

SemFed: Semantic Federated Reinforcement Learning for Heterogeneous LLM Agents

Yuxuan Luan^{1,2,*}, Zilin Zhao^{1,2,*}, Yihan Zhang^{1,2}, Wei Wang³, Chenyu Sun^{1,2},
Lixiang Li^{1,2,†}, Haipeng Peng^{1,2}, Chaoning Zhang⁴, Lianku Xie⁵

¹ Information Security Center, State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China

² National Engineering Laboratory for Disaster Backup and Recovery, Beijing University of Posts and Telecommunications, Beijing 100876, China

³ Institute for Advanced Study, Tsinghua University, Beijing 100084, China

⁴ University of Electronic Science and Technology of China, Chengdu 611731, China

⁵ Information Institute of Ministry of Emergency Management, Beijing 100029, China

{yuxuan.luan,zhaozilin,yihan.zhang}@bupt.edu.cn, wei-wang@mail.tsinghua.edu.cn,

{suncy24,lixiang,penghaipeng}@bupt.edu.cn, chaoningzhang1990@gmail.com, lkxie@iiem.ac.cn

*Equal contribution. †Corresponding author.

Abstract

Training Large Language Model (LLM) agents via Reinforcement Learning (RL) in federated settings faces fundamental challenges, including model heterogeneity across clients, prohibitive communication costs and severe data heterogeneity. We propose a semantic federated learning framework called SemFed that addresses all challenges by communicating behavioral patterns rather than model parameters. Each client extracts compact semantic policy summaries from local training trajectories, which are aggregated at the server into shared, elite and anti-patterns. These aggregated patterns are injected back into local training through different complementary channels. Since the communicated objects are structured behavioral abstractions, SemFed supports heterogeneous model architectures while reducing per-round communication to kilobytes. Experiments on ALFWorld and WebShop demonstrate that SemFed consistently outperforms strong federated baselines across multiple LLM backbones. The mixed backbone evaluation with Qwen2.5-1.5B, Qwen2.5-7B, and Llama-3.2-3B clients directly validates cross-architecture collaboration without parameter alignment.

1 Introduction

LLMs have evolved from passive text processors into autonomous agents that perceive environment states and execute multi-step actions in interactive scenarios (Yao et al., 2023; Shinn et al., 2023; Qin et al., 2024; Yao et al., 2022; Yang et al., 2024). Reinforcement learning has become the dominant paradigm for training such agents, and allows them to discover effective behavioral strategies through trial-and-error interactions with the

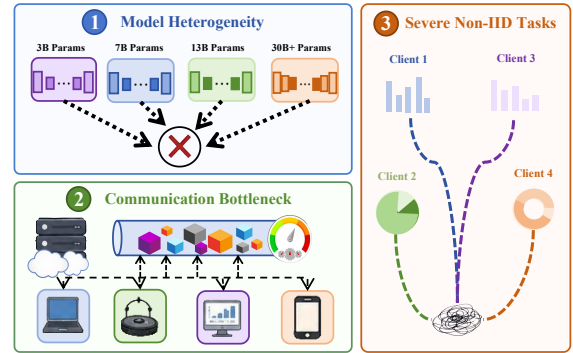


Figure 1: Three intertwined challenges in applying federated learning to LLM agents.

environment (Ouyang et al., 2022; Rafailov et al., 2023; Shao et al., 2024; Guo et al., 2025). Recent agent-tuning frameworks further demonstrate that RL-based optimization on interaction trajectories significantly outperforms supervised behavioral cloning on long-horizon tasks (Xi et al., 2025b; Song et al., 2024; Chen et al., 2024).

Despite these advances, training data for LLM agents is inherently distributed across organizations and individual users. Many task domains involve sensitive interaction logs that cannot be centralized due to privacy regulations and corporate data policies (Kairouz et al., 2021). Federated Learning (FL) offers a natural solution by enabling collaborative training without exposing raw data (McMahan et al., 2017). However, directly applying federated learning to LLM agents exposes three intertwined challenges that no existing method jointly addresses, as illustrated in Figure 1.

❶ **Clients often deploy models of varying architectures and scales**, while classical methods such

as FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020), and SCAFFOLD (Karimireddy et al., 2020) strictly require homogeneous structures across all participants. ② **Per-round communication overhead grows prohibitively large** when synchronizing billion-parameter models or even parameter-efficient adapters (Hu et al., 2022). Each round transmits hundreds of megabytes that overwhelm bandwidth-constrained edge deployments. ③ **Clients face highly skewed non-IID task distributions** that cause local optima to diverge from the global objective (Cheruiyot et al., 2025).

Our key observation is that the transferable knowledge across LLM agents lies in behavioral strategies rather than model parameters. For instance, when an agent discovers that a sequence reliably solves shopping tasks, this strategic knowledge benefits any agent regardless of the underlying architecture. This observation aligns with the principle of semantic communication that conveys task-level meaning rather than raw bit sequences (Yang et al., 2022; Xie et al., 2023) and motivates a fundamentally different design for federated agent learning. We accordingly elevate the federated communication substrate from the parameter space to the semantic behavior space.

We propose SemFed as a semantic federated reinforcement learning framework tailored to heterogeneous LLM agents. SemFed extracts structured semantic policy summaries from local training trajectories on each client, and aggregates them at the server into shared patterns, elite patterns and failure anti-patterns without accessing any raw data or model weights. The aggregated knowledge is then re-injected into local training through three complementary channels including prompt augmentation, reward shaping and trajectory relevance scoring. Because the communicated objects are structured behavioral abstractions rather than gradients or weights, SemFed supports heterogeneous model architectures while reducing per-round communication to kilobytes regardless of model scale. Our main contributions are as follows:

- We propose SemFed as a pioneering attempt to enable federated reinforcement learning for heterogeneous LLM agents through the communication of structured semantic behavioral patterns rather than model parameters.
- Structured Semantic Policy Extraction (SSPE) is designed to distill local training trajectories into compact policy summaries cover-

ing action preferences, success strategies, and failure-mode patterns through a unified action standardization mapping.

- Cross-Client Semantic Knowledge Aggregation (CSKA) is developed to identify shared patterns, elite-exclusive strategies, and high-risk anti-patterns across clients without accessing any raw data or model weights.
- Knowledge-Guided Training Signal Generation (KTSG) is introduced to inject aggregated knowledge into local RL pipelines along three complementary dimensions, namely prompt augmentation, reward shaping, and trajectory relevance scoring.

2 Related Work

2.1 Federated Learning for LLM

Recent efforts have begun adapting FL to large language models. FedIT (Zhang et al., 2024) pioneered federated instruction tuning on decentralized prompt-response pairs. Subsequent toolkits such as FedPETuning (Zhang et al., 2023) systematically benchmark parameter-efficient federated fine-tuning across diverse LoRA-style adapters. Personalized federated LLM methods including FedPepTAO (Che et al., 2023) mitigate client drift through prompt or adapter personalization, while FwdLLM (Xu et al., 2024) reduces on-device memory via backpropagation-free updates. To address privacy beyond data isolation, differentially private federated LLM fine-tuning (Liu et al., 2024) has been investigated under formal noise budgets. Conventional federated RL generally assumes structurally aligned policy networks and communicates gradients or model parameters (Qi et al., 2023). However, federated LLM fine-tuning mainly considers decentralized instruction or preference data without long-horizon environment interaction. FedAgent (Chen et al., 2026) is the closest parameter-space baseline for federated LLM-agent training, but relies on compatible LoRA structures. In contrast, SemFed transfers canonical behavioral abstractions without requiring parameter alignment.

2.2 Reinforcement Learning for LLM Agents

Reinforcement learning has emerged as the central technique for shaping LLM agent behavior on long-horizon interactive tasks. ArCHer (Zhou et al., 2024) introduces a hierarchical actor-critic

framework that scales RL fine-tuning to multi-turn agent rollouts. ETO (Song et al., 2024) alternates between successful and failed trajectories to improve exploration efficiency. RAGEN (Wang et al., 2025) extends GRPO-style optimization with rollout-aware advantage estimation for stochastic agent environments. StepAgent (Cheng et al., 2025) performs step-level credit assignment to alleviate the sparse-reward problem common in web and tool-use tasks. Verifiable-reward training such as RLVR (Lambert et al., 2025) further demonstrates that programmatic feedback signals can scale agent RL beyond human-preference supervision. Complementary surveys on LLM agents (Xi et al., 2025a) and federated reinforcement learning (Qi et al., 2023) map the broader landscape.

