,i} \langle v_{k,i}, h(q) \rangle.$$

It then decodes this response with scale-normalized response energy:

$$D^{\mathrm{PMDRouter}}(\rho_k, q) = \frac{\|\rho_k\|_2^2}{\|h(q)\|_2^2 \cdot \|B_k A_k\|_F^2 + \epsilon}.$$

Compared with Arrow, PMDRouter retains all low-rank response directions. Compared with SpectR, it avoids hand-designed spectral reweighting and decodes the full response after scale normalization. Compared with LAG and SEQR, it constructs the routable object by directly applying the LoRA update to the query-conditioned backbone activation.

Summary and interpretation. Table 6 summarizes these reductions as PMD interpretations rather than strict equivalences. Overall, PMD shifts routing separability from global adapter descriptors to query-conditioned responses, so globally similar LoRA memories can still be separated by their induced responses under a specific query, consistent with Figure 3.

D ADDITIONAL METHOD AND EXPERIMENTAL DETAILS

D.1 DIFFERENT DATASET

D.2 DIFFERENT TRAINING OBJECTIVES

Standard task loss. For each training example, we construct supervision only on assistant-answer tokens by masking prompt, user, and padding tokens with the ignore index -100. Let $\Omega = \{t : y_t \neq -100\}$ denote the supervised answer-token positions. The standard training objective is the causal language modeling loss:

$$\mathcal{L}_{\mathrm{task}} = -\frac{1}{|\Omega|} \sum_{t \in \Omega} \log p_\Theta(y_t \mid y_{<t}, x).$$

Router	Response object $\rho_k(q)$	Decoder $D(\rho_k)$
Arrow (Ostapenko et al., 2024)	Rank-1 truncated linear response	Top-singular projection coefficient
SpectR (Fleshman & Van Durme, 2025c)	Spectrally re-weighted linear response	Norm-style aggregation
LAG (Fleshman & Van Durme, 2025a)	Static or proxy adapter representation	External proxy similarity
SEQR (Fleshman & Van Durme, 2025b)	Contextualized adapter proxy	External proxy similarity
PMDROUTER (ours)	**Full linear response** $B_k A_k h(q)$	**Scale-normalized response energy**

Table 6: Existing zero-shot LoRA routers can be viewed as PMD instances by specifying the response object $\rho_k(q)$ and decoder D. PMDROUTER uses the full query-conditioned linear response with a scale-normalized energy decoder.

Method	Backbone	P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	P11	P12	P13	P14	P15	Avg.
LAG	Llama-3.1-8B	0.700	0.400	0.000	0.033	0.633	0.300	0.633	0.133	0.800	0.700	0.333	0.900	0.333	0.367	0.333	0.440
LAG	Qwen3-4B-Instruct	0.367	0.300	0.400	0.233	0.767	0.633	0.667	0.500	0.800	0.833	0.500	0.867	0.667	0.700	0.733	0.598
LAG	Qwen3-8B	0.567	0.400	0.667	0.233	0.833	0.667	0.667	0.567	1.000	0.833	0.500	0.733	0.567	0.667	0.633	0.636
PMDROUTER	Llama-3.1-8B	0.733	0.100	0.567	0.333	0.933	0.533	0.700	0.500	0.700	0.700	0.833	0.733	0.533	0.567	0.533	0.600
PMDROUTER	Qwen3-4B-Instruct	0.600	0.333	0.500	0.367	0.933	0.567	0.733	0.433	0.600	0.667	0.800	0.767	0.567	0.633	0.700	0.613
PMDROUTER	Qwen3-8B	0.833	0.233	0.467	0.433	0.933	0.533	0.733	0.533	0.767	0.733	0.833	0.767	0.567	0.733	0.767	0.658

Table 7: Per-class routing accuracy on PAPERQA. Columns P01–P15 correspond to `paper_01`–`paper_15`. Both LAG and PMDROUTER use the matched train-split calibration setting.

Signal-writing objective. When training-side signal writing is enabled, we keep the same task loss and add auxiliary losses that shape the LoRA response signals later read by the PMD decoder. All auxiliary terms are computed from frozen-backbone activations, LoRA-induced responses, or loss-derived signals, and therefore do not introduce an additional inference-time router. The overall objective is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \lambda_{\text{addr}}\mathcal{L}_{\text{addr}} + \lambda_{\text{energy}}\mathcal{L}_{\text{energy}} + \lambda_{\text{cal}}\mathcal{L}_{\text{cal}} + \lambda_{\text{ortho}}\mathcal{L}_{\text{ortho}} + \lambda_{\text{mono}}\mathcal{L}_{\text{mono}}.$$

