

Fleet Pilot: Fine-Grained Steering Empowers Real-World LLM Personalization

Kounianhua Du¹, Jianxing Liu¹, Kangning Zhang¹, Wenxiang Jiao², Yuan Lu²,
Jiarui Jin², Jianghao Lin¹, Weiwen Liu¹, Yong Yu¹, Weinan Zhang¹

¹Shanghai Jiao Tong University, ² Xiaohongshu Inc., Shanghai, China

© Higher Education Press 2026

Abstract With the rapid evolution of large language models (LLMs), there is an increasing demand for effective personalization techniques that can adapt model behavior to individual user preferences. Personalized large language models (LLMs) must infer a user’s query-dependent preferences from limited, heterogeneous, and evolving interaction histories. Existing approaches rely primarily on in-context personalization, parametric adaptation, or personalized reward modeling, but respectively incur recurring costs in context management, model optimization and deployment, or multi-candidate generation and scoring. These limitations become particularly restrictive when user observations are sparse and the preference evidence relevant to each query varies across contexts and over time.

To address these challenges, we propose FleetPilot, a fine-grained and instance-tailored steering framework that dynamically generates sample-level interference vectors from user data and injects them into the model’s forward pass for personalized adaptation. Our approach introduces: match contrast

that produces user-relevant signals, a fine-grained steering component that captures nuanced signals by hooking activations from attention and MLP layers, and a query-conditioned aggregation module that synthesizes these signals into contextually relevant enhancements. The method demonstrates high flexibility and data efficiency, excelling in fast-changing distribution and high data sparsity scenarios.

Extensive experiments across diverse scenarios, including short-to-long text generation, and web function calling, validate the effectiveness and compatibility of FleetPilot. Results show that our method consistently brings significant performance gains, and show extreme superiority in fast-shifting and data-sparse environments. Additionally, FleetPilot rids us of training cycles and incurs light overhead of storage or inference latency. These make FleetPilot a strong and real-world deployable method. Implementation is available ¹.

Keywords LLM, Personalization, Steering

¹<https://anonymous.4open.science/r/FleetPilot>

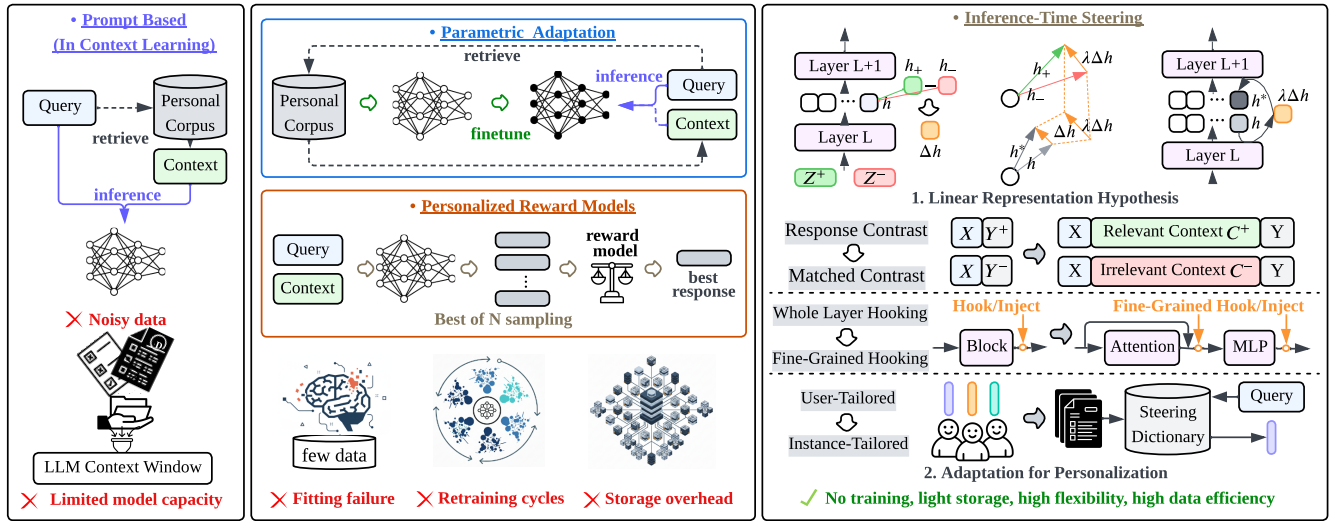


Fig. 1 Personalization paradigms and the design of FleetPilot. Left: prompt-based personalization supplies retrieved history through a context window, where noisy evidence and limited model capacity constrain its use. Middle: parametric adaptation and personalized reward modeling introduce optimization or candidate-scoring stages, with challenges in sparse-data fitting, model updates, storage, and best-of- N inference. Right: inference-time steering modifies intermediate activations in a frozen LLM, motivated by the linear representation hypothesis. FleetPilot adapts this principle to personalization through three design changes: response contrast to matched contrast, whole-layer hooking to fine-grained hooking, and user-tailored to instance-tailored steering.

1 Introduction

The advent of large language models (LLMs) [Dubey et al. \[2024\]](#), [Yang et al. \[2025\]](#) has catalyzed transformative advances across a multitude of domains. A particularly compelling frontier within this landscape is the pursuit of personalization—the ability to tailor model behavior and outputs to individual user preferences [Cai et al. \[2025\]](#), [Kumar et al. \[2024\]](#), [Liu et al. \[2025a\]](#), [Salemi et al. \[2023\]](#), [Wang et al. \[2025\]](#). Achieving effective personalization is paramount for deploying LLMs in real-world applications such as personal assistant [Guha et al. \[2015\]](#), [Wang and Sun \[2025\]](#), conversational agents [Liu et al. \[2025b\]](#), [Yusuf et al. \[2025\]](#), and adaptive learning systems [Iqbal et al. \[2025\]](#), [Saleem et al. \[2025\]](#).

A practical personalization system must infer a

user’s current, query-dependent preference from a limited and continuously evolving interaction history. Formally, for user u at time t , let $\mathcal{H}_u^t = \{(q_i, y_i)\}_{i < t}$ denote the observations available before a new query q_t . The system must generate an output aligned with the latent preference $p_u^t(y \mid q_t)$, although this distribution is only partially observed, may vary with the query, and may change over time. Production environments expose challenging regimes:

- **Limited Preference Observations (Data Scarcity).** For new or infrequent users, $|\mathcal{H}_u^t|$ can be extremely small, often fewer than ten interactions. The resulting cold-start regime [Zhang et al. \[2025b\]](#) provides weak and noisy evidence about individual preferences. An effective system must therefore achieve high statistical efficiency rather than rely on abundant user-specific supervision.
- **Non-Stationary Preferences (Temporal Adaptability).** The target distribution is generally time-dependent, i.e., $p_u^{t+1}(y \mid q) \neq p_u^t(y \mid q)$ after a

Received ; accepted

E-mail: kounianhuadu@sjtu.edu.cn

change in interests, goals, or context [Liu et al. \[2024\]](#). Since such changes may occur within days or even hours, personalization must incorporate recent evidence at inference time; adaptation tied to periodic optimization and redeployment can leave the system aligned with stale preferences.

- **Context-Dependent Preference Heterogeneity (Instance Specificity).** A single user can exhibit different preferences across queries, domains, and output forms, from short responses and long-form generation to structured tool calls [Jiang et al. \[2024\]](#). Consequently, the relevant signal is not fully captured by one static, user-level representation. The system must identify which subset of $\mathcal{H}_u^t$ is pertinent to q_t and apply the corresponding preference without allowing unrelated or conflicting history to dominate.

These challenges are coupled: scarce observations make drift difficult to estimate, while temporal and contextual heterogeneity further reduce the amount of history relevant to any individual query. We therefore require a practical method to be *data-efficient*, *rapidly updateable*, and *instance-specific*, while avoiding repeated training or multi-candidate inference.

Existing methodologies for LLM personalization can be broadly categorized into three paradigms. **Prompt-based** methods leverage in-context-learning (ICL) [Dong et al. \[2022\]](#) ability of llms, strategically inserting user profiles or historical interactions into the LLM context window for personalized output [Madaan et al. \[2022\]](#), [Qian et al. \[2024\]](#), [Shi et al. \[2025\]](#). **Parametric adaptation** methods include One-Peft-All-User [Lin et al. \[2024\]](#) that learns collaborative signals within the whole dataset by tuning LLMs on all the users’ data and One-Peft-Per-User [Tan et al. \[2024a,b\]](#), [Zhang et al.](#)

[\[2024\]](#) that performs parameter-efficient fine-tuning on user-specific data, tailoring user-specific weights to offer personalization. **Personalized reward modeling** methods learn preference-aware scoring functions that evaluate multiple candidate generations and select the output most aligned with the target user [Chen et al. \[2025a\]](#), [Ryan et al. \[2025\]](#), [Zhuang et al. \[2024\]](#).

Despite their effectiveness, the three paradigms expose different bottlenecks under these real-world requirements.

Prompt-based personalization is training-free and immediately updateable, but its performance is governed by the quality of context content used and the llm capacity [Science and Teams \[2025\]](#). Long, noisy records can introduce stale, contradictory, or irrelevant evidence that distracts the generator; truncation or profile summarization can instead discard minority and recently emerging preferences. Longer prompts also increase prefill latency and repeatedly expose raw personal records to the inference service [Liu et al. \[2025a\]](#), [Vuong et al. \[2021\]](#). [\[Lin et al., 2024\]](#) shows that these methods often reach their peak performances at limited context tokens, even far less than the llm capability upper bound.