3 Methodology

3.1 Overview

In this paper, we propose a federated reinforcement learning framework tailored to heterogeneous LLM agents named SemFed, as illustrated in Figure 2. SemFed comprises three tightly integrated components. On the client side, SSPE module drives each client to perform local reinforcement learning over its private data while concurrently recording the complete interaction trajectories with the environment. Building upon the structured summaries uploaded from the clients, the CSKA module on the server consolidates these summaries without accessing any raw data or model weights, and distills the differentiated behavioral patterns into a global semantic knowledge base. Finally, the KTSG module translates the aggregated global knowledge into guidance signals adapted to local training pipelines, and disseminates them to the clients selected in the subsequent training round.

3.2 Problem Formalization

This paper considers a federated learning system composed of K clients $\{C_1, C_2, \dots, C_K\}$ and a central server S , where each client C_k holds a local private dataset $\mathcal{D}_k$ whose interaction tasks follow a heterogeneous distribution P_k . The global optimization objective of federated training is to learn a policy π_θ that maximizes the expected cumulative reward across all client scenarios:

$$J(\theta) = \sum_{k=1}^K \frac{|\mathcal{D}_k|}{|\mathcal{D}|} \mathbb{E}_{\tau \sim P_k} \left[\sum_{t=0}^T \gamma^t r_t(\tau) \right], \quad (1)$$

where $|\mathcal{D}| = \sum_k |\mathcal{D}_k|$ denotes the total amount of global data, P_k is the trajectory distribution of client C_k , $\gamma \in [0, 1]$ is the reward discount factor, $r_t(\tau)$ is the immediate reward at step t of trajectory τ , and T is the total length of a single interaction trajectory.

Within this framework, an LLM agent is defined as a textual policy π_θ that takes textual environment observations and historical interaction information as input. π_θ generates textual actions token by token autoregressively. A Dirichlet distribution is employed to model the heterogeneous task distribution across clients, the task category distribution received by client C_k satisfies:

$$P_k(g_m) = \frac{q_{k,m}}{\sum_{m'=1}^M q_{k,m'}}, \quad (2)$$

where $\mathcal{G} = \{g_1, g_2, \dots, g_M\}$ denotes the set of all task categories, and the category weight vector $q_k \sim \text{Dir}(\alpha \cdot \mathbf{1}_M)$, with α being the concentration parameter of the Dirichlet distribution and $\mathbf{1}_M$ an M -dimensional all-ones vector. A smaller α corresponds to a more skewed client task distribution and stronger heterogeneity. As $\alpha \rightarrow \infty$, all client task distributions degenerate into the i.i.d. setting.

3.3 Structured Semantic Policy Extraction

To reduce the communication overhead of transmitting model parameters in traditional FL, we propose the SSPE module. In each round, the selected clients first perform local reinforcement learning with GRPO (Shao et al., 2024). Given a task instruction x , GRPO samples a group of responses from the current policy and computes the group-relative advantage through intra-group normalization:

$$\hat{A}_i = \frac{r_i - \text{mean}(\{r_1, \dots, r_G\})}{\text{std}(\{r_1, \dots, r_G\}) + \epsilon}, \quad (3)$$

where $\{y_1, \dots, y_G\}$ is the response set sampled under the same instruction and $r_i = R(x, y_i)$ is the corresponding environmental reward. The functions $\text{mean}(\cdot)$ and $\text{std}(\cdot)$ compute the intra-group statistics of rewards. The constant ϵ prevents division by zero. The local training loss combines a clipped surrogate term and a KL regularization

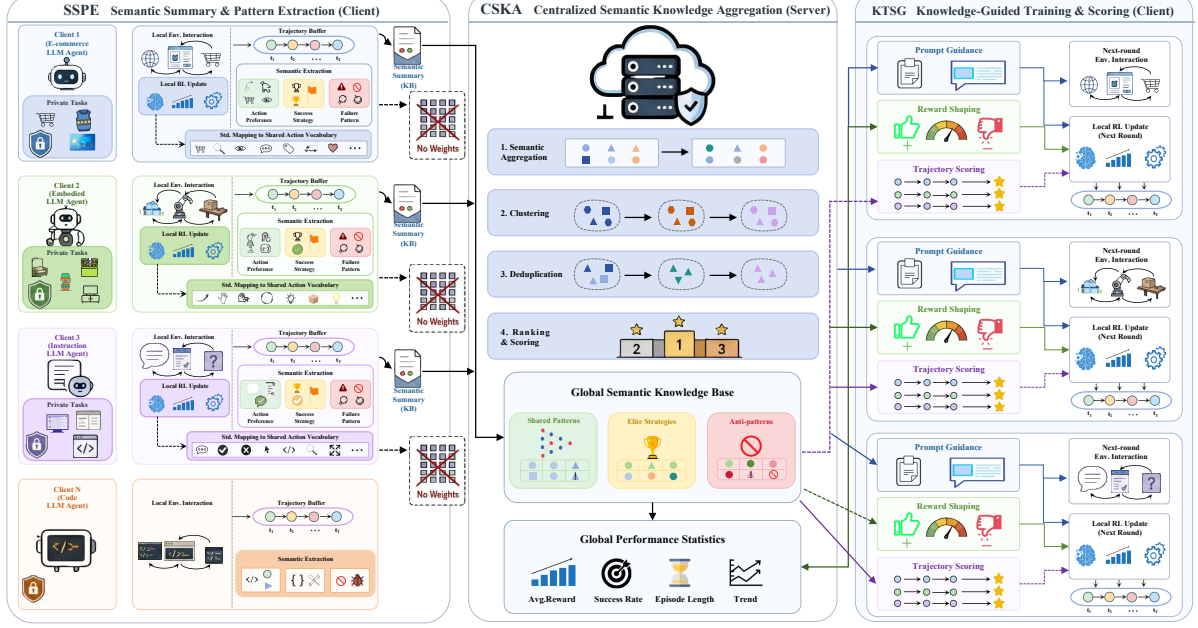


Figure 2: The overview of the SemFed framework which couples Structured Semantic Policy Extraction, Cross-Client Semantic Knowledge Aggregation and Knowledge-Guided Training Signal Generation within a unified pipeline.

term:

$$\mathcal{L}^{(k)}(\theta) = -\mathbb{E}_{x \sim \mathcal{D}_k} \left[\frac{1}{G} \sum_{i=1}^G \min \left(\rho_i(\theta) \hat{A}_i, \text{clip}(\rho_i(\theta), 1 - \varepsilon, 1 + \varepsilon) \hat{A}_i \right) \right] + \beta D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}), \quad (4)$$

where $\rho_i(\theta) = \pi_\theta(y_i | x) / \pi_{\theta_{\text{old}}}(y_i | x)$ is the importance ratio and ε is the clipping threshold. The term π_{ref} denotes the pre-trained reference model and β controls the regularization weight.

During training, SSPE simultaneously collects complete interaction trajectories and augments each with metadata:

$$\tilde{\tau}_j^{(k)} = \left(\tau_j^{(k)}, R_j^{(k)}, s_j^{(k)}, g_j^{(k)} \right), \quad (5)$$

$$\tau_j^{(k)} = \{(o_t, a_t, r_t)\}_{t=0}^{T_j}.$$

The cumulative return is $R_j^{(k)} = \sum_t r_t$. The indicator $s_j^{(k)} \in \{0, 1\}$ marks task completion and $g_j^{(k)} \in \mathcal{G}$ is the task category. All augmented trajectories are stored in the local buffer $\mathcal{B}_k$.

