Computers and Artificial Intelligence; Vol.3, No.6; 2026
E-ISSN: 3006-3728 P-ISSN: 3006-371X
DOI: https://doi.org/10.70267/cai.v3n6.6177
Published by: Zeus Press

A Study on the Coupling Relationship Between AI Personalization and Long-Context Management

Xinghe Fan*

School of Information Engineering, North China University of Water Resources and Electric Power Xinyang 464200, China

**Corresponding author: Xinghe Fan.*

Abstract

With the large-scale deployment of large language models (LLMs) in various intelligent assistants and agent systems, homogenized outputs and the loss of user preferences across sessions have become key bottlenecks restricting improvements in user experience. Personalization technology aims to adapt LLMs to the language habits, thinking patterns, and subjective preferences of different users, while long-context and agent memory management addresses the underlying technical problems of storing, filtering, and reusing temporal information for LLMs. Current academic research mostly treats the two directions as independent fields, and systematic reviews of their internal coupling relationship are relatively scarce. This study first defines the core concepts of LLM personalization and long-context memory management and systematically reviews the technical principles, advantages, and existing limitations of four personalization paradigms: prompt-engineering personalization, lightweight parameter fine-tuning, user-preference alignment, and dynamic user profiling. It then divides long-context and agent-memory technologies into five routes: native-window expansion, context pruning and compression, adaptive context extraction, hierarchical memory architectures, and external retrieval-augmented memory, and analyzes the applicable scenarios and technical shortcomings of each route. On this basis, this study reviews progress in four intersecting directions: preference-aware memory updating, structured persona-memory representation, long-horizon personalization evaluation benchmarks, and training-free collaborative control at inference time. It further analyzes the internal coupling mechanism through which the two directions empower each other and summarizes three core conflicts: the capacity-accuracy trade-off, the conflict between general governance rules and personalized characteristics, and the conflict between computational cost and personalization stability. The study also summarizes five research challenges: difficulty in adapting to the dynamic evolution of user preferences, failure to recall implicit preferences, memory errors and preference drift, fragmented evaluation systems, and difficulty balancing engineering implementation costs. It proposes subsequent research paths in six dimensions: cognitively inspired memory architectures, preference-aware context compression algorithms, multimode hybrid indexing and retrieval, unified long-horizon personalization evaluation protocols, memory-error verification mechanisms, and multimodal long-term personalization. The findings indicate that long-term personalized intelligent systems cannot optimize the personalization module or memory module in isolation. Their coupling relationship should serve as the underlying design logic to promote the iterative upgrading of LLMs toward adaptation to individual user characteristics and to provide a reference for the theoretical research and engineering development of long-term agents and personalized dialogue assistants.

Keywords

large language model, personalization, long context, agent memory, user preference, coupling relationship

© 2026 The Author(s). Authors retain copyright in their articles.

1. Introduction

Since 2022, large language model (LLM) technology has developed rapidly. Model parameter counts have continued to increase, basic capabilities have steadily improved, and strong general-intelligence performance has been demonstrated in tasks such as question answering, text generation, and logical reasoning [1-4]. However, the underlying training logic of general-purpose LLMs is uniformly optimized for all Internet users. Model outputs therefore tend to be generalized and averaged, making it difficult to match each user's unique usage habits, knowledge background, thinking patterns, and value orientations. When different users submit exactly the same prompt to the same foundation model, the returned answers are highly similar. This phenomenon, known as artificial intelligence (AI) output homogenization, reduces user engagement and limits the practical value of intelligent assistants [2]. Therefore, LLM personalization has become an important research direction in artificial intelligence. Researchers seek to use various technical means to enable LLMs to identify, remember, and follow individual user preferences when generating answers, thereby achieving a one-user-one-model interaction experience.

Early research on LLM personalization was mostly limited to single-session interaction scenarios. Such methods can adapt to user preferences only within a single conversation. Once the session ends and the context window is cleared, the model forgets previously accumulated user habits and historical interaction information. When a new conversation begins, all personalization effects disappear, leading to typical problems such as persona reset and preference forgetting [5-8]. To address the loss of preferences across sessions, researchers began to introduce long-context technology and long-term agent memory systems. These approaches overcome the storage limit of the model's native token window and retain multi-turn historical interactions across different dates, thereby providing an information carrier for long-term personalization. However, merely expanding the context window does not directly produce ideal personalization. Empirical studies have shown that, as the context length continues to expand, large amounts of interaction text cause an attention-dilution effect. Important user-preference information is submerged in substantial amounts of irrelevant conversation, making it difficult for the model to locate key user characteristics accurately and instead reducing the quality of personalized responses [9-12]. This phenomenon indicates that personalization and long-context management do not have a simple additive relationship. A complex coupling mechanism of mutual constraint and mutual reinforcement exists between them.

Some reviews in China and other countries have separately summarized LLM personalization technology, long-context technology, and agent memory systems. Some reviews focus on horizontal comparisons among personalization technologies and analyze the advantages and disadvantages of prompts, fine-tuning, alignment, and other methods. Other reviews specifically investigate technical progress in long-context extension, memory retrieval, memory updating, and forgetting [2-3, 9]. However, most existing reviews regard the personalization module and memory module as two independent systems and rarely analyze how they affect each other from the perspective of coupled interaction. Many existing studies conduct only one-way technical validation: some study how to improve personalization algorithms while assuming that the memory system is already a mature and fixed component, whereas others optimize memory and long-context algorithms while assuming that user preferences can be extracted simply and ignoring the constraints imposed by preferences on memory filtering, forgetting, and retrieval. The lack of an integrated analytical framework explaining the internal relationship between the two subsystems has also fragmented research in this direction and made research conclusions difficult to compare with one another.