Parametric adaptation encodes preferences more compactly, but converts each preference update into an optimization and deployment problem. A shared adapter must reconcile heterogeneous users and can suffer negative transfer, whereas one-adapter-per-user systems require user-specific supervision and incur training, storage, versioning, and adapter-switching costs that grow with the user population [Lin et al. \[2024\]](#), [Liu et al. \[2025a\]](#), [Tan et al. \[2024a,b\]](#). With only a few interactions, parameter estimates are unstable and prone to fitting incidental features; once preferences drift, stale adapters must be retrained or replaced rather than conditioned on

the latest interaction at inference time.

Personalized reward modeling moves personalization into an explicit scoring function, yet sparse pairwise feedback underdetermines a user’s context-dependent utility and may encourage reliance on spurious correlates. Models learned from pooled feedback can blur conflicting tastes, while per-user models face a severe bias–variance trade-off when feedback is scarce [Chen et al. \[2025a\]](#), [Ryan et al. \[2025\]](#), [Zhuang et al. \[2024\]](#). Moreover, reranking and rejection-sampling pipelines can only choose among generated candidates: their quality is bounded by candidate coverage, and increasing that coverage through best-of- N or rejection sampling multiplies generation and scoring cost [Gui et al. \[2024\]](#), [Liu et al. \[2023\]](#). A reward function estimated offline can also become miscalibrated as preferences or task distributions change, requiring fresh feedback and model updates.

Activation steering [Rimsky et al. \[2024\]](#), [Turner et al. \[2023\]](#), [Zou et al. \[2023\]](#) has been used in toxicity reduction and deception control, where high-level behavioral attributes are hypothesized to be encoded as approximately linear directions in the model’s residual-stream activation space (the linear representation hypothesis), so that a single contrastive difference vector suffices to identify each concept. Under this hypothesis, adding or subtracting a scaled version of that direction at an appropriate mid-to-late layer produces a monotonic, causally faithful shift of the corresponding behavior at inference time. It offers a complementary interface for personalization, which is attractive because it allows behavioral adaptation without gradient-based optimization or per-user model checkpoints. However, applying it to realistic personalization requires more than a fixed behavioral direction. Sparse and heterogeneous histories provide only indirect evidence of

user preferences, and the evidence relevant to one query may be inappropriate for another. The central challenge is therefore to extract useful personalization signals, preserve their effects at appropriate computational interfaces, and select the signals relevant to the current query.

To address this challenge, we propose FleetPilot, a fine-grained, instance-tailored steering framework built around three coordinated designs.

Matched contrast varies the conditioning context while holding a historical query and its recorded response fixed. By contrasting activations under user-relevant and user-irrelevant contexts, it captures changes associated with personalized conditioning without introducing differences in response content.

Fine-grained hooking constructs a pair of steering directions for each historical interaction by contrasting the model’s activations under user-relevant and user-irrelevant contexts. Rather than collapsing the contrastive signal into a single post-block direction, it separately captures the post-attention residual and the MLP output. These two hook points expose complementary computational pathways and preserve distinct preference-induced effects that whole-layer steering may conflate.

Query-conditioned aggregation then adapts these historical directions to the current query. At inference time, FleetPilot retrieves the most semantically relevant interactions from the user’s steering-vector dictionary and aggregates their attention- and MLP-side directions separately. The resulting pair of query-specific directions is applied through position-matched sequential injection, enabling the model to condition on the most pertinent preference evidence in a single auto-regressive generation. This design supports immediate adaptation to sparse observations and evolving interests without gradient

updates or per-user model deployment.

Overall, FleetPilot constructs a pair of query-specific interference vectors per user query by aggregating contextually relevant steering vectors, which are prebuilt from fine-grained contrasting model’s internal activations under the positive (personal) prompts and the negative (impersonal) prompts. Because the base model remains frozen and the query-specific steering directions are injected only transiently into intermediate activations, FleetPilot requires neither gradient-state storage nor per-user model checkpoints, while introducing only modest inference-time latency. Empirical studies show that FleetPilot outperforms all the competing baselines across different validation scenarios and show extreme superiority in dynamic and data-sparse environments.

Contributions are summarized below:

- **Comprehensive summarization of existing personalization paradigms and their fragility towards realistic challenges. Point out the correctness of inference-time steering as a training-free, flexible, robust, and low-cost paradigm.**

We analyze existing personalization paradigms, including prompt-based conditioning, parametric adaptation, and personalized reward modeling, through the lens of data scarcity, temporal preference drift, and query-dependent preference heterogeneity. This analysis identifies their respective limitations and motivates inference-time activation steering as a training-free, flexible, and resource-efficient interface for personalization, establishing the design requirements for robust adaptation under realistic conditions.

- **Methodology novelty:** We propose FleetPilot, which adapts activation steering to personalization through three coordinated designs. First, matched contrast holds historical queries and re-

sponses fixed while varying user-relevant and user-irrelevant contexts, extracting steering signals associated with personalized conditioning. Second, fine-grained hooking preserves separate directions from post-attention residuals and MLP outputs, while sequential dual-injection applies them at their corresponding computational interfaces for nuanced control. Third, query-conditioned retrieval and aggregation move beyond fixed concept-, behavior-, or user-level directions to instance-tailored interventions, flexibly selecting the historical signals relevant to each query without updating model parameters.

- **Empirical validation and analysis:** Experiments on various datasets including short content generation, long content generation, and personal web function calling demonstrate FleetPilot’s consistent superiority over all competing baselines. Particularly, FleetPilot excels in data-scarce scenario and fast-shifting distribution, with light overhead and no training cost. These justify FleetPilot as a robust and real-world deployable method.

2 Related Work

2.1 Prompt-Based Personalization

Prompt-based methods leverage the in-context learning capabilities of LLMs to personalize generation without changing model parameters. They retrieve relevant records from a user’s personal corpus or construct profiles that summarize historical preferences, and supply this information alongside the current query. LaMP [Salemi et al. \[2023\]](#) studies retrieval-augmented personalization across language tasks, while profile-based approaches compress interaction histories into more concise preference descriptions [Cai et al. \[2025\]](#). Memory and retrieval frameworks further investigate how to orga-

nize and retrieve contextual evidence [Madaan et al. \[2022\]](#), [Qian et al. \[2024\]](#), [Shi et al. \[2025\]](#).

These methods are flexible and can incorporate new records without retraining. Their effectiveness nevertheless depends on retrieval quality, context composition, and the model’s ability to use the supplied evidence. Finite context windows constrain how much history can be included, and adding noisy or weakly relevant records can reduce performance even within that limit [Lin et al. \[2024\]](#), [Science and Teams \[2025\]](#), [Vuong et al. \[2021\]](#). FleetPilot is complementary to this paradigm: retrieved text supplies explicit evidence, while query-conditioned steering supplies an additional activation-space intervention.

2.2 Parametric Adaptation

Parametric personalization incorporates user information through model optimization, commonly using parameter-efficient fine-tuning such as Low-Rank Adaptation (LoRA) [Hu et al. \[2022\]](#). One line of work trains a shared adapter using multiple users’ histories, allowing the model to learn collaborative signals and task-specific response patterns. ReLLa [Lin et al. \[2024\]](#) exemplifies this approach. Another line maintains user-specific adaptations. OPPU [Tan et al. \[2024b\]](#) combines personal PEFT parameters with non-parametric knowledge from retrieval and profiles, while PER-PCS [Tan et al. \[2024a\]](#) composes shared personal parameter components to construct an adaptation suited to a target user’s history.

Compared with full fine-tuning, these methods reduce the number of trainable parameters, but adaptation still depends on optimization and sufficient training evidence. Shared adapters must accommodate heterogeneous users, whereas user-specific adapters introduce storage, loading, and maintenance

costs. Changes in user preferences can require renewed fitting or adapter recomposition. In contrast, FleetPilot leaves the base model fixed and represents personal evidence as a retrievable dictionary of activation directions. Updating this dictionary requires vector preparation rather than per-user gradient-based training.

2.3 Personalized Reward Modeling

Personalized reward modeling represents user preferences through a scoring function over candidate responses. Such scores can guide rejection sampling [Liu et al. \[2023\]](#) or best-of- N selection [Gui et al. \[2024\]](#). Hydra [Zhuang et al. \[2024\]](#), for example, learns user-specific scoring adapters from positive and negative examples. Other approaches use an LLM as a personalized judge, conditioning its evaluation on user information rather than necessarily training a separate reward model [Ryan et al. \[2025\]](#).

Learned reward models depend on the availability and coverage of preference feedback, while both learned and prompted judges must distinguish user-specific suitability across diverse outputs. In candidate-selection pipelines, quality is bounded by the available candidates, and increasing their number raises generation and evaluation costs. FleetPilot instead steers the generation process itself, without requiring explicit pairwise preference labels or multiple completed candidates for each inference query. This distinction concerns the illustrated inference pipeline rather than a claim that all uses of reward models require best-of- N decoding.