The client uploads only lightweight structured policy summaries rather than model parameters or gradients after training. We define three categories of behavioral patterns covering action preferences,

successful experiences, and failure-inducing behaviors. The action preference $p_{\text{pref}} = (\alpha, f_\alpha, \bar{r}_\alpha, c_\alpha)$ records the frequency f_α , average reward $\bar{r}_\alpha$, and triggering context c_α of action category α . The successful strategy $p_{\text{succ}} = (\sigma, f_\sigma, \bar{r}_\sigma, \kappa_\sigma)$ captures high-frequency consecutive action combinations $\sigma = (\alpha_i, \alpha_{i+1})$ extracted from successful samples. The failure pattern p_{fail} identifies erroneous actions that occur abnormally often in failed trajectories.

To unify free-form textual actions, we introduce a standardization mapping function ϕ :

$$\phi(a) = \begin{cases} \alpha_l, & a \in \mathcal{R}_l, l = 1, \dots, n-1; \\ \alpha_n, & \text{otherwise,} \end{cases} \quad (6)$$

where $\mathcal{A}_{\text{type}} = \{\alpha_1, \alpha_2, \dots, \alpha_n\}$ is the predefined standardized action set and $\mathcal{R}_l$ is the matching rule for action α_l . The category α_n absorbs non-canonical actions. In practice, ϕ is implemented using deterministic benchmark-specific parsers for ALFWorld and WebShop, with unmatched free-form actions mapped to an other category. Further implementation details are provided in the Appendix.

The decision criterion for a failure pattern is given as

$$f_\alpha^{\text{fail},(k)} > \lambda \cdot f_\alpha^{\text{succ},(k)}, \quad (7)$$

where $f_\alpha^{\text{fail},(k)}$ and $f_\alpha^{\text{succ},(k)}$ are the average occurrence frequencies of the target action in failure and

success trajectories. The constant $\lambda > 1$ is the identification threshold. The complete policy summary of client C_k at round t is then defined as

$$\mathcal{P}_k^{(t)} = (\bar{R}_k, \eta_k, \Delta_k, \mathbf{p}_k, \mathbf{d}_k), \quad (8)$$

where $\bar{R}_k$ is the average episode reward and η_k is the task success rate. The term $\Delta_k = \bar{R}_k^{(t)} - \bar{R}_k^{(t-1)}$ measures the reward improvement between consecutive rounds. The set $\mathbf{p}_k$ collects behavioral patterns and $\mathbf{d}_k$ encodes the local task distribution.

The summary is serialized as structured text and the per-round uplink cost satisfies:

$$\begin{aligned} \text{Cost}_{\text{upload}}^{(k)} &= |\text{Serialize}(\mathcal{P}_k^{(t)})| \\ &= O(P_{\max} \cdot L_p), \end{aligned} \quad (9)$$

where $P_{\max}$ is the maximum number of stored patterns and L_p is the average serialized length of a single pattern.

3.4 Cross-Client Semantic Knowledge Aggregation

To mine cross-client knowledge without exposing raw data or model weights, we design the CSKA module. The server fuses the uploaded summaries into a global guidance knowledge $\mathcal{G}^{(t)}$. Behavioral patterns commonly observed across clients are regarded as transferable shared knowledge. We define a pattern feature mapping function ψ to unify semantically equivalent patterns:

$$\psi(p) = \text{type}(p) \oplus \text{seq}(\text{actions}(p)), \quad (10)$$

where $\text{type}(p)$ denotes the pattern category and $\text{seq}(\text{actions}(p))$ preserves the original order of standardized actions. The operator $\oplus$ denotes string concatenation, so only patterns with the same type and ordered action sequence share the same feature code. The screening criterion for shared patterns is given as

$$\frac{|C_\kappa|}{K_t} \geq \delta_{\text{shared}}, \quad (11)$$

where K_t is the number of participating clients at round t . The set $C_\kappa = \{k : \exists p \in \mathbf{p}_k, \psi(p) = \kappa\}$ collects clients holding pattern κ . The constant $\delta_{\text{shared}} \in (0, 1)$ is the selection threshold.

To identify high-quality strategies exclusive to top-performing clients, an elite group is partitioned by the global average reward. The elite specificity score of a pattern is defined:

$$\text{ES}(p) = \frac{f_p^{\text{elite}}}{f_p^{\text{non-elite}} + \epsilon}, \quad (12)$$

where f_p^{elite} and $f_p^{\text{non-elite}}$ are the occurrence proportions of the pattern in elite and non-elite clients. A pattern with score above the threshold is treated as an elite-exclusive strategy. Globally collected failure patterns are deduplicated and ranked by severity:

$$\text{Severity}(p) = f_p^{\text{fail}} \cdot \kappa_p, \quad (13)$$

where f_p^{fail} is the occurrence frequency in failed trajectories and κ_p is the behavioral confidence. Integrating shared patterns and elite patterns and failure anti-patterns yields the global guidance information:

$$\mathcal{G}^{(t)} = (\bar{R}_{\text{global}}^{(t)}, \eta_{\text{global}}^{(t)}, \mathbf{p}_{\text{shared}}^{(t)}, \mathbf{p}_{\text{elite}}^{(t)}, \mathbf{p}_{\text{anti}}^{(t)}, \mathbf{h}^{(t)}), \quad (14)$$

where $\bar{R}_{\text{global}}^{(t)}$ and $\eta_{\text{global}}^{(t)}$ are the global average reward and global success rate. The sets $\mathbf{p}_{\text{shared}}^{(t)}$ and $\mathbf{p}_{\text{elite}}^{(t)}$ and $\mathbf{p}_{\text{anti}}^{(t)}$ contain the three pattern categories. The set $\mathbf{h}^{(t)}$ collects the training guidance prompts derived from the aggregated knowledge.

3.5 Knowledge-Guided Training Signal Generation

To make the aggregated global knowledge directly usable in local reinforcement learning, we propose the KTSG module. The transformation is realized through three mechanisms including prompt augmentation, reward shaping, and trajectory relevance scoring. Prompt augmentation converts the global knowledge into natural language statements. The statements are embedded into the system prompt and concatenated with environment observations to form the decision input. The optimized policy is expressed as

$$\pi_{\theta_k}^{(t+1)}(a | o, h) = \pi_{\theta_k}(a | [I^{(t)}; o], h), \quad (15)$$

where $I^{(t)}$ is the guidance prompt generated from the aggregated knowledge. The expression $[I^{(t)}; o]$ denotes the concatenation of the prompt and the environment observation. The term h is the historical interaction context.

Reward shaping rectifies the original environmental reward according to the matching results against behavioral patterns. This mechanism injects an explicit knowledge constraint into the local

training signal:

$$\begin{aligned}\tilde{R}(\tau) = R(\tau) &+ \sum_{p \in \mathbf{P}_{\text{shared}}} \Delta r_p^{\text{shared}} \mathbb{I}[\tau \models p] \\ &+ \sum_{p \in \mathbf{P}_{\text{elite}}} \Delta r_p^{\text{elite}} \mathbb{I}[\tau \models p] \\ &+ \sum_{p \in \mathbf{P}_{\text{anti}}} \Delta r_p^{\text{anti}} \mathbb{I}[\tau \models p],\end{aligned}\quad (16)$$

where $R(\tau)$ is the original environmental reward and $\tau \models p$ indicates that trajectory τ matches pattern p . The function $\mathbb{I}[\cdot]$ takes the value 1 on a successful match and 0 otherwise. The terms $\Delta r_p^{\text{shared}}$ and $\Delta r_p^{\text{elite}}$ are positive gains rewarding high-quality behaviors. The term Δr_p^{anti} is a negative penalty suppressing erroneous behaviors.

For each trajectory in the local buffer, we further compute a relevance score against the global knowledge:

$$\begin{aligned}\text{Rel}(\tau, \mathcal{G}^{(t)}) = &\sum_{p \in \mathbf{P}_{\text{shared}}} w_s \kappa_p \mathbb{I}[\tau \models p] \\ &+ \sum_{p \in \mathbf{P}_{\text{elite}}} w_e \kappa_p \mathbb{I}[\tau \models p] \\ &+ \sum_{p \in \mathbf{P}_{\text{anti}}} w_a f_p \mathbb{I}[\tau \models p],\end{aligned}\quad (17)$$

where w_s and w_e and w_a are the relevance weights for shared patterns and elite patterns and anti-patterns.