Based on this research status, this study limits its survey to academic literature in the arXiv preprint database from 2021 to 2026 concerning LLM personalization, user-profile modeling, preference alignment, long-context optimization, long-term agent memory, and long-horizon personalization evaluation benchmarks. Literature on traditional Internet recommendation systems, unimodal computer-vision tasks, and embodied robotic agents is excluded. Original research papers are given priority and are supplemented by review papers. A total of 60 relevant preprints were selected as the analytical basis for this review. The core contribution of this study is not a new algorithm but a coupling-relationship analytical framework that connects the relatively separate research branches of personalization and long-context memory, clarifies their bidirectional mechanisms, core conflicts, existing research difficulties, and future research directions, addresses the missing

© 2026 The Author(s). Authors retain copyright in their articles.

perspective in existing reviews, and provides a theoretical reference for the design of long-term personalized agents.

2. Definition of Related Concepts

2.1 Personalization of Large Language Models

LLM personalization refers to using various technical means to modify or constrain the output behavior of an LLM so that its responses adapt to the unique characteristics of a specific user. User characteristics can be divided into explicit preferences and implicit preferences. Explicit preferences are habits that users directly express in language, such as telling an AI system, "Keep the answer as brief as possible and avoid excessive background." Implicit preferences are not described directly in words. They are embedded in the writing style, question tendencies, and value choices found in extensive historical conversations and must be inferred indirectly by the model from large numbers of interaction records [13].

According to the position of technical intervention, personalization technologies can be divided into three categories. The first is inference-stage intervention, which does not change the model's internal weights and relies on external prompts, retrieval results, and decoding strategies to achieve personalized adaptation. The second is parameter fine-tuning intervention, which uses parameter-efficient fine-tuning methods such as low-rank adaptation (LoRA) to modify some model weights and encode user-preference information into the model. The third is preference-alignment intervention, which relies on alignment paradigms such as direct preference optimization (DPO) and reinforcement learning from human feedback (RLHF) to optimize the model's preference distribution using user-feedback data [14]. Dynamic user-profile modeling is an intermediate component. It does not directly control model output but extracts and maintains a structured set of user characteristics from historical interaction data and provides user-side input information for upper-level personalization algorithms [15].

Personalization can also be evaluated at several levels. Evaluation considers not only whether response content matches the user's subjective preferences but also personalization consistency, namely whether preferences remain stable across multiple sessions; adaptation to preference evolution, namely whether the model can update when the user's views change; cold-start performance, namely whether basic user characteristics can be captured quickly when interaction data are scarce; and resistance to drift, namely whether the persona deviates after long-term interaction [6]. Many early studies focused only on response matching within a single session and ignored long-term indicators such as consistency and adaptation to evolution. Consequently, algorithms performed well in short-term experiments but deteriorated rapidly after deployment in long-term interaction scenarios [5].

2.2 Long Context and Agent Memory Management

Long-context technology was originally intended to address the limited length of a single text input to an LLM. The attention mechanism of a foundation LLM has a native token-window limit, and text exceeding this length cannot be sent to the model in full for inference [16]. Long-context technology follows two approaches. The first modifies the model's internal attention module and directly expands the maximum length that the native window can process. The second does not modify the model itself but uses an external storage module, namely an agent memory system, to save historical interactions that exceed the window limit in an external database. During inference, only some relevant historical content is selected and inserted into the prompt, thereby indirectly reusing very long temporal histories [17].

An agent memory system generally contains four basic subfunctions: memory writing, which receives a new round of dialogue and saves the interaction in the storage system; memory retrieval, which filters relevant historical records from the history after receiving a new user query; memory compression and summarization, which condenses long historical conversations to reduce token consumption; and memory forgetting, which identifies outdated or invalid historical information and deletes it or reduces its weight [18]. These four subfunctions jointly determine the overall performance of the memory system. Traditional long-context and agent-memory systems are mostly designed for document question answering and general tasks. Their core evaluation indicators focus on general text question-answering accuracy, retrieval recall, and token cost, and they do not include protection mechanisms designed specifically for user-preference information. This is the

© 2026 The Author(s). Authors retain copyright in their articles.

fundamental reason why various problems arise when general memory systems are directly applied to personalized interaction scenarios [19].

2.3 Definition of the Coupling Relationship

The coupling relationship in this study refers to the bidirectional relationship of mutual dependence and mutual constraint between the personalization module and the long-context memory-management module. On the one hand, the long-context memory system is the underlying carrier of long-term personalization. Without cross-session memory storage, all personalization effects are limited to a single session and cannot support stable, long-term user adaptation [20]. On the other hand, user-preference information becomes a decision signal for operations within the memory system. Ranking scores during memory retrieval, the fragments retained during memory compression, and the historical content deleted during memory forgetting are all affected by user-preference characteristics. A change in either subsystem is transmitted to the other and ultimately affects the overall personalized interaction. Therefore, the two systems cannot be optimized separately as completely independent modules. Optimizing only one module while ignoring their mutual constraints produces various performance bottlenecks [21].