2.4 Inference-Time Activation Steering

Activation steering modifies intermediate representations to influence a model’s behavior without updating its parameters. Prior work studies additive

activation interventions and contrastive direction construction Frandsen et al. [2022], Rimskey et al. [2024], Turner et al. [2023], while representation engineering explores the monitoring and control of high-level model representations Zou et al. [2023]. Persona- and style-oriented approaches further investigate steering directions associated with behavioral traits or personalized generation Chen et al. [2025b], Zhang et al. [2025a].

Within this family, methods differ in how contrasts are constructed, where directions are extracted and injected, and whether the intervention varies across inputs. Figure 1 organizes the design of FleetPilot along these three axes. It contrasts contexts while holding a historical query and response fixed, preserves separate post-attention and MLP-output directions, and retrieves historical directions according to the current query. These choices distinguish its matched, fine-grained, instance-tailored formulation from response-contrast, whole-layer, or fixed user-level alternatives. The basic steering formulation is introduced next; the concrete construction and inference procedure are given in Section 4.

3 Preliminary

3.1 Personalization Setting

Let $\mathcal{M}_\theta$ denote a base language model with parameters θ . For user u , let

$$D_u = \{(X_i, Y_i)\}_{i=1}^{n_u} \quad (1)$$

denote the historical interactions available before the current query X , where X_i is a historical query and Y_i is its recorded response. The output Y may be natural-language text or a task-specific textual serialization of a structured action. The personalization objective is to generate a response aligned with the

user’s current, query-dependent preferences using the evidence in D_u .

The history provides observations rather than a complete specification of those preferences. Its relevance can vary across queries and over time. Under the temporal evaluation protocol, only information available before the evaluated interaction may be used for personalization. We write D_u without an explicit time index for simplicity. Historical responses are available during preparation; the unknown response to the current query is not available for retrieval or direction construction.

3.2 Personalization Interfaces

The paradigms in Figure 1 differ in the interface through which user evidence influences generation. Let $\phi(X, D_u)$ retrieve or summarize the personal context relevant to X , and let $\parallel$ denote textual concatenation. Prompt-based personalization conditions the model on

$$P_u(X) = X \parallel \phi(X, D_u), \quad Y \sim \mathcal{M}_\theta(\cdot \mid P_u(X)). \quad (2)$$

Actual inputs use the model’s formatting template; the concatenation notation abstracts away those formatting details.

For user-specific parametric adaptation, personal evidence is encoded in a parameter change δ_u :

$$Y \sim \mathcal{M}_{\theta+\delta_u}(\cdot \mid P_u(X)). \quad (3)$$

The prompt may be the query alone or may additionally contain retrieved context. Shared-adapter methods instead learn an adaptation across users and use conditioning information to distinguish their requests.

For reward-based candidate selection, let $\{Y^{(j)}\}_{j=1}^N$ be candidate responses generated for the current query and let $r_u(X, Y^{(j)})$ be a user-conditioned score.

Best-of- N selection takes the form

$$Y^* = \arg \max_{Y^{(j)}, 1 \leq j \leq N} r_u(X, Y^{(j)}). \quad (4)$$

The scorer may be learned from preference data or implemented by a prompted judge. This expression describes the candidate-selection setting, not every possible use of a reward model.

3.3 Activation Steering

Activation steering introduces personalization at an internal computational interface instead of requiring a parameter update. At a selected layer L , let $h_t^L \in \mathbb{R}^d$ denote the activation at token position t . A basic additive intervention is

$$\tilde{h}_t^L = h_t^L + \gamma \Delta^L, \quad (5)$$

where $\Delta^L \in \mathbb{R}^d$ is a steering direction and $\gamma \in \mathbb{R}$ controls its strength. The modified activation is passed to the remaining computation while θ remains unchanged. The layer, interface, token positions, and intervention strength are design choices rather than universal constants.

A common construction uses a contrastive pair of complete inputs Z^+ and Z^- :

$$\Delta^L = \bar{h}^L(Z^+) - \bar{h}^L(Z^-), \quad (6)$$

where $\bar{h}^L$ denotes an activation pooled over a specified token set. The meaning of this direction depends on what differs between the paired inputs and which activations are pooled. In a response contrast, the continuations differ; in the matched contrast used by FleetPilot, the historical query and response remain fixed while the conditioning contexts differ. Section 4 specifies the context construction, response-token pooling, and two extraction interfaces.

The linear representation hypothesis motivates additive steering by suggesting that some high-level attributes are approximately encoded as directions in activation space [Rimsky et al. \[2024\]](#), [Turner et al. \[2023\]](#). Separately, a local approximation characterizes how a sufficiently small intervention propagates. Let F_L map the activation at the selected interface to downstream logits, with the other inputs fixed. If F_L is twice continuously differentiable with bounded second derivative in a neighborhood of h , then for a perturbation δ within that neighborhood,

$$F_L(h + \delta) - F_L(h) = J_{F_L}(h)\delta + O(\|\delta\|_2^2), \quad (7)$$

where $J_{F_L}(h)$ is the local Jacobian. This approximation does not establish global linearity, guarantee a monotonic behavioral change, or show that a contrast isolates user preference. Steering quality still depends on the evidence, extraction interface, intervention strength, and downstream nonlinear computation.

For personalization, a fixed user-level direction can be replaced by a query-conditioned direction $\Delta_u^L(X)$ obtained from historical evidence. FleetPilot instantiates this idea with a dictionary of paired, fine-grained directions and query-dependent retrieval. It also retains retrieved textual context in the generation prompt, so activation-space steering and prompt-based conditioning act as complementary information channels. The sequential dual-injection procedure is detailed in Section 4.3.2.

4 Methodology

4.1 Overview

Despite the increasing demand for personalized LLMs services, existing methods face critical problems in dynamic and data-sparse environments. Limitations of existing methods include structural interference, slow adaptation, and context window

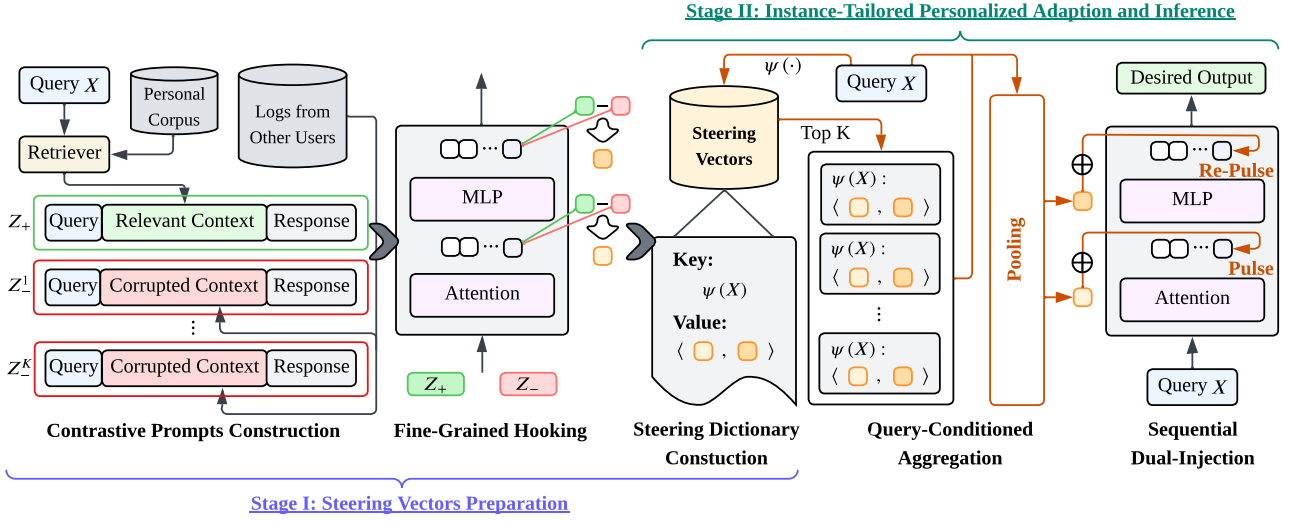


Fig. 2 Overview of FleetPilot. 1) Steering Vectors Preparation. During this stage, we construct contrastive prompts from user logs, where relevant context retrieved from personal corpus is concatenated with target query to form the positive sample and irrelevant context sampled from other users is concatenated with target query to form the negative sample. For each positive sample, we generate K negative samples. Each pair is then fed into the LLM in the teacher forcing mode, during which the response tokens’ activations of attention and MLP blocks are hooked. We store the difference between two sample activations of a pair to serve as the steering vector, with the text of each pair being the key for indexing convenience. 2) Instance-Tailored Personalized Adaption and Inference. During this stage, we sample from target user’s steering vectors set to interfere model for personalized output. Concretely, we rank the similarity between query text and sample pair text to select the top- k steering vectors, which are then aggregated and injected into LLM for personalized adaptation.

constraints, which hinder their applicability in real-world scenarios with fast-changing user interests and limited interaction data. In this paper, we propose FleetPilot, a novel inference-time personalization framework that treats personalization as a sample-level activation shift rather than a parametric update. FleetPilot operates through a two-stage pipeline designed for instance-aware personalization without model training:

- **Offline Steering Vectors Preparation:** For each user, we construct a personalized dictionary of steering vectors from historical interactions. Each vector is derived by contrasting model activations under user-relevant and user-irrelevant contexts, using a **Fine-Grained Hooking** mechanism that separately extracts signals from attention and MLP layers.
- **Online Inference with Instance-Tailored Dual**

Steering: During inference, for a target query, FleetPilot dynamically selects and aggregates the most relevant steering vectors from the user’s dictionary based on semantic similarity. The aggregated vectors are then injected into the model’s forward pass at a predefined layer via a **Sequential Dual-Injection** mechanism, enabling real-time, query-specific personalization. This approach ensures that each query receives a tailored activation shift, offering instance-level adaptation. The position-matched injection offers reinforced interference with subtle signals from each component, under the constraints of geometric consistency and linear compatibility.