4 Experiments

4.1 Datasets

We evaluate SemFed on two interactive agent benchmarks. WebShop (Yao et al., 2022) is a simulated HTML e-commerce environment in which the agent completes purchase tasks through search, filter and selection actions. ALFWorld (Shridhar et al., 2021) is an embodied benchmark covering six household-activity categories, namely Pick & Place, Examine in Light, Clean & Place, Heat & Place, Cool & Place and Pick Two & Place. Detailed dataset statistics are provided in the Appendix.

4.2 Baselines and Configurations

SemFed is compared against representative federated, centralized, and isolated training baselines. FedAgent (Chen et al., 2026) combines federated GRPO (Shao et al., 2024) with LoRA-based parameter aggregation. FedAvg (McMahan et al., 2017)

averages LoRA adapter weights across clients, while FedProx (Li et al., 2020) adds proximal regularization to mitigate client drift. Local GRPO trains each client independently without communication. Centralized trains a single model on the union of all client training instructions under the same total interaction and optimization budget as the federated methods. Homogeneous experiments use the same backbone within each run, covering Qwen2.5-1.5B/3B/7B-Instruct (Hui et al., 2024) and Llama-3.2-3B-Instruct (Grattafiori et al., 2024), with Qwen2.5-1.5B-Instruct as the default backbone. The default homogeneous setting uses $K=100$ clients, samples $M=2$ clients per round, runs for 70 rounds, and adopts a Dirichlet concentration of $\tau=0.5$. For the mixed-backbone evaluation, 60 clients from Qwen2.5-1.5B, Qwen2.5-7B, and Llama-3.2-3B jointly participate in one federation, with six clients sampled per round. Success Rate (SR), Task Score (TS), and Communication Cost (Comm.) are reported as the primary metrics, with further details provided in the Appendix.

4.3 Overall Performance Comparison

Table 1 reports the per-subtask Success Rate on ALFWorld and the dual metrics on WebShop across four LLM backbones. On ALFWorld, SemFed attains the highest overall Success Rate under all four backbones, surpassing FedAgent by +3.2, +3.6, +2.9 and +3.4 points on Qwen2.5-1.5B, Qwen2.5-3B, Qwen2.5-7B and Llama-3.2-3B, respectively. The gains concentrate on the Cool and Pick2 subtasks, where SemFed exceeds FedAgent by up to 5.2 and 12.5 points (Qwen2.5-1.5B), and by 7.7 and 12.6 points (Llama-3.2-3B). Both subtasks involve rare or specialized action patterns that benefit from client specialization, which parameter aggregation tends to dilute through weight averaging. On WebShop with Qwen2.5-7B-Instruct, SemFed reaches a Task Score of 91.2 and a Success Rate of 71.5%, exceeding FedAgent by 2.2 and 2.6 points, respectively. The improvements remain stable across 1.5B–7B parameter scales and across Qwen and Llama families, indicating that semantic guidance is not coupled to a specific backbone configuration.

Table 2 further provides a complete comparison with standard federated baselines under the same Qwen2.5-1.5B setting. SemFed outperforms FedAvg, FedProx, and FedAgent by 6.6, 5.3, and 3.2 points on ALFWorld Success Rate, respectively. On WebShop, SemFed also exceeds FedAvg by 4.2

Table 1: Performance comparison on ALFWorld and WebShop across four LLM backbones. Pick, Look, Clean, Heat, Cool and Pick2 denote the six ALFWorld subtasks. Score and Succ. denote Task Score and Success Rate on WebShop.

Method	ALFWorld						WebShop		
	Pick	Look	Clean	Heat	Cool	Pick2	All	Score	Succ.
Qwen2.5-1.5B-Instruct									
Local (Client 21)	42.9	25.0	38.5	37.5	14.3	14.3	29.7	69.9	57.0
Local (Client 42)	50.0	37.5	76.9	25.0	42.9	14.3	45.3	75.1	53.1
Local (Client 84)	50.0	37.5	46.2	25.0	28.6	0.0	34.4	72.7	47.7
Centralized	64.3	37.5	69.2	50.0	42.9	28.6	51.6	79.9	57.8
FedAgent	80.0	75.0	53.8	37.5	83.3	50.0	64.1	83.2	61.7
SemFed	82.5	81.3	58.0	45.0	88.5	62.5	67.3	85.7	64.8
Qwen2.5-3B-Instruct									
Local (Client 21)	41.5	12.5	34.9	51.0	18.9	21.2	31.3	59.8	55.0
Local (Client 42)	46.5	37.5	24.4	15.0	33.7	33.3	28.2	61.3	59.3
Local (Client 84)	22.8	27.5	39.1	46.3	48.3	36.5	29.9	77.6	58.6
Centralized	94.1	80.0	64.3	42.9	50.0	22.2	62.5	70.0	53.9
FedAgent	95.5	62.5	49.7	47.5	85.3	45.1	65.2	85.5	63.1
SemFed	95.0	75.0	54.0	53.0	89.5	56.3	68.8	87.6	65.9
Qwen2.5-7B-Instruct									
Local (Client 21)	35.5	25.0	61.0	25.9	35.8	45.2	38.4	70.9	49.2
Local (Client 42)	29.0	45.0	18.8	25.6	15.9	38.0	42.1	78.2	33.6
Local (Client 84)	34.7	47.5	44.4	51.3	40.1	21.8	35.7	60.6	39.3
Centralized	93.7	82.5	71.5	47.9	63.2	31.9	73.3	78.8	64.7
FedAgent	94.5	85.0	56.0	62.5	86.7	42.8	75.5	89.0	68.9
SemFed	95.5	87.5	64.5	67.0	90.0	53.1	78.4	91.2	71.5
Llama-3.2-3B-Instruct									
Local (Client 21)	39.8	50.0	17.9	40.0	20.7	34.0	38.1	65.3	50.5
Local (Client 42)	18.2	55.0	41.9	34.3	41.0	25.0	35.0	67.0	51.0
Local (Client 84)	29.9	32.5	39.0	18.9	18.8	37.6	29.7	70.2	55.7
Centralized	72.4	62.5	59.3	45.2	53.7	27.9	54.9	72.3	56.2
FedAgent	83.7	57.5	60.6	55.9	65.3	24.9	61.2	74.4	57.8
SemFed	85.0	62.5	64.5	58.5	73.0	37.5	64.6	77.8	60.5

Table 2: Complete baseline comparison on Qwen2.5-1.5B-Instruct. Results are reported as mean $\pm$ standard deviation over three random seeds.

Method	ALFWorld SR	WebShop Score	WebShop SR
Local GRPO	36.5 $\pm$ 2.1	72.6 $\pm$ 1.8	52.6 $\pm$ 2.0
Centralized	51.6 $\pm$ 1.5	79.9 $\pm$ 1.2	57.8 $\pm$ 1.4
FedAvg	60.7 $\pm$ 1.3	81.5 $\pm$ 1.1	59.6 $\pm$ 1.3
FedProx	62.0 $\pm$ 1.2	82.3 $\pm$ 1.0	60.4 $\pm$ 1.2
FedAgent	64.1 $\pm$ 0.9	83.2 $\pm$ 0.8	61.7 $\pm$ 0.9
SemFed	67.3$\pm$0.8	85.7$\pm$0.7	64.8$\pm$0.8

points in Task Score and 5.2 points in Success Rate. Paired bootstrap tests further confirm that the main improvements over FedAgent are statistically significant ($p < 0.01$). Detailed confidence intervals are reported in Appendix.

4.4 Mixed-Backbone Federated Collaboration

The previous experiments evaluate SemFed across different backbone configurations, while model

heterogeneity in practical federations requires clients with incompatible architectures to collaborate within the same training run. We therefore construct a mixed-backbone federation with 60 clients, consisting of 20 Qwen2.5-1.5B-Instruct, 20 Qwen2.5-7B-Instruct, and 20 Llama-3.2-3B-Instruct clients. Six clients participate in each communication round, with stratified sampling selecting two clients from each backbone group. Each client maintains its own model parameters, LoRA adapter, optimizer, and local trajectory buffer, while only canonical semantic policy summaries are exchanged through the shared aggregation server.