3. Current Research on Technologies in the Separate Fields

3.1 Mainstream Technical Paradigms for AI Personalization

3.1.1 Personalization Based on Prompt Engineering

Prompt-engineering personalization is a parameter-free inference-side method and is also one of the most widely used personalization methods in current industrial products. Its core idea is to organize a user's habits and background information into a text placed in the system prompt. Before each round of dialogue inference, the user-profile text and the user's query are concatenated and sent to the LLM input to constrain model output [22]. MemPrompt is an early representative work in this direction. The system constructs an external memory dictionary to store historical user-feedback information and reads the user information in the dictionary during each round of dialogue to augment the prompt, thereby achieving within-session personalization [23].

Prompt-engineering personalization has clear advantages: it requires no model-training process, is simple and fast to deploy, has low development costs, and allows rapid iterative modification of user-profile text. However, its shortcomings are also prominent. First, traditional prompt-engineering personalization is closely bound to the context of a single session. Once the session is closed, the user-profile information stored in the prompt disappears. Without external memory storage, all personalization effects are lost across sessions [24]. Second, as user interactions accumulate, the user-profile text continues to grow. Profile descriptions consume many tokens and reduce the input budget available for the question and answer itself, causing context congestion and increasing inference costs [25]. To address excessive token use, later studies introduced user-profile summarization and compression, condensing long user histories into short profile text and reducing the token cost of each prompt [26]. However, text-summarization algorithms inherently lose information. Compression can easily discard implicit user preferences that are difficult to describe briefly and retain only a few explicit habits, resulting in a partial profile [27]. In general, prompt engineering alone is suitable only for short-term session personalization. To achieve long-term personalization across sessions, it must be used with an external memory-storage system [14].

3.1.2 Personalization Through Lightweight Parameter Fine-Tuning

Lightweight parameter fine-tuning includes LoRA, adapters, and prefix tuning as representative technical routes. These methods do not update all LLM parameters. They select only a small set of auxiliary parameters within the model for training, use the user's historical interactions as the training dataset, and produce user-specific adaptation weights after training [28-30]. Compared with prompt engineering, lightweight fine-tuning can encode some user-preference information in the model parameters. A long user-profile text therefore no longer needs to be written into every prompt, which reduces the token cost of a single dialogue turn [14].

However, the lightweight fine-tuning paradigm has several inherent shortcomings in long-term interaction scenarios. First, after a user's views or usage habits change, trained LoRA weights cannot quickly synchronize the updated user preferences. Adapting to new habits requires the training process to be run again, which adds

© 2026 The Author(s). Authors retain copyright in their articles.

training time and computational cost and makes real-time dynamic adaptation to user preferences difficult [31-32]. Second, the number of effective historical interaction samples provided by one user is usually limited. A small user dataset can easily cause overfitting, in which the model conforms excessively to a small number of historical dialogues, loses generalization ability, and shows a marked decline in response quality when encountering new types of questions [32]. Third, as the number of users increases substantially, the system must save an independent fine-tuning weight file for each user. Large numbers of user-weight files create substantial storage pressure and hinder large-scale deployment for ordinary users [33]. In terms of applicable scenarios, lightweight fine-tuning is more suitable when user preferences remain stable over time and interaction samples are sufficient. In daily dialogue services for ordinary Internet users, preferences change frequently, which clearly limits practical application of this method [34].

3.1.3 Personalization Through User-Preference Alignment

Preference-alignment technologies are represented by RLHF and DPO. Their core idea is to optimize the model using user-feedback samples and adjust the model's output distribution so that responses better conform to the user's value orientation [35-36]. General-domain alignment is usually conducted for all users and seeks outputs that conform to common public values. Personalized preference alignment moves the alignment process to the individual-user level and optimizes alignment using each user's feedback data [37]. User-specific DPO is a typical approach in this direction. It collects one user's positive and negative feedback on different model responses, constructs a user-specific preference dataset, and completes alignment training through the DPO loss function [38].

Personalized alignment can change the model's output tendencies at a deeper level and produces stronger personalization than surface-level prompt constraints, but it also faces practical obstacles. High-quality preference-feedback samples from individual users are scarce, making it difficult to construct high-quality personal-preference datasets. Most ordinary users are unwilling to label individual AI responses as good or bad, and the insufficient dataset size directly limits the final effect of alignment training [39]. Alignment also has a time-validity problem. Users' subjective views change over time, so past feedback no longer represents current views and old samples can misguide the model. Existing studies have shown that, as dialogue time increases, early alignment effects decay naturally and the model gradually returns to generalized output [40]. Training-free inference-time alignment has emerged in recent years as an improvement. It no longer performs separate training for each user. Instead, it introduces preference-guidance strategies at the decoding stage and uses external user-profile information to intervene in sampling. This approach avoids the cost of repeated training, but its degree of personalization is lower than that of training-based alignment [41-42].

3.1.4 Dynamic User-Profile Modeling

Dynamic user-profile modeling is an intermediate supporting technology and does not directly intervene in model output. Its task is to continuously extract and update user characteristics from the historical interaction stream and output a structured user description for upper-level modules such as prompt engineering, fine-tuning, and alignment [43]. A static user profile generates user characteristics only once and is not subsequently modified, so it cannot adjust as the user changes. A dynamic profile is a continuously iterative mechanism. After each new user dialogue, the system evaluates the existing profile, adds the user's latest habitual characteristics, modifies changed understandings, and marks outdated preference records [44].