These designs enable FleetPilot to adapt instantly with fine-grained signals to meet evolving user interests, with minimal memory and latency overhead, making it particularly suitable for dynamic and data-

sparse environments where traditional parametric methods struggle.

4.2 Steering Vectors Preparation

Based on the time-splitting setting, we use old user logs for steering vectors preparation.

4.2.1 Contrastive Prompt Construction

We construct contrastive prompt by varying the conditioning context while holding the historical query and target response fixed. For user u , let (X_i, Y_i) denote a historical interaction, where X_i is the query and Y_i is its recorded response. A relevant context is retrieved from the user's history, excluding the current interaction, whereas a negative context is sampled from other users' histories:

$$C_i^+ = \text{Retrieve}(X_i, D_u \setminus \{(X_i, Y_i)\}), \quad (8)$$

$$C_i^- = \text{Sample}(D_{\setminus u}). \quad (9)$$

Here, $D_{\setminus u}$ denotes the available histories of users other than u . The positive context supplies user-relevant evidence, while the negative context serves as a user-irrelevant contrast. Excluding the current interaction prevents its target response from being directly retrieved into the conditioning history. All context candidates must be restricted to the information available under the evaluation protocol.

Using the same formatting template, we construct

$$Z_i^+ = \text{Format}(X_i, C_i^+, Y_i), \quad (10)$$

$$Z_i^- = \text{Format}(X_i, C_i^-, Y_i). \quad (11)$$

The superscripts $+$ and $-$ refer to the contextual conditions, not to different responses: the same X_i and Y_i are supplied in both inputs. For structured outputs, Y_i denotes the recorded response in its task-specific textual serialization, which is also kept identical across the pair.

This construction examines how relevant versus irrelevant conditioning changes the model's representation of a fixed response, rather than comparing independently generated continuations.

4.2.2 Fine-Grained Activation Hooking

Consider a pre-normalized Transformer block at a selected layer L for a complete input sequence Z . Let $r_t^L(Z) \in \mathbb{R}^d$ denote the incoming residual state at token t , and let $r_{\leq t}^L(Z)$ denote the corresponding causal prefix. The block computes

$$a_t^L(Z) = \text{Attn}^L(\text{LN}_a^L(r_{\leq t}^L(Z))), \quad (12)$$

$$u_t^L(Z) = r_t^L(Z) + a_t^L(Z), \quad (13)$$

$$m_t^L(Z) = g^L(u_t^L(Z)), \quad (14)$$

$$v_t^L(Z) = u_t^L(Z) + m_t^L(Z), \quad (15)$$

where $g^L = \text{MLP}^L \circ \text{LN}_m^L$. These equations distinguish three interfaces: the post-attention residual u_t^L , the MLP output m_t^L before residual addition, and the post-block residual v_t^L . FleetPilot extracts directions from u_t^L and m_t^L , preserving the signal entering the MLP separately from the signal it contributes to the residual stream.

We process each complete input $Z_i^\pm$ with the frozen language model in teacher-forcing mode: the recorded response Y_i is supplied in both passes rather than sampled from the model. The **fine-grained hooking** vectors are then captured from the **matched contrast** mode.

- **Matched Contrast.** Let $\mathcal{T}_i^+$ and $\mathcal{T}_i^-$ denote the nonempty position sets corresponding to Y_i in the respective fully formatted and tokenized inputs. Only response-token activations are pooled; query, context, formatting-only, and padding positions are excluded.

Our working hypothesis is that, for a fixed query and response, contrasting relevant and irrelevant

contexts reveals an activation displacement associated with user-specific conditioning. Holding the query and response fixed removes differences in their content from the pair construction. Interpreting the remaining contrast as a preference-related displacement additionally requires other contextual effects to be balanced in expectation. For a fixed user, layer, and interface, consider the idealized decomposition

$$\bar{h}_{p,i}^{L,\pm} = \mu_p^L(X_i, Y_i) + \rho_{p,i}^{L,\pm} + \varepsilon_{p,i}^{L,\pm}, \quad (16)$$

where μ_p^L is a shared query–response component, $\rho_{p,i}^{L,\pm}$ is the preference-related contribution under each context, and $\varepsilon_{p,i}^{L,\pm}$ collects remaining contextual effects. If

$$\mathbb{E}[\varepsilon_{p,i}^{L,+} - \varepsilon_{p,i}^{L,-} \mid X_i, Y_i] = 0, \quad (17)$$

then the contrastive estimator satisfies

$$\mathbb{E}[\Delta_{p,i}^L \mid X_i, Y_i] = \mathbb{E}[\rho_{p,i}^{L,+} - \rho_{p,i}^{L,-} \mid X_i, Y_i]. \quad (18)$$

The expectation is over context-pair construction for the fixed interaction. This is a conditional identification statement, not a guarantee of the construction procedure: context topic, length, factual content, and induced positional shifts may contribute systematically to the contrast. Accordingly, the method uses empirical steering directions without claiming that every direction isolates an unbiased preference effect.

- **Fine-Grained Hooking.** The term *fine-grained* refers to retaining two within-block contrastive signals rather than collapsing them into a single post-block difference. Attention steering modulates the query-key-value interactions (capturing stylistic focus patterns), while MLP steering adjusts the feed-forward transformations (capturing lexical and semantic preferences). These operate on distinct representational subspaces. We therefore capture these complementary signals.

Concretely, for $p \in \{\text{attn}, \text{mlp}\}$, define

$$\bar{h}_{p,i}^{L,\pm} = \frac{1}{|\mathcal{T}_i^{\pm}|} \sum_{t \in \mathcal{T}_i^{\pm}} h_{p,t}^L(Z_i^{\pm}), \quad (19)$$

where $h_{\text{attn},t}^L = u_t^L$ and $h_{\text{mlp},t}^L = m_t^L$. The paired steering directions are

$$\Delta_{\text{attn},i}^L = \bar{h}_{\text{attn},i}^{L,+} - \bar{h}_{\text{attn},i}^{L,-}, \quad (20)$$

$$\Delta_{\text{mlp},i}^L = \bar{h}_{\text{mlp},i}^{L,+} - \bar{h}_{\text{mlp},i}^{L,-}. \quad (21)$$

The two response representations are averaged separately before subtraction.

4.2.3 Steering Dictionary Construction

To support query-conditioned retrieval, each record is associated with the representation

$$e_i = Z_i^+, \quad \mathbf{k}_i = \psi(e_i), \quad (22)$$

where ψ is the retrieval encoder, distinct from the frozen language model used for activation extraction. The retrieval text is the positive-context sequence, containing X_i , C_i^+ , and the historical response Y_i . The negative context is used to estimate the activation contrast but is not included in this retrieval key. The historical response is available during preparation; the target response to the current inference query is never used for retrieval. At layer L , the user-specific dictionary is

$$\mathcal{S}_u = \left\{ (\mathbf{k}_i, \Delta_{\text{attn},i}^L, \Delta_{\text{mlp},i}^L) \right\}_{i=1}^{N_u}. \quad (23)$$

Thus, each retained contrastive record provides both a semantic retrieval key and a pair of activation-space directions. This separation allows the historical evidence to be indexed in the encoder’s representation space while its behavioral effect is expressed in the base model’s hidden space. For clarity, the subsequent formulation uses one selected layer; the same construction applies separately to each layer when multiple layers are used.

4.3 Input-Aware Personalized Inference

4.3.1 Query-Conditioned Retrieval and Aggregation

Given a query X , FleetPilot selects relevant entries from $\mathcal{S}_u$ rather than applying a fixed average over the user's entire history. Let $\bar{X}$ denote the task-specific retrieval text extracted from X . We use a Contriever encoder with mean pooling over non-padding tokens and score each entry by the unnormalized inner product

$$s_i(X) = \psi(\bar{X})^\top \mathbf{k}_i. \quad (24)$$

For a nonempty dictionary and a positive retrieval budget K , let $K_u = \min(K, N_u)$. The selected index set is

$$\mathcal{I}_u(X) = \text{TopK}_{i \in \{1, \dots, N_u\}}(s_i(X), K_u), \quad (25)$$

where TopK returns the indices of the K_u highest scores. The selected directions are aggregated independently at the two interfaces:

$$\mathbf{s}_p^L(X) = \frac{1}{K_u} \sum_{i \in \mathcal{I}_u(X)} \Delta_{p,i}^L, \quad p \in \{\text{attn}, \text{mlp}\}. \quad (26)$$

The generation prompt additionally includes retrieved textual history:

$$P_u(X) = \text{Format}(X, H_u(X; B)), \quad (27)$$

where $H_u(X; B)$ denotes history retrieved with a prompt budget B . The budgets B and K govern different information channels: the former determines explicit contextual evidence, whereas the latter determines activation-space steering. Their retrieved records need not coincide. This distinction makes the role of steering explicit even when it is combined with retrieval-augmented prompting.