Direct parameter averaging across these heterogeneous LoRA structures is undefined. We therefore introduce FedAgent Clustered as a feasible baseline, which aggregates parameters independently within each compatible backbone family

Table 3: Same-federation mixed-backbone evaluation. The federation contains Qwen2.5-1.5B, Qwen2.5-7B, and Llama-3.2-3B clients. ALF SR denotes ALFWorld success rate. Macro is averaged across the three backbone groups. Comm. denotes total communication over training.

Method	Qwen-1.5B	Qwen-7B	Llama-3B	Macro ALF SR	WebShop Score	WebShop SR	Comm.
Local GRPO	50.8±1.6	61.7±1.4	54.2±1.8	55.6±1.2	73.6±1.5	53.8±1.7	0
FedAgent-Clustered	63.4±1.1	74.2±0.9	60.2±1.3	65.9±0.8	84.8±0.8	62.7±1.0	79.8 GB
SemFed-Homogeneous	67.3±0.8	78.4±0.7	64.6±1.0	70.1±0.6	84.9±0.7	65.6±0.8	8.43 MB
SemFed-Mixed	66.8±0.9	77.6±0.8	64.1±0.9	69.5±0.7	84.6±0.6	65.1±0.7	3.64 MB

Table 4: Per-round communication breakdown for SemFed on Qwen2.5-1.5B-Instruct. The first five rounds are shown as a representative window. The mean per-round cost is 40.1 KB and the cumulative cost over 70 rounds is 2.81 MB. Upload denotes client→server policy summaries while Download denotes server→client aggregated guidance. Shared and Elite denote the number of patterns identified by the server in each round.

Round	Upload	Download	Total	Shared	Elite
1	12.5 KB	26.2 KB	38.7 KB	15	5
2	11.9 KB	26.8 KB	38.7 KB	15	5
3	12.5 KB	31.4 KB	43.9 KB	15	9
4	12.4 KB	26.2 KB	38.6 KB	15	5
5	11.8 KB	29.0 KB	40.8 KB	15	7
Mean / round	12.2 KB	27.9 KB	40.1 KB	—	—
70-round total	0.83 MB	1.98 MB	2.81 MB	—	—

Table 5: Communication-frequency analysis on Qwen2.5-1.5B-Instruct. Every 2 rounds applies semantic guidance only on even-numbered rounds, while odd rounds run local GRPO without guidance exchange.

Frequency	Best SR	Rounds	Comm.	Compress.
Every round	64.8	70	2.81 MB	5.6×10^3
Every 2 rounds	<u>57.7</u>	70	1.41 MB	1.1×10^4

under the same participation and interaction budget.

As shown in Table 3, SemFed-Mixed achieves a macro ALFWorld success rate of 69.5%, outperforming FedAgent-Clustered by 3.6 points, while improving WebShop success rate from 62.7% to 65.1%. Compared with three separate homogeneous SemFed federations, the mixed setting retains 99.1% of the ALFWorld performance and 99.2% of the WebShop success rate. Meanwhile, the total communication is only 3.64 MB, compared with 79.8 GB for FedAgent-Clustered. These results provide direct evidence that SemFed enables heterogeneous LLM backbones to collaborate through a shared semantic space without requiring parameter alignment.

Table 6: Ablation on guidance channels for SemFed on Qwen2.5-1.5B-Instruct. Best Success Rate (%) is reported. Each variant retains only one guidance channel. Isolated refers to the no-communication baseline in which clients train independently with GRPO. The reported score is the average best SR across the participating clients. Δ denotes the absolute improvement over Isolated.

Variant	Best SR	Δ vs. Isolated
Isolated (no guidance)	52.6	—
Elite Only	53.4	+0.8
Anti Only	54.7	+2.1
Shared Only	<u>60.5</u>	+7.9
SemFed (Full)	64.8	+12.2

4.5 Communication Efficiency Analysis

Table 4 reports the per-round upload and download budget of SemFed during training. The mean per-round cost is 40.1 KB and the cumulative cost over 70 rounds reaches 2.81 MB, of which 0.83 MB corresponds to client-to-server uploads and 1.98 MB to server-to-client downloads. As reference, FedAvg with rank-32 LoRA adapters on the 1.5B base model transmits approximately 224 MB per round and 15.7 GB across 70 rounds under the same configuration, corresponding to a compression ratio of 5.6×10^3 . Table 5 examines whether every-round guidance exchange is necessary. Reducing the exchange frequency to once every two rounds yields a best Success Rate of 57.7%, which retains 89.0% of the every-round score (64.8%) while further reducing total communication from 2.81 MB to 1.41 MB. The compression ratio over FedAvg correspondingly increases from 5.6×10^3 to 1.1×10^4 .

4.6 Ablation Study on Guidance Channels

Table 6 ablates the three semantic guidance channels by retaining one channel at a time, with the Isolated variant (52.6%) as the no-guidance reference. Shared patterns alone yield the largest single-channel improvement (+7.9), accounting for 64.8% of the gain achieved by the full sys-

Table 7: Training trajectory of SemFed on Qwen2.5-1.5B-Instruct, reported at ten evenly spaced checkpoints. SR (%) denotes Success Rate, and Comm. denotes cumulative communication up to the checkpoint.

Round	7	14	21	28	35	42	49	56	63	70
SR (%)	42.5	50.8	55.6	58.9	61.0	62.5	63.6	64.2	<u>64.6</u>	64.8
Comm. (MB)	0.28	0.56	0.84	1.12	1.40	1.68	1.97	2.25	2.53	2.81

Table 8: Scalability under different federation sizes on Qwen2.5-1.5B-Instruct. All experiments run with $\tau=0.5$. Comm. denotes the cumulative communication cost.

Setting	Clients	Per Round	SR	Comm.
FedAgent	100	2	61.7	~17.8 GB
Small federation	50	2	62.0	1.40 MB
Default	100	2	64.8	2.81 MB
Large federation	200	2	<u>65.6</u>	5.61 MB

tem. Anti-patterns and Elite patterns contribute marginally on their own (+2.1 and +0.8), since penalty signals do not drive directed exploration and high-performing-client strategies are tightly coupled to local context. The full SemFed reaches 64.8% with a gain of +12.2 over Isolated, which exceeds the linear sum of the three individual channels (+10.8) by 1.4 points.

4.7 Extended Training and Scalability Analysis

Two complementary studies assess long-horizon convergence and scaling behavior. Table 7 reports the training trajectory across ten evenly spaced checkpoints within the 70-round schedule. The Success Rate increases monotonically from 42.5% at round 7 to 64.8% at round 70, with the gain per checkpoint decreasing from +8.3 in the early phase to +0.2 near convergence. Cumulative communication scales linearly with the number of rounds, reaching 2.81 MB at round 70.

Table 8 evaluates the scaling behavior under varying federation sizes. The default configuration attains a Success Rate of 64.8% with a total communication of 2.81 MB. Scaling the federation to $K=200$ raises the Success Rate to 65.6% at the cost of doubling the cumulative communication to 5.61 MB, indicating that additional clients contribute behavioral diversity but with diminishing returns. A larger candidate pool brings more diverse clients into rotation across rounds, which leads the server to accumulate more distinct shared and elite patterns. Reducing the federation to $K=50$ produces a 2.8-point drop, since the pool of distinct behavioral patterns shrinks accordingly.

Table 9: Performance under different heterogeneity levels (Dirichlet τ) on Qwen2.5-1.5B-Instruct. Lower τ produces more skewed client task distributions.