Dynamic user profiles face several typical difficulties in actual operation. The first is identifying outdated preferences. The system must distinguish habits that remain valid from views that have changed. If outdated information is not identified promptly, old preference content remains in the profile for a long time and misguides later interactions [45]. The second is profile distortion caused by context truncation. Many historical dialogues are discarded directly because of window limits, and some key interaction records that could be used to infer user characteristics are lost, ultimately causing deviations in the profile [46]. In addition, the quality of interaction data varies greatly among users. Some users have scattered dialogues with very low effective information density, which further increases the difficulty of profile extraction [47]. At present, dynamic user profiles are used mainly as supporting components. Their performance directly determines the upper limit of various upper-level personalization technologies [13].

© 2026 The Author(s). Authors retain copyright in their articles.

3.2 Mainstream Technical Paradigms for Long Context and Agent Memory

3.2.1 Native Model-Window Expansion

Native-window expansion directly modifies the internal attention module of an LLM and adjusts attention computation so that the model can read longer input sequences at one time. Longformer and BigBird are early representative models in this direction. They use sparse-attention designs to reduce the computational cost of long-sequence inference [48-49]. Later studies further optimized sparse-attention implementations and continuously extended the upper limit of native model context [50]. The main advantage of native-window expansion is that no external database is required. All historical text is sent directly to the model's attention mechanism, so retrieval omissions do not occur [16].

However, the shortcomings of native-window expansion cannot be ignored. As the number of tokens increases, inference computation still rises substantially. Even with sparse attention, the cost of very-long-sequence inference remains high. More importantly, attention is diluted. When the input sequence reaches millions of tokens, the model's ability to capture scattered key information in the text decreases markedly, and user-preference information embedded in extensive conversational text becomes difficult for the model to notice [51-52]. Simply expanding the native window without limit does not automatically solve long-term personalization. Large amounts of historical interaction instead interfere with the model's judgment of key user characteristics [10]. Because of the dual constraints of computation and attention dilution, native-window expansion is generally used together with other memory methods and is rarely used alone to process user dialogue histories spanning several months [9].

3.2.2 Context Pruning and Compression

The core purpose of context pruning and compression is to control the total input-sequence length, retain high-value information under a limited token budget, and remove redundant and invalid text. Pruning techniques generally use importance scores to assess each segment in the context and directly delete segments with lower scores. Summarization and compression techniques retain the core semantics of the text while rewriting and shortening the original text to reduce token use [53]. Sliding-window summarization and hybrid vector-text compression are two current mainstream approaches [54-55].

General pruning and compression algorithms were developed for document question-answering tasks. Their scoring mechanisms prioritize factual information and do not protect user-preference information. This is their most prominent defect in personalization scenarios. Many implicit user preferences are distributed across multiple dialogues. Each text segment has a low importance score when considered separately and can be deleted easily by general pruning, causing irreversible loss of user-preference information [19]. Related empirical studies have confirmed that directly applying general context-pruning tools to user dialogue histories has a clear negative effect on long-term personalization [12]. Identifying and protecting user-preference information during compression and pruning has therefore become an urgent practical problem [20].

3.2.3 Adaptive Context Extraction

Adaptive context extraction does not read the entire history at once. When a new user query is received, segments semantically related to the current query are extracted from all historical records and sent to the model input. This method does not occupy a fixed number of tokens. The historical records retrieved for each dialogue change dynamically with the user's current question [18]. Semantic-similarity retrieval is the most commonly used implementation. It calculates the semantic similarity between historical records and the current query through vector embeddings and selects the highest-ranked historical segments [17].

Semantic-similarity retrieval has an inherent shortcoming: semantic matching can capture only historical content directly related to the current query. Many user preferences have low semantic similarity to the current query but still constrain how the model should answer. For example, a user may have a long-term preference for concise responses. This preference is not strongly tied to any specific type of question, and semantic-similarity retrieval cannot reliably retrieve such global user habits, causing them to be lost frequently [56]. Adaptive extraction based only on semantic similarity is more suitable for factual information queries and has poor recall for non-task preferences such as global user habits [57].

© 2026 The Author(s). Authors retain copyright in their articles.

3.2.4 Hierarchical Memory Architectures

Hierarchical memory architectures draw on human memory mechanisms and divide agent memory into several levels, generally short-term, medium-term, and long-term memory. Short-term memory corresponds to the model's current context window and stores the most recent rounds of raw dialogue. Medium-term memory stores initially summarized session content. Long-term memory stores deeply condensed core user characteristics and important historical events [58]. Migration logic is established between levels. After short-term memory accumulates to a certain size, it is summarized and refined, important information is moved to a higher level, and unimportant content is discarded. MemoryBank and HiMem are typical hierarchical memory systems [58-59].

A hierarchical architecture distinguishes the retention periods of different information. Raw dialogues, session summaries, and core user characteristics are stored separately, reducing the number of tokens read during a single inference. However, its design difficulty lies in the migration logic between levels. It is difficult to manually define a complete set of migration rules that determines which information should be promoted into long-term memory and which information should be discarded directly [21]. Most existing hierarchical memory systems still use general standards of information importance and lack migration logic specific to user-preference information. Whether global user habits are consolidated into long-term memory is therefore highly random [20]. Using user-preference characteristics as a basis for hierarchical migration decisions is essential for adapting hierarchical memory systems to personalization scenarios [58].