4.3.2 Sequential Dual-Injection

Afterwards, we perform a sequential dual-injection for inference-time enhancement. Overall, this position-matched injection ensures:

- **Geometric consistency:** The intervention occurs at the same representational locus where the steering signal was derived.
- **Linear compatibility:** Per the linear representation hypothesis, additive vectors in the same latent space propagate without nonlinear distortion through subsequent layers.

Concretely, at layer L , each aggregated direction is applied at the same computational interface from which it was extracted. For a token position t receiving an intervention, sequential dual-injection computes

$$\tilde{u}_t^L = u_t^L + \alpha \mathbf{s}_{\text{attn}}^L(X), \quad (28)$$

$$\tilde{m}_t^L = g^L(\tilde{u}_t^L) + \beta \mathbf{s}_{\text{mlp}}^L(X), \quad (29)$$

$$\tilde{v}_t^L = \tilde{u}_t^L + \tilde{m}_t^L. \quad (30)$$

The modified block output is then passed to the remaining layers and the language-model head. The coefficients α and β independently regulate the attention-side and MLP-side interventions. Setting both coefficients to zero recovers the unsteered forward computation for the same generation prompt; setting either coefficient to zero yields the corresponding single-site intervention.

The order of these operations is essential. The attention-side perturbation first changes the residual entering the MLP, so the MLP is evaluated on $\tilde{u}_t^L$, not on the original u_t^L . The MLP-side direction is then added to this recomputed output before residual addition. The resulting operation is therefore not, in general, equivalent to adding $\alpha \mathbf{s}_{\text{attn}}^L + \beta \mathbf{s}_{\text{mlp}}^L$ to the unmodified block output.

In the cached dual-site configuration, these interventions are applied during both prompt processing and autoregressive decoding. This inference-time scope is distinct from the response-token ranges used to estimate the directions. Matching extraction and injection interfaces establishes where each vector acts; it does not require identical token ranges between preparation and inference.

5 Experiment

5.1 Setup

5.1.1 Datasets

To evaluate FleetPilot under different scenarios, we utilize datasets including short personalized content generation (headline generation²⁾), long personalized content generation (abstract writing³⁾), and personal web function calling (PersonalWAB⁴⁾) for evaluation. Each of them contains user id, corresponding profile, and interaction histories.

Table 1 Statistics of Used Datasets.

Datasets	#Users	Avg. Profiles	Avg. Text Length (per user)	Avg. Text Length (per sample)
Headline Generation	200	68.26	2,257	33
Abstract Writing	200	117.91	18,369	156
PersonalWAB (product)	200	60.72	20,348	335
PersonalWAB (interaction)	200	12.2	656	54

- **News-Headline Generation.** This dataset [Misra \[2022\]](#) evaluates the ability of a large language model to capture the stylistic patterns of an author, querying the LLM to generate a headline given the input news article.

- **Abstract Writing.** This dataset evaluates LLMs’ ability to distill complex ideas and generate accurate, concise, and coherent output over the span of multiple paragraphs on domain-specific tasks. The input of the model is the title of the paper along with some keywords to guide the content. The expected output is an abstract conditioned on the title and keywords in the user’s style [Kumar et al. \[2024\]](#).
- **PersonalWAB.** This dataset integrates personalized instruction comprehension and action execution, where LLMs must infer personalized user requirements and preferences to determine which Web function to call and formulate the corresponding function parameters. Subsequently, the results of these function calls are returned to users [Cai et al. \[2025\]](#).

We sample 200 users from the original dataset, based on the time-split setting, earlier question-answer (QA) pairs form the train set and the latter pairs form the test set. The statistics of used datasets are summarized in Table 1.

5.1.2 Baselines

We evaluate FleetPilot against: 1) Prompt-based method (In Context Learning) and 2) Parametric adaptation methods including One-Peft-All-User [Lin et al. \[2024\]](#) and One-Peft-Per-User [Tan et al. \[2024a,b\]](#). 3) Personalized reward models [Ryan et al. \[2025\]](#), 3) Steering-based baseline [Chen et al. \[2025b\]](#), [Zhang et al. \[2025a\]](#), [Zou et al. \[2023\]](#).

5.1.3 Metric

We use Rouge Values [Lin \[2004\]](#) and METEOR [Banerjee and Lavie \[2005\]](#) to measure the quality of content generation and use Overall Accuracy (ACC) to measure the quality of web function calling.

²⁾<https://lamp-benchmark.github.io/>

³⁾<https://longlamp-benchmark.github.io/download>

⁴⁾<https://github.com/HongruCai/PersonalWAB/tree/main/PersonalWAB>

Table 2 Main results on content generation and personal web function calling. All baselines are done with n-shot demonstrations, consistent with the ICL setting. (The symbol * indicates statistically significant improvement with p-value ≤ 0.001 . Hyperparameters sensitivity is discussed in Appendix 5.4.4.)

Datasets		Headline Generation			Abstract Writing			PersonalWAB
Methods		Rouge-1	Rouge-L	METEOR	Rouge-1	Rouge-L	METEOR	ACC
Direct	ZeroShot	0.1432	0.1297	0.1002	0.3503	0.2024	0.1999	0.6017
In Context Learning	k=1	0.1389	0.1287	0.0994	0.3648	0.2093	0.2298	0.6767
	k=3	0.1643	0.1515	0.1257	0.3701	0.2104	0.2443	0.8415
	k=5	0.1665	0.1506	0.1267	0.3771	0.2153	0.2517	0.8330
	k=10	0.1630	0.1488	0.1271	0.3768	0.2127	0.2559	0.7966
Parametric Adaptation	Rella	0.1635	0.1463	0.1263	0.3818	0.2141	0.2562	0.8330
	OPPU	<u>0.1779</u>	<u>0.1613</u>	<u>0.1348</u>	0.3848	0.2146	<u>0.2580</u>	0.8394
	PER-PCS	0.1763	0.1577	0.1255	0.3602	0.2169	0.2446	0.8402
Reward Models	SynthesizeMe	0.1672	0.1490	0.1288	0.3906	0.2156	0.2565	0.8275
Steering	REPE	0.1685	0.1531	0.1325	0.3783	0.2134	0.2470	<u>0.8501</u>
	PersonaVector (Response)	0.1714	0.1540	0.1339	<u>0.3954</u>	<u>0.2291</u>	0.2567	0.8116
	PersonaVector (All)	0.1700	0.1536	0.1306	0.3825	0.2169	0.2512	0.8308
	PersonaVector (Prompt)	0.1640	0.1481	0.1267	0.3834	0.2148	0.2494	0.8458
	StyleVector (Mean)	0.1694	0.1540	0.1345	0.3790	0.2178	0.2547	0.8137
	StyleVector (Logistic)	0.1712	0.1539	0.1342	0.3831	0.2136	0.2505	0.8436
	FleetPilot	0.1816*	0.1666*	0.1473*	0.3990*	0.2306*	0.2609*	0.8566*

Table 3 Impact of hooking strategy. (The symbol * indicates statistically significant improvement with p-value ≤ 0.001 .)

Datasets			Headline Generation			Abstract Writing			PersonalWAB
Methods			Rouge-1	Rouge-L	METEOR	Rouge-1	Rouge-L	METEOR	ACC
FleetPilot	Fine-Grained Hooking	Post-Attention	0.1752	0.1590	0.1384	0.3938	0.2271	0.2556	0.8588
		Post-MLP	0.1704	0.1544	0.1301	0.3826	0.2133	0.2503	0.8501
		Post-Block	0.1737	0.1575	0.1369	0.3946	0.2277	0.2556	0.8458
		Hybrid Strategy	0.1816*	0.1666*	0.1473*	0.3990*	0.2306*	0.2609*	0.8566

5.2 Research Questions

- **(RQ1)** Does FleetPilot outperform the competing baselines?
- **(RQ2)** Does fine-grained hooking bring performance gain?
- **(RQ3)** How is the robustness of FleetPilot against heterogeneous/fast changing distribution compared to other methods?
- **(RQ4)** How is the data efficiency of FleetPilot compared to other methods? Particularly, under high data sparsity?
- **(RQ5)** How is the overhead brought by FleetPilot? Is FleetPilot real-world feasible?
- **(RQ6)** How is the hyperparameters sensitivity of FleetPilot?

5.3 Main Results (RQ1-RQ2)

The main results of the experiments are included in Table 2.

(RQ1). From the results, we can see that FleetPilot outperforms all other baselines. Concretely, parametric methods show better performances than the prompt-based methods, showcasing the benefits of personalization in parameters to better digest user patterns. Furthermore, FleetPilot outperforms traditional personalized methods, justifying the superiority of the steering-based methodology.

(RQ2). To evaluate the efficacy of steering vectors beyond layer-wise granularity, we investigate different hooking strategies:

- **Post-Attention.** The direction is extracted from

Table 4 Results of different methods against heterogeneous distribution among temporal dimension.