Heterogeneity	τ	Best SR	Avg SR
FedAgent (high)	—	58.2	—
FedAgent (low/IID)	—	62.3	—
High	0.1	62.7	44.6
Moderate	0.5	<u>64.8</u>	<u>47.5</u>
Low (near-IID)	1.0	65.6	48.5

4.8 Robustness to Data Heterogeneity

Table 9 reports performance under varying Dirichlet concentrations τ , with smaller τ corresponding to stronger heterogeneity. SemFed attains a best Success Rate of 65.6% at $\tau=1.0$, 64.8% at $\tau=0.5$, and 62.7% at $\tau=0.1$. The total range from the IID setting to the strongly heterogeneous setting is 2.9 points, which is 1.2 points narrower than the corresponding range of 4.1 points observed for FedAgent over the same span. SemFed also surpasses FedAgent at both ends of the heterogeneity spectrum, by +4.5 points under strong heterogeneity (62.7 vs. 58.2) and by +3.3 points under the near-IID setting (65.6 vs. 62.3). Behavioral patterns are extracted from individual client policies and combined at the semantic level, which does not require gradient or weight alignment across clients and is therefore less affected by distributional skew.

5 Conclusion

This work presents a semantic federated learning framework for heterogeneous LLM agents called SemFed that communicates compact behavioral patterns instead of model parameters. By decoupling federated knowledge transfer from the parameter space, SemFed jointly addresses the model-heterogeneity, communication-overhead and data-heterogeneity challenges that constrain prior methods. Extensive experiments on WebShop and ALFWorld across four LLM backbones show that SemFed consistently surpasses parameter-aggregation baselines in task quality while reducing communication by over three orders of magnitude.

Limitations

SemFed currently focuses on text based interactive environments and relies on benchmark specific action canonicalization, which may limit transfer when client action spaces cannot be aligned semantically. Its performance is also affected by the thresholds used to identify shared, elite, and failure patterns. Moreover, although raw trajectories and model parameters remain local, semantic summaries may still reveal task or behavior information.

Acknowledgments

This work was supported in part by the National Key R&D Program of China (Grant No. 2024YFB2906503); in part by the National Natural Science Foundation of China (Grant No. U25B2031); in part by the 111 Project (Grant No. B21049) and the Key Science and Technology Project of Ministry of Emergency Management of the People's Republic of China (Grant No. 2024EMST080801).

References

- Tianshi Che, Ji Liu, Yang Zhou, Jiaxiang Ren, Jiwen Zhou, Victor Sheng, Huaiyu Dai, and Dejing Dou. 2023. Federated learning of large language models with parameter-efficient prompt tuning and adaptive optimization. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7871–7888.
- Canyu Chen, Kangyu Zhu, Zhaorun Chen, Zhanhui Zhou, Shizhe Diao, Yiping Lu, Tian Li, Manling Li, and Dawn Song. 2026. Federated agent reinforcement learning. In *AAAI Workshop on LLM-based Multi-Agent Systems (LaMAS)*.
- Zehui Chen, Kuikun Liu, Qiuchen Wang, Jiangning Liu, Wenwei Zhang, Kai Chen, and Feng Zhao. 2024. AgentFLAN: Designing data and methods of effective agent tuning for large language models. In *Findings of the Association for Computational Linguistics: ACL*.
- Mingyue Cheng, Jie Ouyang, Shuo Yu, Ruiran Yan, Yucong Luo, Zirui Liu, Daoyu Wang, Qi Liu, and Enhong Chen. 2025. Agent-r1: Training powerful llm agents with end-to-end reinforcement learning. *arXiv e-prints*, pages arXiv–2511.
- Kemboi Cheruiyot, Nickson Kiprotich, Vyacheslav Kungurtsev, Kennedy Mugo, Vivian Mwirigi, and Marvin Ngesa. 2025. A survey of multi agent reinforcement learning: Federated learning and cooperative and noncooperative decentralized regimes. *arXiv preprint arXiv:2507.06278*.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shitong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*.
- Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, and 1 others. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*.
- Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, and 1 others. 2021. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14(1–2):1–210.
- Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. 2020. SCAFFOLD: Stochastic controlled averaging for federated learning. In *Proceedings of the International Conference on Machine Learning (ICML)*.
- Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, and 1 others. 2025. Tulu 3: Pushing frontiers in open language model post-training with reinforcement learning from verifiable rewards. *arXiv preprint arXiv:2411.15124*.
- Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems (MLSys)*.
- Xiang-Yang Liu, Rongxin Zhu, Yuxiang Zhang, and Bingsheng Chen. 2024. Differentially private low-rank adaptation of large language models using federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
- Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Jiaju Qi, Qihao Zhou, Lei Lei, and Kan Zheng. 2023. Federated reinforcement learning: Techniques, applications, and open challenges. *Intelligence & Robotics*, 3(1):18–57.
- Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, and 1 others. 2024. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Yu Li, Yu Wu, and Daya Guo. 2024. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.
- Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2021. ALFWorld: Aligning text and embodied environments for interactive learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Yifan Song, Da Yin, Xiang Yue, Jie Huang, Sujian Li, and Bill Yuchen Lin. 2024. Trial and error: Exploration-based trajectory optimization for LLM agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*.
- Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, and 1 others. 2025. RAGEN: Understanding self-evolution in LLM agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*.
- Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, and 1 others. 2025a. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):1–26.
- Zhiheng Xi, Yiwen Ding, Wenxiang Chen, Boyang Hong, Honglin Guo, Junzhe Wang, Xin Guo, Dingwen Yang, Chenyang Liao, Wei He, and 1 others. 2025b. Agentgym: Evaluating and training large language model-based agents across diverse environments. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 27914–27961.
- Huiqiang Xie, Zhijin Qin, Xiaoming Tao, and Khaled B Letaief. 2023. Task-oriented multi-user semantic communications for cooperative edge inference. *IEEE Journal on Selected Areas in Communications*, 41(9):2711–2725.
- Mengwei Xu, Dongqi Cai, Wangsong Yin, Shangguang Wang, Xin Jin, and Xuanzhe Liu. 2024. FwdLLM: Efficient federated finetuning of large language models with perturbed inferences. In *Proceedings of the USENIX Annual Technical Conference (USENIX ATC)*.
- John Yang, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. 2024. SWE-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Wanting Yang, Zi Qin Liew, Wei Yang Bryan Lim, Zehui Xiong, Dusit Niyato, Xuefen Chi, Xianbin Cao, and Khaled B Letaief. 2022. Semantic communication meets edge intelligence. *IEEE wireless communications*, 29(5):28–35.
- Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *Proceedings of the International Conference on Learning Representations (ICLR)*.
- Jianyi Zhang, Saeed Vahidian, Martin Kuo, Chunyuan Li, Ruiyi Zhang, Tong Yu, Yufan Zhou, Guoyin Wang, and Yiran Chen. 2024. Towards building the federated GPT: Federated instruction tuning. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.
- Zhuo Zhang, Yuanhang Yang, Yong Dai, Qifan Wang, Yue Yu, Lizhen Qu, and Zenglin Xu. 2023. When federated learning meets pre-trained language models’ parameter-efficient tuning methods. In *Findings of the Association for Computational Linguistics: ACL*.
- Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. 2024. ArCher: Training language model agents via hierarchical multi-turn RL. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Algorithm 1 Structured Semantic Policy Extraction

Require: Local dataset $\mathcal{D}_k$, Current policy π_{θ_k} , Reference policy π_{ref} , Standardization mapping ϕ , Failure threshold λ , Group size G