3.2.5 External Retrieval-Augmented Memory

External retrieval-augmented memory uses a vector database to persist historical interaction data. All historical dialogues are vectorized and stored in the database. When a user query arrives, it is converted into a vector and used for similarity retrieval in the vector database. Matching historical records are returned and concatenated into the model prompt for inference [18]. This method is a commonly used external-memory implementation in engineering applications. Its storage capacity is almost unlimited and it can retain user-interaction histories spanning several months [8].

A vector database is responsible only for storage and similarity matching and cannot understand changes in user preferences. The performance ceiling of the entire system depends on the retrieval strategy, memory updates, and memory down-weighting logic. Old records of user habits remain in the vector database. Without corresponding down-weighting and forgetting mechanisms, preference records that are no longer valid may still be retrieved and interfere with the model's current output [57]. Semantic representation errors in vector embeddings also cause retrieval errors, and some user-preference information cannot be recalled normally. Retrieval-augmented memory is only a storage and retrieval foundation. To support long-term personalization tasks, it must be combined with preference-aware memory-update and forgetting strategies [20].

4. Progress in Intersecting Research and Analysis of the Coupling Mechanism

4.1 Current Intersecting Research on Personalization and Long-Context Memory

As researchers have gradually recognized that the two systems cannot be designed separately, studies on their collaborative optimization have increased. Preference-aware memory updating is an important research route. It introduces extracted user-preference information into memory-system operations. Memory writing, scoring, compression, and forgetting no longer depend solely on text semantics; they also refer to the value weights of user preferences, prioritize the protection of dialogue segments containing user habits, and reduce the probability that preference information is deleted incorrectly during memory processing [20]. Such work demonstrates that user-preference information can reshape the filtering logic of the memory system in reverse and reduce preference loss during long-term personalization. However, most current preference-aware memory algorithms still use the user profile as external input. The extraction quality of the profile directly determines the final performance of the algorithm. If the extracted profile is biased, memory filtering will also be incorrect [21].

Structured persona-memory representation does not store historical dialogues as raw text. Instead, it parses user preferences into structured data, such as a graph, and stores them in the memory system. Graph nodes store specific user preferences, and edges record relationships among preferences. Compared with ordinary

© 2026 The Author(s). Authors retain copyright in their articles.

text records, structured representations can express the internal relationships among user characteristics more clearly, allow the system to read global usage habits quickly, and improve the poor recall of global preferences in traditional semantic retrieval [57]. However, structured representations also have clear shortcomings. Preference parsing itself causes information loss, some subtle implicit preferences are difficult to convert fully into graph nodes, and the dynamic updating and maintenance of the graph structure are difficult as the user produces new interactions [21].

Long-horizon personalization evaluation benchmarks are supporting research in this direction. Traditional LLM evaluation datasets mostly cover only a single session and cannot measure actual personalization performance under long-term cross-session interaction. Benchmarks such as LaMP-2, VitaBench 2.0, and AdaMem construct continuous user interactions spanning dozens or even hundreds of turns and can test personalization consistency, adaptation to preference evolution, and memory forgetting across sessions [5, 56, 60]. Only after the emergence of these evaluation tools could researchers quantitatively assess the effect of memory systems on personalization. However, task settings and evaluation indicators differ across datasets. Different studies use different benchmarks, making horizontal comparison difficult, and the evaluation system remains fragmented [6].

Training-free collaborative control at inference time attempts to connect the memory-reading process with personalized decoding. After the memory system retrieves relevant historical user information, the content is not only inserted into the prompt; the extracted user-preference characteristics also act directly on the model's decoding stage and guide preferences during sampling [41]. This approach does not require repeated user-specific training. It directly connects memory-retrieval output with personalization intervention during inference and enables data exchange between the memory and personalization modules. However, research in this area remains at an early stage, methods have limited maturity, and performance varies greatly among users [42].

Overall, intersecting research has confirmed that data exchange between the personalization and memory modules can substantially improve the final effect of long-term interaction. However, most existing collaborative methods still involve only simple data-level connections and have not achieved deep coupling. The optimization objectives within the two modules are still set separately, and no unified overall loss function or evaluation objective guides joint optimization of the complete system. Joint training and joint optimization remain unresolved difficulties in this direction [21].

4.2 Analysis of the Bidirectional Coupling Mechanism

The complete data-flow relationship between the two subsystems is shown in Figure 1.