Addressability regularizer. This term encourages the low-rank addressing activation to concentrate on distinguishable coordinates. Let $a_{\ell,k}(x) = A_{\ell,k}u(x)$ be the low-rank addressing activation, and let $p_i(x) = a_{\ell,k,i}(x)^2/(\sum_j a_{\ell,k,j}(x)^2+\epsilon)$ be its coordinate-wise energy distribution. The entropy-based addressability loss is

$$\mathcal{L}_{\text{addr}} = -\sum_i p_i(x)\log p_i(x).$$

When a top-1/top-2 separation term is used, $p_{(1)}(x)$ and $p_{(2)}(x)$ denote the largest two coordinates of $p(x)$, and m denotes the target margin. The margin penalty is

$$\mathcal{L}_{\text{margin}} = \max\big(0, m - (p_{(1)}(x) - p_{(2)}(x))\big).$$

Energy contrastive loss. This term encourages the target memory unit to produce a stronger query-conditioned response than a competing unit. Let $E_k(x)$ denote the response energy computed from LoRA-induced responses, let k^+ and k^- denote positive and negative memory units, and let $\tau > 0$ denote the temperature. The contrastive energy loss is

$$\mathcal{L}_{\text{energy}} = -\log \frac{\exp(E_{k^+}(x)/\tau)}{\exp(E_{k^+}(x)/\tau) + \exp(E_{k^-}(x)/\tau)}.$$

Energy calibration loss. This term explicitly calibrates positive and negative response magnitudes. Let c^+ and c^- denote the target positive and negative energy levels, and let β control the negative-energy penalty. The calibration loss is

$$\mathcal{L}_{\text{cal}} = \max(0, c^+ - E_{k^+}(x))^2 + \beta\max(0, E_{k^-}(x) - c^-)^2.$$

Method	Backbone	Ent.	Hist.	Sports	Geog.	Music	Pol.	Lit.	Bus.	Sci.	Bio.	Tech.	Food	Avg.
LAG	Llama-3.1-8B	0.633	0.893	0.880	0.480	0.960	0.587	0.747	0.627	0.600	0.827	0.927	0.640	0.733
LAG	Qwen3-4B-Instruct	0.807	0.747	0.927	0.407	0.867	0.840	0.707	0.780	0.813	0.927	0.860	0.833	0.793
LAG	Qwen3-8B	0.707	0.667	0.873	0.320	0.893	0.740	0.767	0.660	0.767	0.880	0.960	0.820	0.754
PMDROUTER	Llama-3.1-8B	0.727	0.800	0.873	0.533	0.933	0.627	0.913	0.653	0.687	0.940	0.873	0.673	0.769
PMDROUTER	Qwen3-4B-Instruct	0.667	0.847	0.873	0.480	0.900	0.720	0.887	0.727	0.740	0.967	0.860	0.707	0.781
PMDROUTER	Qwen3-8B	0.693	0.833	0.873	0.493	0.913	0.733	0.900	0.713	0.787	0.973	0.847	0.720	0.790

Table 8: Per-domain routing accuracy on NQ-DOMAINLORA. Both LAG and PMDROUTER use the matched train-split calibration setting.

Method	Backbone	BoolQ	Edit	ExtQA	GenCoT	Instr.	MC	MathCoT	NLI	Avg.
LAG	Llama-3.1-8B	0.593	0.013	0.093	0.000	0.080	0.267	0.980	0.493	0.315
LAG	Qwen3-4B-Instruct	0.067	0.467	0.087	0.000	0.647	0.393	0.860	0.087	0.326
LAG	Qwen3-8B	0.293	0.287	0.000	0.000	0.320	0.233	0.607	0.000	0.217
PMDROUTER	Llama-3.1-8B	0.187	0.413	0.493	0.993	0.233	1.000	1.000	0.987	0.663
PMDROUTER	Qwen3-4B-Instruct	1.000	0.893	1.000	0.993	0.707	0.980	1.000	0.607	0.897
PMDROUTER	Qwen3-8B	0.913	0.740	1.000	1.000	0.733	1.000	1.000	1.000	0.923

Table 9: Per-task routing accuracy on TASK-LORA. Both LAG and PMDROUTER use the matched train-split calibration setting.

Response orthogonality loss. This term reduces overlap between the current response and an anchor or competing response. Let $\rho_k(x)$ denote the current response, and let $\rho_a(x)$ denote an anchor or competing response. The response orthogonality loss is

$$\mathcal{L}_{\text{ortho}} = \left(\frac{\langle \rho_k(x), \rho_a(x) \rangle}{\|\rho_k(x)\|_2 \|\rho_a(x)\|_2 + \epsilon} \right)^2 .$$

A weight-space variant can be obtained by replacing the response vector with the corresponding LoRA update.