Ensure: Policy summary $\mathcal{P}_k^{(t)}$

```
1: Initialize local trajectory buffer  $\mathcal{B}_k \leftarrow \emptyset$ 
2: for each task instruction  $x \sim \mathcal{D}_k$  do
3:   Sample responses  $\{y_1, \dots, y_G\} \sim \pi_{\theta_k}(\cdot | x)$ 
4:   Compute rewards  $r_i = R(x, y_i)$ ,  $i = 1, \dots, G$ 
5:    $\hat{A}_i \leftarrow \frac{r_i - \text{mean}(\{r_j\})}{\text{std}(\{r_j\}) + \epsilon}$ 
6:   Update  $\pi_{\theta_k}$  via  $\mathcal{L}^{(k)}(\theta)$  with KL reg.
7:   for each trajectory  $\tau_j$  during training do
8:      $\tilde{\tau}_j^{(k)} \leftarrow (\tau_j^{(k)}, R_j^{(k)}, s_j^{(k)}, g_j^{(k)})$ 
9:      $\mathcal{B}_k \leftarrow \mathcal{B}_k \cup \{\tilde{\tau}_j^{(k)}\}$ 
10:  end for
11: end for
12: for each  $\tilde{\tau} \in \mathcal{B}_k$ , each action  $a_t$  in  $\tilde{\tau}$  do
13:    $\alpha_t \leftarrow \phi(a_t)$ 
14: end for
15: Compute  $p_{\text{pref}} = (\alpha, f_\alpha, \bar{r}_\alpha, c_\alpha) \forall \alpha \in \mathcal{A}_{\text{type}}$ 
16: Extract  $p_{\text{succ}} = (\sigma, f_\sigma, \bar{r}_\sigma, \kappa_\sigma)$  from  $s_j^{(k)} = 1$ 
17: for each action category  $\alpha$  do
18:   if  $f_\alpha^{\text{fail}, (k)} > \lambda \cdot f_\alpha^{\text{succ}, (k)}$  then
19:     Add  $\alpha$  to failure patterns  $p_{\text{fail}}$ 
20:   end if
21: end for
22:  $\mathcal{P}_k^{(t)} \leftarrow (\bar{R}_k, \eta_k, \Delta_k, \mathbf{p}_k, \mathbf{d}_k)$ 
23: return Serialize( $\mathcal{P}_k^{(t)}$ )
```

A Algorithm Details

This appendix provides the detailed pseudocode for the three core components of SemFed introduced in the main text, together with the overall procedure that integrates them into a unified federated training loop. For readability, each algorithm is preceded by a short description that clarifies its position in the pipeline, its main steps, and the object it ultimately produces.

A.1 Structured Semantic Policy Extraction

Algorithm 1 formalizes the client-side procedure of SSPE. It interleaves group-relative policy optimization with structured trajectory bookkeeping, and converts raw action sequences into preference, success and failure patterns through the unified action standardization mapping ϕ . The output is a

Algorithm 2 Cross-Client Semantic Knowledge Aggregation

Require: Policy summaries $\{\mathcal{P}_k^{(t)}\}_{k \in \mathcal{U}^t}$, Thresholds $\delta_{\text{shared}}, \delta_{\text{elite}}$

Ensure: Global guidance $\mathcal{G}^{(t)}$

```
1:  $K_t \leftarrow |\mathcal{U}^t|$ ;  $\mathcal{M} \leftarrow \{\}$ 
2: for each client  $k \in \mathcal{U}^t$ , each  $p \in \mathbf{p}_k$  do
3:    $\kappa \leftarrow \psi(p) = \text{type}(p) \oplus \text{seq}(\text{actions}(p))$ 
4:    $\mathcal{M}[\kappa] \leftarrow \mathcal{M}[\kappa] \cup \{(k, p)\}$ 
5: end for
6:  $\mathbf{P}_{\text{shared}}^{(t)} \leftarrow \emptyset$ 
7: for each  $\kappa \in \mathcal{M}$  do
8:    $C_\kappa \leftarrow \{k : \exists p \in \mathbf{p}_k, \psi(p) = \kappa\}$ 
9:   if  $|C_\kappa|/K_t \geq \delta_{\text{shared}}$  then
10:     $\mathbf{P}_{\text{shared}}^{(t)} \leftarrow \mathbf{P}_{\text{shared}}^{(t)} \cup \{\kappa\}$ 
11:   end if
12: end for
13:  $\bar{R}_{\text{global}}^{(t)} \leftarrow \frac{1}{K_t} \sum_k \bar{R}_k$ 
14:  $\mathcal{E} \leftarrow \{k : \bar{R}_k > \bar{R}_{\text{global}}^{(t)}\}$ ;  $\mathbf{P}_{\text{elite}}^{(t)} \leftarrow \emptyset$ 
15: for each pattern  $p$  in elite clients  $\mathcal{E}$  do
16:    $\text{ES}(p) \leftarrow f_p^{\text{elite}} / (f_p^{\text{non-elite}} + \epsilon)$ 
17:   if  $\text{ES}(p) > \delta_{\text{elite}}$  then
18:     $\mathbf{P}_{\text{elite}}^{(t)} \leftarrow \mathbf{P}_{\text{elite}}^{(t)} \cup \{p\}$ 
19:   end if
20: end for
21:  $\mathbf{P}_{\text{anti}}^{(t)} \leftarrow \emptyset$ 
22: for each failure pattern  $p$  across all clients do
23:    $\text{Severity}(p) \leftarrow f_p^{\text{fail}} \cdot \kappa_p$ 
24: end for
25: Deduplicate, rank by severity  $\rightarrow \mathbf{P}_{\text{anti}}^{(t)}$ 
26: Calculate  $\mathcal{G}^{(t)}$ 
27: return  $\mathcal{G}^{(t)}$ 
```

compact policy summary $\mathcal{P}_k^{(t)}$ that serves as the sole payload uploaded to the server in each communication round.

A.2 Cross-Client Semantic Knowledge Aggregation

Algorithm 2 describes the server-side aggregation procedure of CSKA. The server first hashes individual patterns into semantic equivalence classes, and then distills three complementary categories of global guidance, namely shared patterns that recur across a majority of clients, elite patterns disproportionately frequent among high-reward clients, and severity-ranked anti-patterns associated with frequent failures. The whole procedure operates purely on structured semantic descriptors so that no raw trajectories or model weights are ever ex-

Algorithm 3 Knowledge-Guided Training Signal Generation

Require: Global guidance $\mathcal{G}^{(t)}$, Local buffer $\mathcal{B}_k$, Policy π_{θ_k} , Weights w_s, w_e, w_a , Bonuses $\Delta r^{\text{shared}}, \Delta r^{\text{elite}}, \Delta r^{\text{anti}}$

Ensure: Augmented training signals for local RL

- 1: *{Prompt Augmentation}*
- 2: Generate $I^{(t)}$ from $\mathbf{p}_{\text{shared}}^{(t)}, \mathbf{p}_{\text{elite}}^{(t)}, \mathbf{p}_{\text{anti}}^{(t)}$
- 3: **for** each step with observation o and history h **do**
- 4: $\pi_{\theta_k}^{(t+1)}(a \mid o, h) \leftarrow \pi_{\theta_k}(a \mid [I^{(t)}; o], h)$
- 5: **end for**
- 6: *{Reward Shaping}*
- 7: **for** each trajectory $\tau \in \mathcal{B}_k$ **do**
- 8: $\tilde{R}(\tau) \leftarrow R(\tau)$
- 9: **for** each $p \in \mathbf{p}_{\text{shared}}^{(t)}$ **do**
- 10: **if** $\tau \models p$ **then**
- 11: $\tilde{R}(\tau) += \Delta r_p^{\text{shared}}$
- 12: **end if**
- 13: **end for**
- 14: **for** each $p \in \mathbf{p}_{\text{elite}}^{(t)}$ **do**
- 15: **if** $\tau \models p$ **then**
- 16: $\tilde{R}(\tau) += \Delta r_p^{\text{elite}}$
- 17: **end if**
- 18: **end for**
- 19: **for** each $p \in \mathbf{p}_{\text{anti}}^{(t)}$ **do**
- 20: **if** $\tau \models p$ **then**
- 21: $\tilde{R}(\tau) += \Delta r_p^{\text{anti}} \{\text{penalty}\}$
- 22: **end if**
- 23: **end for**
- 24: **end for**
- 25: *{Trajectory Relevance Scoring}*
- 26: **for** each trajectory $\tau \in \mathcal{B}_k$ **do**
- 27: Calculate $\text{Rel}(\tau)$
- 28: **end for**
- 29: **return** $\{\tilde{R}(\tau)\}, \{\text{Rel}(\tau)\}, I^{(t)}$

changed.