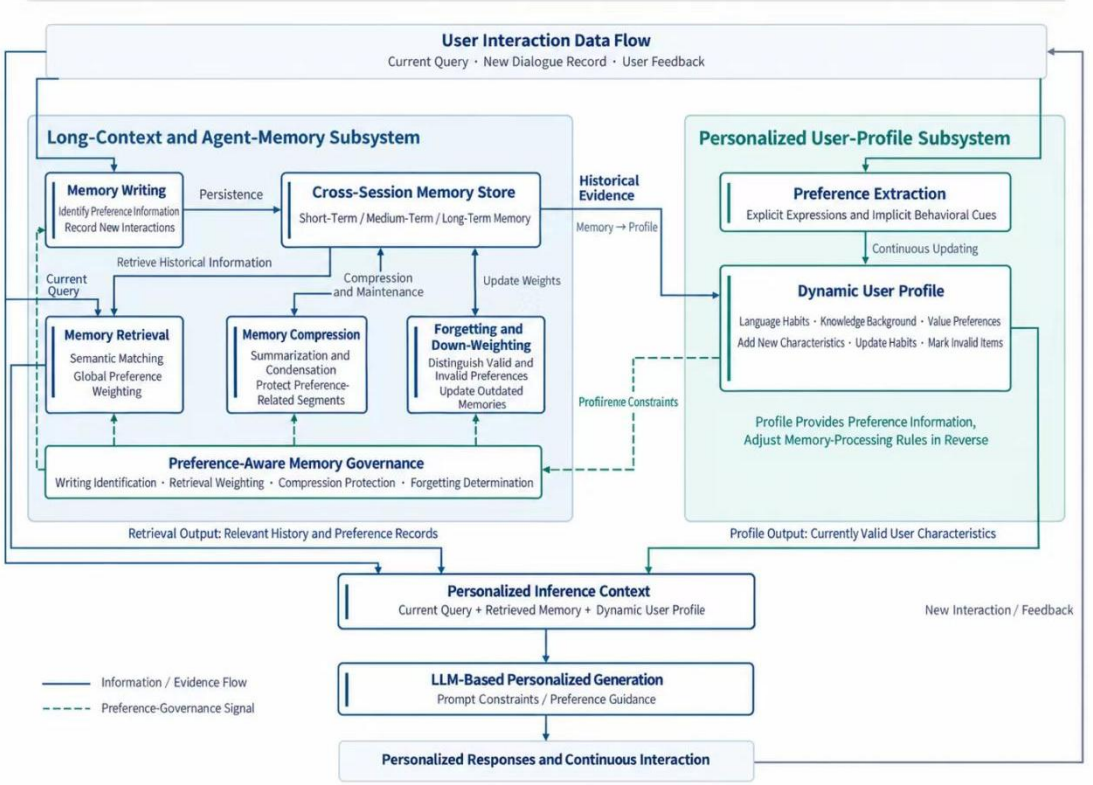
Personalization and long-context memory management form a bidirectional relationship of mutual reinforcement. First, long-context memory management provides the underlying information carrier for long-term personalization. All user preferences originate from past interactions. Without cross-session memory storage, all historical interactions are cleared whenever a new dialogue begins, and personalization is limited to a single session [20]. The writing, storage, and retrieval capabilities of the memory system determine how far back the model can access user interactions and directly determine how long the personalization effect can be maintained. Different memory technologies change how historical information is retained and accessed, which in turn changes the user preferences available to the model. Native windows, pruning and compression, hierarchical memory, and retrieval augmentation each impose different constraints on personalization. Defects in the memory module are transmitted upward and directly become performance bottlenecks for long-term personalization systems [21].

Second, personalized user preferences, in turn, constrain the internal processing logic of the memory system. Traditional memory systems optimize factual information storage and query accuracy. In personalization scenarios, the evaluation objective changes, and protecting user-preference information and adapting to the evolution of user habits become important tasks [19]. User-preference characteristics can act on the entire memory life cycle. During memory writing, the system identifies whether the current dialogue segment contains user preferences. During memory retrieval, it increases the matching weight of global preference records. During memory compression, it reduces the compression applied to segments containing preferences to avoid excessive information reduction. During memory forgetting and down-weighting, it distinguishes outdated preferences from valid preferences to avoid incorrectly down-weighting or deleting valid user habits

© 2026 The Author(s). Authors retain copyright in their articles.

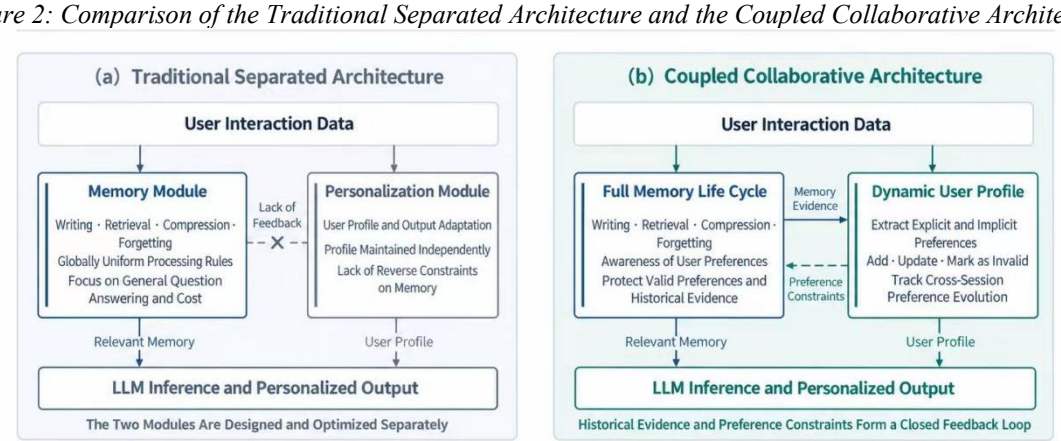
[20]. Without constraints from the personalization side, general memory logic processes dialogue history according to the standards of factual-query tasks, causing preference information to be lost [60].

Figure 1: Flowchart of a Coupled Collaborative System for Long-Term Personalized Agents



Traditional research mostly uses a separated architecture in which the memory module and personalization module are isolated from each other. The key differences between the two architectures are shown in Figure 2.