(Headline Generation)		Time-Range 1			Time-Range 2			Time-Range 3		
Methods		Rouge-1	Rouge-L	METEOR	Rouge-1	Rouge-L	METEOR	Rouge-1	Rouge-L	METEOR
In Context Learning		0.1620	0.1362	0.1157	0.1291	0.1109	0.0823	0.1491	0.1368	0.1216
Parametric adaptation	OPPU	0.1751	0.1544	0.1245	0.1496	0.1339	0.1121	0.1560	0.1397	0.1425
Personalized Reward Models	SynthesizeMe	0.1583	0.1380	0.1109	0.1497	0.1355	0.0974	0.1551	0.1336	0.1295
Steering	REPE	0.1659	0.1517	0.1219	0.1435	0.1276	0.0929	0.1657	0.1474	0.1325
	PersonaVector	0.1709	0.1536	0.1220	0.1475	0.1363	0.0928	0.1642	0.1403	0.1341
	StyleVector	0.1556	0.1444	0.1118	0.1508	0.1350	0.0961	0.1602	0.1359	0.1272
	FleetPilot	0.1787	0.1554	0.1251	0.1531	0.1411	0.1219	0.1782	0.1424	0.1448

and injected into the post-attention residual u^L , after the attention skip addition and before the MLP normalization.

- **Post-MLP.** The direction is extracted from and injected into the raw MLP sub-layer output m^L , before its addition to the residual stream.
- **Post-Block.** The direction is extracted from and injected into the post-block residual $v^L = u^L + m^L$.
- **Hybrid Strategy.** Position-matched directions are injected sequentially at u^L and m^L , following Equations (28)–(30).

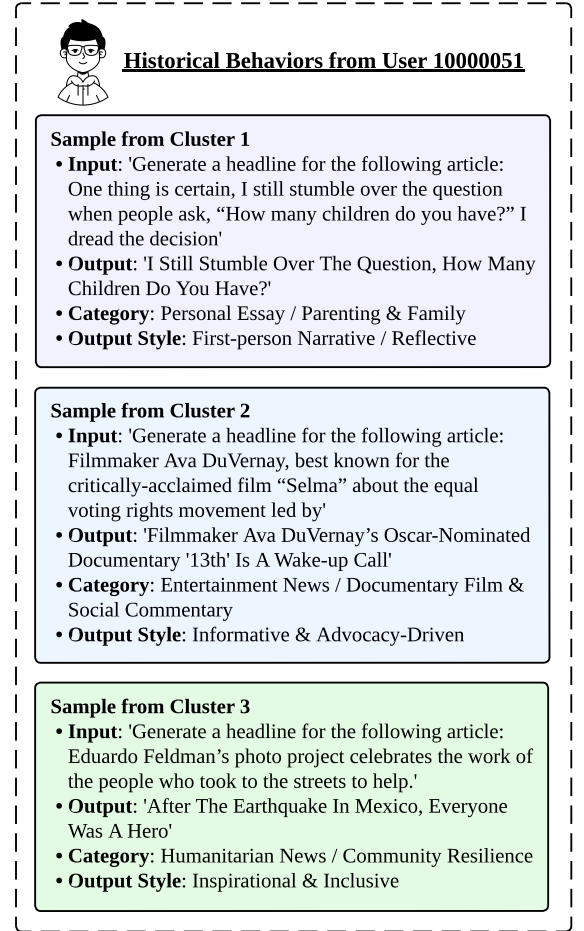
Table 3 reports the performance of the hooking variants. The Hybrid Strategy improves content-generation metrics by 0.36%–1.64% over the strongest single-site or post-block alternative ($p \leq 0.001$).

5.4 Ablation Studies

5.4.1 (RQ3) Robustness to Drifting Distribution

In this section, we study the robustness of different methods against drifting distributions. Particularly, we use the realistic temporal drift setting, where samples are split based on the timestamps and only earlier logs are available. We then test models’ performances on different time ranges. The results of different methods are summarized in Table 4.

From the results, we can see that FleetPilot outperforms all methods in each time-range, showing FleetPilot’ strong superiority in realistic temporal drift setting. Since the parametric methods cannot quickly realign to rapidly changing user preferences


Fig. 3 Illustration of the heterogeneous data.

due to their lock to sluggish and offline training loops. While FleetPilot offers inference-time and instance-tailored adaptation, and therefore it offers flexibility and robustness to heterogeneous distributions. This feature makes FleetPilot distinctive in scenario where user patterns change fast, which is a commonly-seen and important problem in LLMs personalization.

Table 5 Data efficiency analysis of different methods (Abstract Writing).

#Used Data		5% (Avg. 5.82)			15% (Avg. 17.46)			25% (Avg. 29.10)			50% (Avg. 58.72)		
Methods		Rouge-1	Rouge-L	METEOR	Rouge-1	Rouge-L	METEOR	Rouge-1	Rouge-L	METEOR	Rouge-1	Rouge-L	METEOR
Parametric adaptation	ReLLa	0.3701	0.2036	0.2382	0.3705	0.2134	0.2441	0.3730	0.2137	0.2437	0.3758	0.2185	0.2508
	OPPU	0.3764	0.2000	0.2488	0.3788	0.2033	0.2515	0.3794	0.2132	0.2523	0.3834	0.2200	0.2571
	PER-PCS	0.3600	0.2154	0.2445	0.3644	0.2158	0.2491	0.3597	0.2150	0.2441	0.3594	0.2152	0.2442
Personalized Reward Models	SynthesizeMe	0.3820	0.2091	0.2564	0.3853	0.2107	0.2584	0.3862	0.2109	0.2585	0.3875	0.2123	0.2575
	StyleVector	0.3827	0.2151	0.2499	0.3851	0.2158	0.2518	0.3819	0.2147	0.2488	0.3829	0.2130	0.2507
Steering	PersonaVector	0.3908	0.2256	0.2516	0.3908	0.2277	0.2543	0.3953	0.2268	0.2544	0.3950	0.2288	0.2580
	REPE	0.3760	0.2100	0.2446	0.3779	0.2103	0.2440	0.3806	0.2133	0.2466	0.3788	0.2143	0.2495
	FleetPilot	0.3920	0.2236	0.2542	0.3926	0.2247	0.2561	0.3956	0.2257	0.2563	0.3974	0.2270	0.2594

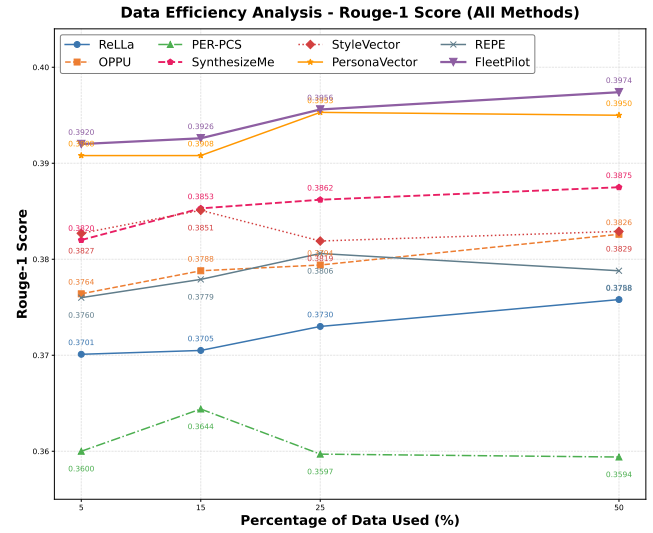
To further illustrate the discrepancy existing in the drifting distributions, we select different samples of the same user 10000051 as illustrated in Figure 3. These illustrations demonstrate the heterogeneous content themes, audience engagement strategies, and emotional tones existing in Question-Answer pairs of the same user.

5.4.2 (RQ4) Robustness to Data Scarcity and Data Efficiency Discussion

Despite the robustness of heterogeneous distribution, data sparsity is also an important problem in LLMs personalization, where only a small amount of data is available to conduct effective personalization. Therefore, we study the data efficiency of different methods in this section. Concretely, we use different amounts of personalized data to train models/serve as steering materials. The results are summarized in Table 5 and visualized in Figure 4.

The results show that FleetPilot remains effective with limited personalization histories, achieving the highest ROUGE-1 across all four data fractions. Particularly, under extreme data sparsity (<5 shots), steering presents strong superiority over the parametric methods. The increasing trend indicates that FleetPilot preserves much of its performance under substantial history reduction while still benefiting from additional evidence.

A plausible explanation for this effectiveness under sparse histories is that activation steering reuses

**Fig. 4** Data efficiency analysis of different methods.

behavioral structure already encoded in the pre-trained model through a constrained intervention, rather than estimating a parameter update from limited user-specific data. For FleetPilot, matched query–response contrasts may reduce variation unrelated to contextual conditioning, while response-token averaging may stabilize the extracted signals. Query-conditioned retrieval further selects potentially relevant historical directions instead of merging all available records into a single user-level representation. Together, these mechanisms offer a plausible account of its effective use of limited histories.

5.4.3 (RQ5) Overhead Discussion and Deployment Analysis

We sample 200 datapoints from each dataset and conduct the evaluation. The results are displayed in

Table 6.

Table 6 Latency Analysis.