A.3 Knowledge-Guided Training Signal Generation

Algorithm 3 translates the global guidance $\mathcal{G}^{(t)}$ broadcast from the server into concrete signals that shape the next round of local reinforcement learning. The aggregated knowledge is injected through three complementary channels, namely prompt augmentation at the input side, reward shaping on the trajectory return, and trajectory relevance scoring over the local buffer. These channels jointly bias optimization towards shared and elite patterns

Algorithm 4 Overall Procedure of SemFed Framework

Require: K clients $\{C_1, \dots, C_K\}$ with datasets $\{\mathcal{D}_k\}$, Mapping ϕ , Hyperparams $(\delta^{\text{shared}}, \delta^{\text{elite}}, \lambda, G, T)$

Ensure: Trained policies $\{\pi_{\theta_k}\}_{k=1}^K$

- 1: Initialize π_{θ_k} and $\bar{R}_k^{(0)} \leftarrow 0$ for all k
- 2: **for** round $t = 1, 2, \dots, T$ **do**
- 3: Server selects $\mathcal{U}^t \subseteq \{C_1, \dots, C_K\}$
- 4: **if** $t > 1$ **then**
- 5: Broadcast $\mathcal{G}^{(t-1)}$ to $\mathcal{U}^t$
- 6: **end if**
- 7: **for** each $C_k \in \mathcal{U}^t$ **in parallel do**
- 8: **if** $t > 1$ **then**
- 9: **Algorithm 3:** generate signals from $\mathcal{G}^{(t-1)}$
- 10: **end if**
- 11: **Algorithm 1:** local GRPO $\rightarrow \mathcal{P}_k^{(t)}$
- 12: Upload $\text{Serialize}(\mathcal{P}_k^{(t)})$ to server
- 13: **end for**
- 14: **Algorithm 2:** aggregate $\rightarrow \mathcal{G}^{(t)}$
- 15: **end for**
- 16: **return** $\{\pi_{\theta_k}\}_{k=1}^K$

while explicitly penalizing trajectories that match anti-patterns.

A.4 Overall Procedure

Algorithm 4 integrates SSPE, CSKA and KTSG into the end-to-end SemFed training loop. Each round alternates between server-side broadcasting of the previous global guidance, client-side guided GRPO that produces local policy summaries, and server-side semantic aggregation that yields the next-round guidance. The loop is iterated for T communication rounds and returns the final set of personalized per-client policies.

A.5 Semantic Processing Implementation

SemFed uses deterministic benchmark-specific parsers to map free-form actions into predefined canonical action vocabularies, with unmatched actions assigned to an other category. Pattern confidence is computed from the relative occurrence of each pattern in successful and failed trajectories. After canonicalization, patterns are grouped by pattern type and ordered action sequence, and their frequency, reward, and confidence statistics are aggregated within each group.

Multi-action strategies are matched only when

Table 10: Detailed experimental protocol.

Configuration	ALFWorld	WebShop
Environment	Text-only v0.3	Official simulator
Training instructions	1,200	8,000
Validation instructions	200	500
Held-out test instructions	300	1,000
Evaluation episodes / seed	300	1,000
Maximum interaction steps	50	30
Client-task allocation	Dirichlet 0.5	Dirichlet 0.5
Rollouts / instruction	4	4
Local epochs / round	1	1
Training rounds	70	70
Random seeds	42, 123, 2026	42, 123, 2026
Checkpoint	Best validation	Best validation

the corresponding ordered sequence occurs contiguously in a trajectory, whereas single-action preferences and anti-patterns are matched according to the relevant action and state condition. The server selects the highest-ranked shared, elite, and failure patterns to construct the guidance prompt, with anti-pattern constraints taking priority when conflicting recommendations occur.

B Experimental Configuration Details

This section reports the default hyperparameters and hardware environment used throughout the main experiments. Unless explicitly varied in an ablation, all studies in the main text follow the configuration described below.

B.1 Federated Protocol

We instantiate a federation of $K=100$ clients and uniformly sample $M=2$ clients per communication round. Training proceeds for $T=70$ rounds in total. Task heterogeneity across clients is controlled by a Dirichlet distribution with concentration parameter $\tau=0.5$, which corresponds to a moderately skewed non-IID setting; the IID and strongly-heterogeneous regimes ($\tau \in \{1.0, 0.1\}$) are only used in the robustness analysis.

Table 10 summarizes the detailed dataset splits and training/evaluation protocol used for ALFWorld and WebShop.

B.2 Local Optimization

The default LLM backbone is Qwen2.5-1.5B-Instruct, with additional results reported on Qwen2.5-3B/7B-Instruct and Llama-3.2-3B-Instruct. All backbones are fine-tuned with LoRA at rank $r=32$ and scaling factor $\alpha=64$, attached to the query and value projections. Local policy optimization uses GRPO with group size $G=8$,

Table 11: Ablation on pattern-key construction.

Pattern Key	ALFWorld SR	WebShop Score	WebShop SR
sort(actions)	65.9 $\pm$ 0.9	84.3 $\pm$ 0.8	62.7 $\pm$ 0.9
seq(actions)	67.3$\pm$0.8	85.7$\pm$0.7	64.8$\pm$0.8

Table 12: Sensitivity to semantic-pattern selection thresholds.

Setting	δ_{shared}	Elite Thres.	λ	ALFWorld SR	WebShop SR
Conservative	0.6	2.0	2.0	66.5 $\pm$ 0.9	64.0 $\pm$ 0.9
Default	0.5	1.5	1.5	67.3$\pm$0.8	64.8$\pm$0.8
Permissive	0.3	1.2	1.2	65.8 $\pm$ 1.0	63.6 $\pm$ 1.0

KL regularization weight $\beta=0.04$, and clipping threshold $\varepsilon=0.2$. The learning rate is fixed at 1×10^{-5} with a cosine schedule, the per-client batch size is 16, the maximum response length is 512 tokens, and each round performs one local epoch over the sampled task instructions.

B.3 Hyperparameters

For SSPE, the failure-pattern threshold is $\lambda=1.5$, meaning that an action category is reported as a failure pattern when its failure frequency exceeds $1.5 \times$ its success frequency. For CSKA, the shared-pattern frequency threshold is $\delta_{\text{shared}}=0.5$ and the elite-pattern enrichment threshold is $\delta_{\text{elite}}=1.5$. For KTSG, the relevance weights are set to $w_s=1.0$, $w_e=1.2$ and $w_a=0.8$, and the reward-shaping bonuses are $\Delta r^{\text{shared}}=+0.1$, $\Delta r^{\text{elite}}=+0.2$ and $\Delta r^{\text{anti}}=-0.15$. These values are kept fixed across all backbones and benchmarks unless otherwise specified.

B.4 Ablation

We examine whether preserving temporal action order is necessary when constructing semantic pattern keys. As shown in Table 11, replacing the set-based key `sort(actions)` with the order-preserving `seq(actions)` improves performance consistently on both benchmarks.

B.5 Threshold Sensitivity

We further evaluate the sensitivity of SemFed to the thresholds used for shared-pattern selection, elite-pattern identification, and failure-pattern detection. As shown in Table 12, the default configuration achieves the best performance on both benchmarks.

B.6 Hardware and Software Environment

Experiments run on a workstation equipped with an Intel Core Ultra 7 265K CPU, 96 GB of RAM and an NVIDIA RTX PRO 6000 GPU (96GB), under

Table 13: Paired bootstrap significance tests between SemFed and FedAgent. Δ denotes SemFed minus FedAgent.

Backbone	Metric	Δ	95% CI	p
Qwen2.5-1.5B	ALFWorld All SR	+3.2	[1.1, 5.3]	0.004
Qwen2.5-1.5B	WebShop Score	+2.5	[0.8, 4.2]	0.006
Qwen2.5-1.5B	WebShop SR	+3.1	[1.0, 5.2]	0.005
Qwen2.5-3B	ALFWorld All SR	+3.6	[1.4, 5.8]	0.003
Qwen2.5-7B	ALFWorld All SR	+2.9	[1.0, 4.8]	0.005
Llama-3.2-3B	ALFWorld All SR	+3.4	[1.2, 5.6]	0.004

Ubuntu 22.04 LTS. The implementation is built on PyTorch with the HuggingFace Transformers and PEFT libraries, while ALFWorld and WebShop are instantiated through their official Python interfaces.

C Statistical Significance

We conduct paired bootstrap evaluation over the same held-out episodes using 10,000 bootstrap samples. As shown in Table 13, SemFed consistently achieves statistically significant improvements over FedAgent across all evaluated backbones and metrics, with all confidence intervals remaining above zero and $p < 0.01$.