Figure 2: Comparison of the Traditional Separated Architecture and the Coupled Collaborative Architecture



Comparison Dimension	Traditional Separated Architecture	Coupled Collaborative Architecture
Memory Governance	Filtering, Compression, and Forgetting Based on General Rules	Preferences Guide Writing, Retrieval, Compression, and Forgetting
User-Profile Maintenance	Profile and Memory Processing Lack Bidirectional Feedback	Historical Evidence Updates the Profile, and the Profile Constrains Memory in Reverse
Evaluation and Optimization	Focus on Each Module's Individual Task Metrics	Collaborative Design Around Long-Term Consistency, Adaptation to Evolution, and Cost

Note: The two architectures are conceptual illustrations of the analytical framework in this paper; joint optimization remains a direction for further research.

© 2026 The Author(s). Authors retain copyright in their articles.

4.3 Analysis of Three Core Conflicts

First, a trade-off conflict exists between capacity and accuracy. The more historical interactions the memory system stores, the more user-related information is theoretically available. However, as the total amount of historical data increases, retrieval becomes more difficult and attention dilution intensifies. More irrelevant conversation instead interferes with the model's identification of actual user preferences [10]. Deliberately reducing memory capacity deletes many historical records and can reduce noise, but it can also directly remove important user-preference records. Therefore, long-term personalization systems always face a trade-off between memory-storage capacity and personalization accuracy. There is no fixed optimal storage size. The optimal threshold changes dynamically with the information density of each user's interactions, and a static upper limit on memory capacity cannot adapt to all users [60].

Second, a conflict exists between general governance rules and personalized characteristics. Most existing rules for memory compression, retrieval, and forgetting are fixed globally, and one set of parameters is applied to all users [20]. However, the forms of preference expression vary greatly among users. Some users' preferences are widely distributed across casual conversations, whereas other users' habits are concentrated in a few explicit statements. Fixed and uniform memory-processing rules cannot adapt to every user's characteristics. They may work reasonably well for some users but incorrectly delete preferences or fail to retrieve them for others [21]. Adapting memory-governance rules to individual users is essential for resolving this conflict [60].

Third, a conflict exists between computational cost and personalization stability. Maintaining high personalization stability requires retaining more historical interactions and performing more complex preference identification and memory filtering, which increases the computational cost of retrieval, summarization, and inference [8]. Reducing system computation requires simplifying memory processing and compressing historical storage, but this increases the probability of preference loss and reduces personalization stability. Engineering implementation requires a balance between the two. Current quantitative research on this trade-off remains insufficient and cannot provide quantitative reference indicators that can be used directly for engineering deployment [21].

4.4 Comparative Performance Analysis of Typical Memory Methods

To quantify the actual effects of different memory architectures on long-horizon personalization, this study compiled the core performance indicators of five representative memory methods on public evaluation datasets, as shown in Table 1. The overall numerical changes show that the five methods improve progressively in explicit-preference accuracy, cross-session consistency, and implicit-preference recall. The main reason for the performance differences is the degree of coupling between the memory system and the user-preference module, further confirming that their coupling relationship has a decisive effect on final personalization performance.

From the perspective of technical evolution, MemPrompt and general vector-retrieval memory are early separated designs. Their memory modules store and retrieve information only according to general semantic rules and are completely unaware of user-preference characteristics. Their explicit-preference accuracy remains in the range of 70% to 76%, but implicit-preference recall is below 60%, and cross-session consistency is the lowest. The fundamental reason is that the memory-processing workflow lacks reverse constraints from the user profile. General importance scoring and semantic-retrieval logic cannot identify implicit habitual information scattered across conversations and can easily delete preference-related segments incorrectly during compression and filtering, causing rapid deterioration of personalization during long-term interaction.

The hierarchical memory architecture represented by MemoryBank improves information-retention efficiency through a three-level storage structure and produces small improvements in all indicators compared with general retrieval methods. However, the hierarchical migration rules still use global general standards and do not assign specific retention priority to user preferences. Whether long-term user habits are consolidated into the long-term memory level is therefore highly random. Performance gains are limited and do not overcome the overall bottleneck of separated designs.

PAMU preference-aware memory and AdaMem adaptive memory are representative coupled collaborative methods. Both introduce user-preference characteristics into memory processing throughout the memory life

© 2026 The Author(s). Authors retain copyright in their articles.

cycle and use the user profile to constrain memory in reverse. PAMU uses preference weights to correct memory scoring and filtering rules, directly reducing the probability that preference information is deleted incorrectly and increasing implicit-preference recall to more than 70%. AdaMem further adapts memory-governance rules to individual users and dynamically optimizes parameters according to each user's interaction characteristics, so all three indicators reach their best levels. However, stronger coupling also increases computational complexity. Preference identification and rule adaptation add computational costs and are major constraints on large-scale deployment of such methods.

Overall, the memory system cannot optimize performance independently of the personalization module. Increasing storage capacity and retrieval speed alone cannot effectively improve long-horizon personalization. Significant improvements in evaluation indicators require a bidirectionally coupled collaborative mechanism in which user preferences are deeply involved throughout memory writing, retrieval, compression, and forgetting. This also indicates the core direction for subsequent technical optimization.