Datasets (200 samples)	Headline Generation	
	Direct Inference	+ FleetPilot
Latency (s)	63.23	72.49
Datasets (200 samples)	Abstract Writing	
	Direct Inference	+ FleetPilot
Latency (s)	670.34	712.36
Datasets (200 samples)	PersonalWAB	
	Direct Inference	+ FleetPilot
Latency (s)	1161.88	1343.89

The inference overhead for FleetPilot is primarily attributed to retrieving the relevant steering vectors from the set, which is the most time-consuming step, and subsequently inserting the aggregated steering signal into the LLM’s forward pass. While latency varies with the steering vector size per user and the query’s token count, Table 6 displays the results of time latency analysis brought by FleetPilot, which is conducted on a single NVIDIA H100 device. Results demonstrate that FleetPilot incurs only a light overhead during inference. This efficiency offers a crucial advantage by circumventing the huge upload burden typically associated with transferring distinct personalized weights.

We further compare the per-user storage overhead of FleetPilot with that of personalized parametric adaptation methods. Let d denote the hidden dimension and N denote the number of steering vectors retained for each user. A steering vector has d elements, resulting in a storage complexity of $O(d)$. In our implementation, $d = 4096$ and each element is stored in bfloat16 format, which occupies two bytes. Thus, each steering vector requires

$$4096 \times 2 = 8192 \text{ bytes}, \quad (31)$$

or approximately 10 KB on disk after accounting for serialization overhead. Since fewer than 100

steering vectors are retained for each user, the total per-user storage remains below 1 MB. The resulting per-user storage complexity of FleetPilot is therefore

$$O(Nd). \quad (32)$$

By contrast, personalized parametric adaptation methods must store a separate set of LoRA weights for each user. Let r denote the LoRA rank and L denote the number of adapted Transformer layers. The storage complexity of a personalized LoRA checkpoint is

$$O(rdL), \quad (33)$$

up to constant factors determined by the number and dimensions of the adapted projection matrices. In our experiments, where $r = 8$ and $L = 32$, each user-specific LoRA weight dictionary occupies approximately 18 MB on disk.

Because $N \ll rL$ under our configuration, FleetPilot requires substantially less per-user storage than personalized LoRA methods: less than 1 MB compared with approximately 18 MB per user. This difference becomes increasingly consequential as the number of users grows. Moreover, updating a user profile in FleetPilot only involves constructing and storing lightweight steering vectors, without gradient computation or optimizer-state storage. Parametric adaptation methods, in contrast, require iterative training and the associated computational and memory overhead for every personalized model update. Together with the high flexibility and robustness validated above, FleetPilot is a real-world feasible method that offers superior performance gains under scarce and shifting environment.

5.4.4 (RQ6) Hyperparameters Analysis

We perform grid search on (α, β) in a range of $(0.6, 1.4) \times (0.6, 1.4)$. The results are visualized in Figure 5. Concisely, we compute the variances

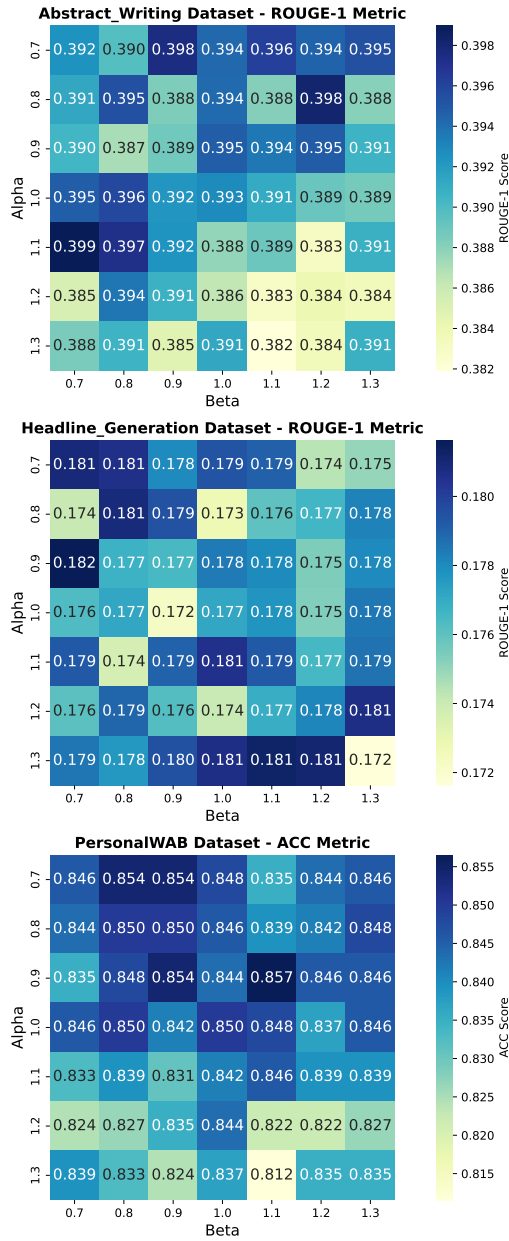


Fig. 5 Hyperparameter sensitivity analysis.

of the values and find that resulting performances show low variance: 0.000019 on abstract writing, 0.000006 on headline generation, and 0.000092 on PersonalWAB.

We further conduct one-way/two-way analysis and visualize the main effects analysis in Figure 6, Figure 7, and Figure 8. From the anova analysis, we can find that α is the only systematically impactful parameter, with $\alpha \leq 1.1$ as a general guideline.

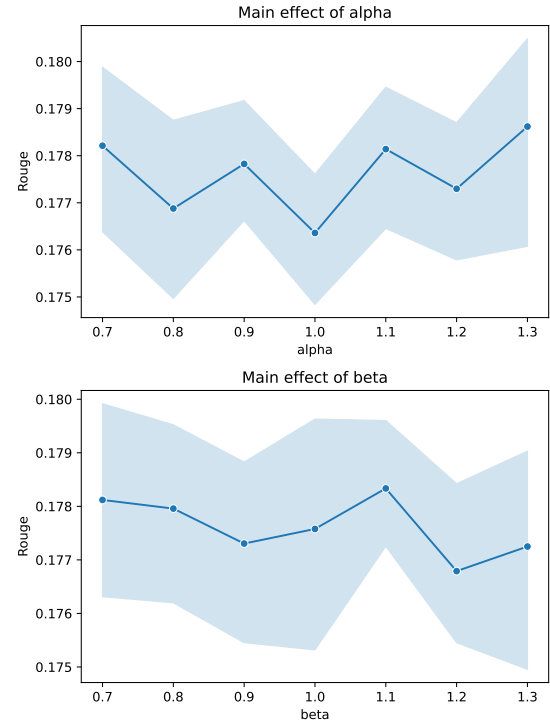


Fig. 6 Main effects analysis on Headline Generation.

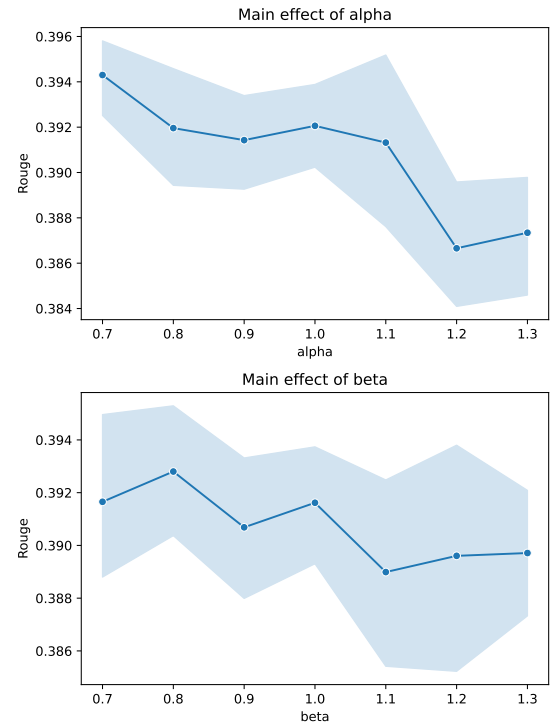


Fig. 7 Main effects analysis on Abstract Writing.

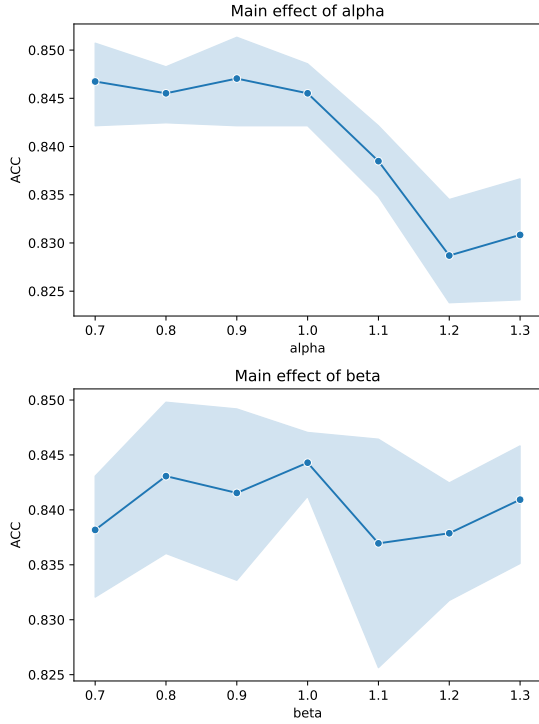


Fig. 8 Main effects analysis on PersonalWAB.