Monotonic alignment loss. This term aligns response energy with the actual improvement induced by the LoRA. Let $I_k(x) = \ell_{\text{base}}(x) - \ell_{\text{lora}}(x)$ denote the loss improvement of the LoRA-augmented model over the frozen backbone, and let $\text{Corr}(\cdot, \cdot)$ denote the Pearson correlation over a minibatch. The monotonic alignment loss is

$$\mathcal{L}_{\text{mono}} = -\operatorname{Corr}\big(E_k(x), I_k(x)\big).$$

D.3 DIFFERENT PATHS OF PMDROUTER

This section summarizes several PMD routing paths used in our implementation. All paths follow the same PMD principle: each candidate LoRA induces a query-conditioned response object, and a decoder maps this response into a final routing score.

Unified notation. Let x denote the input query and let $a \in \mathcal{A}$ denote a candidate LoRA memory unit. For layer ℓ, position-pooling rule p, and adapted module m, let the LoRA update be $\Delta W_{a,\ell,m} = B_{a,\ell,m} A_{a,\ell,m}$. Let $h_{\ell,p,m}(x)$ denote the backbone-side module input or hidden representation, where p may correspond to the assistant-prefix last token, question last token, question mean, or full-prompt mean. The LoRA-induced response is $\rho_{a,\ell,p,m}(x) = B_{a,\ell,m} A_{a,\ell,m} h_{\ell,p,m}(x)$. When multiple modules are used, module-level scores are summed and optionally normalized by $\text{Norm}(\cdot)$.

PMDRouter-Energy. PMD-Energy decodes the LoRA-induced response by its squared Euclidean norm. This score measures how strongly the current query activates the parametric response of candidate LoRA a. The final routing signal is

$$S_a^{\text{Energy}}(x) = \text{Norm}\left(\sum_{m \in \mathcal{M}} \|B_{a,\ell,m} A_{a,\ell,m} h_{\ell,p,m}(x)\|_2^2 \right).$$

Model	Method	Gold Top1	Oracle Top1	Oracle Top5
Qwen3-4B	GradCos	8.89	8.44	35.11
Qwen3-4B	LogitCE	8.67	8.22	35.33
Qwen3-4B	TopK-PseudoCE	8.44	8.89	76.44
Qwen3-4B	Energy	65.56	51.11	79.56
Qwen3-8B	GradCos	9.33	9.33	32.89
Qwen3-8B	LogitCE	9.33	8.00	37.78
Qwen3-8B	TopK-PseudoCE	1.56	2.44	73.56
Qwen3-8B	Energy	67.11	52.44	75.56
Llama-3.1-8B	GradCos	9.78	9.33	34.89
Llama-3.1-8B	LogitCE	11.33	8.44	34.22
Llama-3.1-8B	TopK-PseudoCE	11.11	8.67	73.56
Llama-3.1-8B	Energy	63.33	48.67	73.56

Table 10: Comparison of different PMD routing paths on PAPERQA. Gold Top1 reports routing accuracy against the gold LoRA, while Oracle Top1 and Oracle Top5 report oracle-based selection performance under top-1 and top-5 candidate settings.

PMDRouter-GradCos. PMD-GradCos compares the LoRA-induced response with a pseudo-loss descent direction. Given a pseudo answer y, let $\mathcal{L}_{\text{pseudo}}(x, y)$ denote the pseudo cross-entropy loss, and let $-\nabla_{h_{\ell,p,m}} \mathcal{L}_{\text{pseudo}}(x, y)$ denote the corresponding hidden-space descent direction. The final routing signal is

$$s^{\text{GradCos}}_{a,\ell,p,m}(x) = \frac{\left\langle B_{a,\ell,m} A_{a,\ell,m} h_{\ell,p,m}(x), -\nabla_{h_{\ell,p,m}} \mathcal{L}_{\text{pseudo}}(x, y) \right\rangle}{\|B_{a,\ell,m} A_{a,\ell,m} h_{\ell,p,m}(x)\|_2 \left\|\nabla_{h_{\ell,p,m}} \mathcal{L}_{\text{pseudo}}(x, y)\right\|_2 + \epsilon}.$$

PMDRouter-LogitCE. PMD-LogitCE uses a no-backward first-order approximation of pseudo-token cross entropy. For the first pseudo-answer token y, let $p_\theta(t \mid x)$ be the base-model next-token distribution and let W_t denote the output embedding or LM-head vector of token t. The logit-space direction is $W_y - \sum_t p_\theta(t \mid x) W_t$. The final routing signal is

$$s^{\text{LogitCE}}_{a,\ell,p,m}(x) = \left\langle B_{a,\ell,m} A_{a,\ell,m} h_{\ell,p,m}(x), W_y - \sum_t p_\theta(t \mid x) W_t \right\rangle.$$