6 Conclusion

In this paper, we present FleetPilot, a novel inference-time personalization framework that addresses the critical challenges of data efficiency and adaptability in LLM personalization. Unlike traditional parametric methods that require extensive fine-tuning, FleetPilot operates through activation steering at inference time, making it particularly suitable for scenarios with fast-changing user preferences and limited training data. The core innovations of FleetPilot include: (1) *matched-contrast* that produces user-relevant signals, (2) *fine-grained hooking* that separately extracts steering signals from attention and MLP layers, capturing nuanced user preferences that whole-layer approaches miss; and (3) *query-conditioned aggregation* that dynamically retrieves relevant historical steering vectors based on semantic similarity to the current query, enabling instance-level adaptation without retraining. Extensive evaluation across diverse tasks demonstrates

FleetPilot’s superior performance over both prompt-based and parameter-efficient fine-tuning baselines. Notably, FleetPilot maintains strong performance even under extreme data scarcity (as few as 5-6 examples per user) and rapidly shifting user distributions, scenarios where traditional methods exhibit significant performance degradation.

Acknowledgements We thank the anonymous reviewers for their constructive comments and suggestions, which helped us refine the methodological presentation, strengthen the experimental evaluation, and clarify the contributions and limitations of this work.

References

- Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.
- Hongru Cai, Yongqi Li, Wenjie Wang, Fengbin Zhu, Xiaoyu Shen, Wenjie Li, and Tat-Seng Chua. Large language models empowered personalized web agents. In *Proceedings of the ACM on Web Conference 2025*, pages 198–215, 2025.
- Daiwei Chen, Yi Chen, Aniket Rege, Zhi Wang, and Ramya Korlakai Vinayak. PAL: Sample-efficient personalized reward modeling for pluralistic alignment. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL <https://openreview.net/forum?id=1kFDrYCuSu>.
- Runjin Chen, Andy Arditi, Henry Sleight, Owain Evans, and Jack Lindsey. Persona vectors: Monitoring and controlling character traits in language models. *arXiv preprint arXiv:2507.21509*, 2025b.

- Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, et al. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407, 2024.
- Abraham Frandsen, Rong Ge, and Vatsal Sharan. Extracting latent state representations with linear dynamics from language models. In *Proc. of ICML*, 2022.
- Ramanathan Guha, Vineet Gupta, Vivek Raghunathan, and Ramakrishnan Srikant. User modeling for a personal assistant. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, pages 275–284, 2015.
- Lin Gui, Cristina Gârbasea, and Victor Veitch. Bonbon alignment for large language models and the sweetness of best-of-n sampling. *Advances in Neural Information Processing Systems*, 37: 2851–2885, 2024.
- Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.
- Muddesar Iqbal, Sohail Sarwar, Muhammad Safyan, and Moustafa Nasralla. Personalized and adaptive e-learning systems for semantic web: a systematic review and roadmap. *International Journal of Web Information Systems*, 21(4):327–352, 2025.
- Yifan Jiang et al. Importance weighting can help large language models self-improve. *arXiv preprint arXiv:2408.09849*, 2024.
- Ishita Kumar, Snigdha Viswanathan, Sushrita Yerra, Alireza Salemi, Ryan A Rossi, Franck Dernoncourt, Hanieh Deilamsalehy, Xiang Chen, Ruiyi Zhang, Shubham Agarwal, et al. Longlamp: A benchmark for personalized long-form text generation. *arXiv preprint arXiv:2407.11016*, 2024.
- Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.
- Jianghao Lin, Rong Shan, Chenxu Zhu, Kounianhua Du, Bo Chen, Shigang Quan, Ruiming Tang, Yong Yu, and Weinan Zhang. Rella: Retrieval-enhanced large language models for lifelong sequential behavior comprehension in recommendation. In *Proceedings of the ACM Web Conference 2024*, pages 3497–3508, 2024.
- Jiahong Liu, Zexuan Qiu, Zhongyang Li, Quanyu Dai, Jieming Zhu, Minda Hu, Menglin Yang, and Irwin King. A survey of personalized large language models: Progress and future directions. *arXiv preprint arXiv:2502.11528*, 2025a.
- Jianghao Liu, Rong Shan, Chenxu Zhu, Kounianhua Du, Bo Chen, Shigang Quan, Ruiming Tang, Yong Yu, and Weinan Zhang. Relia: Retrieval-enhanced large language model for lifelong sequential behavior comprehension in recommendation. In *Proceedings of the ACM Web Conference 2024*, pages 3497–3508, 2024.
- Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J Liu, and Jialu Liu. Statistical rejection sampling im-

- proves preference optimization. *arXiv preprint arXiv:2309.06657*, 2023.
- Xingyu Bruce Liu, Shitao Fang, Weiyan Shi, Chien-Sheng Wu, Takeo Igarashi, and Xiang’Anthony’ Chen. Proactive conversational agents with inner thoughts. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, pages 1–19, 2025b.
- Aman Madaan, Niket Tandon, Peter Clark, and Yiming Yang. Memory-assisted prompt editing to improve gpt-3 after deployment. *arXiv preprint arXiv:2201.06009*, 2022.
- Rishabh Misra. News category dataset. *arXiv preprint arXiv:2209.11429*, 2022.
- Hongjin Qian, Peitian Zhang, Zheng Liu, Kelong Mao, and Zhicheng Dou. Memorag: Moving towards next-gen rag via memory-inspired knowledge discovery. *arXiv preprint arXiv:2409.05591*, 1, 2024.
- Nina Rimskey, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. Steering Llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15504–15522. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.828.
- Michael J Ryan, Omar Shaikh, Aditri Bhagirath, Daniel Frees, William Barr Held, and Diyi Yang. Synthesizeme! inducing persona-guided prompts for personalized reward models in llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8045–8078, 2025.
- Sadaf Saleem, Muhammad Umar Aziz, Muhammad Jawed Iqbal, and Shahid Abbas. Ai in education: Personalized learning systems and their impact on student performance and engagement. *The Critical Review of Social Sciences Studies*, 3 (1):2445–2459, 2025.
- Alireza Salemi, Sheshera Mysore, Michael Bender-sky, and Hamed Zamani. Lamp: When large language models meet personalization. *arXiv preprint arXiv:2304.11406*, 2023.
- Amazon Science and Research Teams. Context length alone hurts llm performance despite perfect retrieval. *Amazon Science Publications*, 2025. URL <https://www.amazon.science/publications/context-length-alone-hurts-llm-performance-des>
- Teng Shi, Jun Xu, Xiao Zhang, Xiaoxue Zang, Kai Zheng, Yang Song, and Han Li. Retrieval augmented generation with collaborative filtering for personalized text generation. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’25, page 1294–1304, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400715921. doi: 10.1145/3726302.3730075. URL <https://doi.org/10.1145/3726302.3730075>.
- Zhaoxuan Tan, Zheyuan Liu, and Meng Jiang. Personalized pieces: Efficient personalized large language models through collaborative efforts. *arXiv preprint arXiv:2406.10471*, 2024a.
- Zhaoxuan Tan, Qingkai Zeng, Yijun Tian, Zheyuan Liu, Bing Yin, and Meng Jiang. Democratizing large language models via personalized parameter-efficient fine-tuning. In *Proceedings*

- of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 6476–6491, 2024b.
- Alexander M Turner, Laura Thiergart, Gabriel Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*, 2023.
- Tung Vuong, Salvatore Andolina, Giulio Jacucci, and Tuukka Ruotsalo. Does more context help? effects of context window and application source on retrieval performance. *ACM Transactions on Information Systems (TOIS)*, 40(2):1–40, 2021.
- Haoming Wang, Boyuan Yang, Xiangyu Yin, and Wei Gao. Never start from scratch: Expediting on-device llm personalization via explainable model selection. *arXiv preprint arXiv:2504.13938*, 2025.
- Shaofeng Wang and Zhuo Sun. Roles of artificial intelligence experience, information redundancy, and familiarity in shaping active learning: Insights from intelligent personal assistants. *Education and Information Technologies*, 30(2):2525–2546, 2025.
- 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.
- Habeeb Yusuf, Arthur Money, and Damon Daylamani-Zad. Pedagogical ai conversational agents in higher education: a conceptual framework and survey of the state of the art. *Educational technology research and development*, 73(2):815–874, 2025.
- Jinghao Zhang, Yuting Liu, Wenjie Wang, Qiang Liu, Shu Wu, Liang Wang, and Tat-Seng Chua. Personalized text generation with contrastive activation steering. *arXiv preprint arXiv:2503.05213*, 2025a.
- Weizhi Zhang, Yuanchen Bei, Liangwei Yang, Henry Peng Zou, Peilin Zhou, Aiwei Liu, Yinghui Li, Hao Chen, Jianling Wang, Yu Wang, et al. Cold-start recommendation towards the era of large language models (llms): A comprehensive survey and roadmap. *arXiv preprint arXiv:2501.01945*, 2025b.
- You Zhang, Jin Wang, Liang-Chih Yu, Dan Xu, and Xuejie Zhang. Personalized lora for human-centered text understanding, 2024. URL <https://arxiv.org/abs/2403.06208>.
- Yuchen Zhuang, Haotian Sun, Yue Yu, Rushi Qiang, Qifan Wang, Chao Zhang, and Bo Dai. Hydra: Model factorization framework for black-box llm personalization. *Advances in Neural Information Processing Systems*, 37:100783–100815, 2024.
- Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023.