PMDRouter-TopK-PseudoCE. PMD-TopK-PseudoCE uses PMD-Energy as a cheap recall stage and reranks the retained candidates by pseudo cross entropy. Let $\mathcal{C}_K(x)$ denote the top-K candidate set produced by the PMD-Energy prefilter, and let y_{pseudo} denote the pseudo answer used for reranking. The final routing decision is

$$a^*(x) = \arg \min_{a \in \mathcal{C}_K(x)} \text{CE}\left(M_{\theta+\Delta_a}(x), y_{\text{pseudo}}\right).$$

D.4 JUDGE SCORES ON PAPERQA

Method	PaperQA		
	Llama 3.1-8B	Qwen3 4B	Qwen3 8B
Text retrieval			
embedding	5.516	5.833	5.842
bm25	5.753	6.000	5.964
Backbone-only			
B-LogitVar	3.696	3.260	3.331
B-CtrL2	3.727	3.604	3.304
B-S4a	3.691	3.469	3.436
LoRA-only			
L-AlphaZ	4.164	3.564	3.507
L-HidL2-L4Z	4.240	3.496	3.413
L-HidL2-LastMed	3.836	3.231	3.411
Joint Backbone-LoRA			
J-Global	3.700	3.231	3.596
J-MLPGlobal	3.602	3.189	3.244
J-WAllZ	3.944	3.096	3.464
Arrow	4.453	4.080	4.184
SPECTR	3.862	3.300	3.362
LAG	4.522	4.993	5.176
PMDRouter	5.271	5.093	5.309

Table 11: PaperQA judge scores. **Color:** 1st.

E LIMITATIONS

This work has three main limitations. (1) We focus on one concrete separable response object, namely the response-energy surrogate related to estimated loss, while other PMD response objects, such as gradient alignment, output-space perturbations, or layer-specific response patterns, remain unexplored. (2) Although our results show that performance varies across backbones and memory granularities, we do not deeply analyze how model architecture affects document-level, domain-level knowledge, and task/skill-oriented memory routing. (3) PMD-Bench covers three representative EPM settings, but more scenarios, such as larger LoRA banks, multi-hop memory access, continually updated memory banks, and more diverse domain- or user-specific memory modules, should be studied in future work.

F BROADER IMPACTS

This work studies zero-shot routing for LoRA-based external parametric memory. Its potential positive impact is to make modular LLM deployment more efficient, maintainable, and reusable by reducing the need for separately trained routing components. Potential risks include misuse of specialized adapter banks, biased or harmful behavior encoded in adapters, and reliability issues caused by incorrect routing to irrelevant memory units. Although this work is methodological and does not target high-stakes deployment, private data, or human-subject applications, practical use of LoRA memory banks should include adapter auditing, routing-failure monitoring, and appropriate access or safety controls.

G CASES STUDY

Failure type	Diagnostic rule	Llama	Qwen3-4B	Qwen3-8B	Rank	Gap	Representative example
Response overlap	Gold is not separated by the matched response ranking.	64 / 35.6%	82 / 47.1%	73 / 47.4%	4.68	5.17	**Q:** What is the second main contribution listed in the paper? **A:** The second main contribution is the introduction of three key innovations: Denoising as Tree-Rollout, Guidance Levels as Meta-Actions, and Jumpy Denoising as Fast Simulation. *Route:* Llama-3.1-8B; `paper_01_q29`; `paper_01`→`paper_12`; rank > 5, gap 10.0
Decoder mismatch	Gold is near the top under the matched response ranking, but is not selected.	39 / 21.7%	51 / 29.3%	33 / 21.4%	2.00	5.26	**Q:** What are the two data-driven approaches mentioned by name for improving the reasoning ability of LLMs? **A:** RHO-1 and Phi-4. *Route:* Llama-3.1-8B; `paper_02_q02`; `paper_02`→`paper_05`; rank 2, gap 10.0
Output near-equivalence	Wrong top-1 route, but the downstream judge gap is small.	77 / 42.8%	41 / 23.6%	48 / 31.2%	3.46	-0.62	**Q:** What is the primary research gap identified concerning diffusion-based planners in the text? **A:** The primary research gap is that it remains uncertain how diffusion-based planners can achieve 'inference-time scalability'—that is, effectively enhance planning accuracy by scaling computation at inference time. *Route:* Llama-3.1-8B; `paper_01_q22`; `paper_01`→`paper_05`; rank 2, gap 0.0

Table 12: Failure taxonomy of PMDROUTER-AE on PAPERQA. Counts denote the number/share of failures for each backbone. Rank is the mean gold-LoRA rank in the matched response top-k list, and Gap is the mean judge-score difference between the gold and routed LoRA.