Table 1: Performance Comparison of Typical Memory Methods for Long-Horizon Personalization Tasks

Method	Technical Category	Evaluation Benchmark	Explicit-Preference Accuracy (%)	Cross-Session Consistency (%)	Implicit-Preference Recall (%)	Core Characteristics
MemPrompt	Basic Prompt-Memory Method	LaMP-2	71.2	62.3	48.7	Simple deployment and no additional training; rapid deterioration during long-term interaction and severe loss of implicit preferences.
General Vector-Retrieval Memory	External Retrieval-Augmented Method	LaMP-2	75.8	68.5	55.2	Almost unlimited storage capacity; semantic matching only, with poor recall of global preferences.
MemoryBank	Hierarchical Memory Architecture	VitaBench 2.0	78.3	74.6	61.4	Three-level hierarchical storage reduces token costs; general migration rules make preference consolidation highly random.
PAMU Preference-Aware Memory	Preference-Aware Update Method	LaMP-2	82.7	79.1	70.8	Preferences constrain the full memory workflow in reverse; performance depends on profile-extraction quality, and profile bias is propagated.
AdaMem Adaptive Memory	User-Centered Collaborative Method	AdaMem-Bench	85.4	83.6	76.3	User-adaptive governance rules; high computational cost and expensive large-scale deployment.

5. Major Challenges Facing Current Research

First, adapting to the dynamic evolution of user preferences is difficult. Users' views and usage habits do not remain unchanged permanently but change over time. The system must identify newly developed preferences and promptly identify outdated preferences to reduce interference from old records. Distinguishing between a preference that has temporarily not appeared and a preference that has changed completely is difficult. Users often do not directly state that their views have changed, so changes can only be inferred

© 2026 The Author(s). Authors retain copyright in their articles.

indirectly from scattered conversations [45]. Most current systems use relatively simple methods to process preference evolution and lack a mature mechanism for determining whether an old preference remains valid [44].

Second, the recall of implicit preferences can fail. Compared with explicit preferences stated directly, implicit user habits are distributed across extensive historical conversations and lack a unified and explicit textual description. Summarization and compression, memory pruning, and semantic retrieval are all poorly suited to implicit preferences. Vector embeddings cannot represent implicit preferences precisely, and scoring mechanisms have difficulty identifying their value. As a result, such preferences are often stored in the memory database but cannot be retrieved normally [19]. Failed recall of implicit preferences is an important cause of deteriorating personalization after long-term interaction and is a major difficulty in current research [57].

Third, memory errors and preference drift occur. A memory error means that the memory system produces an incorrect user-preference record, incorrectly parses dialogue content, and extracts a user habit that does not actually exist. As interaction turns accumulate, incorrect memories can be retrieved repeatedly, and the incorrect information is continuously reinforced. Preference drift then occurs, and the user characteristics retained by the model deviate from the user's actual habits [60]. Once an incorrect memory is consolidated into the long-term memory level, it is difficult to remove. Current technologies have very limited methods for detecting memory errors [21].

Fourth, the evaluation system is fragmented. The long-horizon personalization datasets and evaluation indicators used by different research teams are not uniform. Some evaluations focus more on the accuracy of explicit-preference matching, while others emphasize implicit preferences. The interaction lengths and task types in the datasets also differ. Research lacks a unified test standard, making experimental results from different papers difficult to compare and hindering technical iteration across the field [6]. Establishing a commonly accepted unified evaluation system is an urgent need for subsequent development [56].

Fifth, engineering implementation costs are difficult to balance. When long-term personalized agent services are deployed for large numbers of users, an independent personal memory database must be maintained for each user. Memory writing, vectorization, retrieval, and summarization all consume server computing resources [8]. Interaction frequency varies greatly among users, and frequent users produce extensive dialogue histories that create substantial computational pressure. Controlling total computing costs while maintaining personalization quality and balancing service quality against operating cost are unavoidable practical challenges in moving from academic research to product implementation [21].

6. Future Research Paths

First, cognitively inspired hierarchical memory architectures should be constructed. Agent memory-level migration logic should be redesigned according to the cognitive principles of human memory. General importance standards should no longer be used, and user-preference identification results should be incorporated into memory-level migration decisions [58]. Information identified as user preferences should receive higher retention priority. Temporary factual information should be distinguished from long-term user habits, and migration and elimination rules among short-term, medium-term, and long-term memory should be optimized to reduce the loss of user-preference information during movement between levels. The capacity limit of each memory level should also be adapted to each user instead of using a globally uniform static capacity [60].

Second, preference-aware context compression and pruning algorithms should be developed. The traditional compression and pruning logic based only on textual-semantic scores should be changed by embedding a preference-identification submodule in the compression workflow to identify whether text segments contain explicit or implicit user preferences [20]. Dialogue segments containing user preferences should receive less compression and higher retention weights to prevent irreversible deletion of preference information during text reduction. Specific identification strategies should be designed for implicit preferences to reduce the damage caused by summarization and compression to distributed implicit habits and to produce a better trade-off between the token budget and preference protection [19].

Third, a multimode hybrid indexing and retrieval mechanism should be developed. The mode that relies only on semantic vector similarity should be replaced with a hybrid indexing system. In addition to a basic

© 2026 The Author(s). Authors retain copyright in their articles.

semantic vector index, a separate user-preference index should be added, with an independent index entry for every extracted user habit [57]. Retrieval should refer to both semantic similarity and preference-index matching results. This would increase the recall probability of global user habits, improve the poor retrieval of global preferences that are not tied to specific tasks in traditional semantic retrieval, and reduce the frequency of failed implicit-preference recall [60].

Fourth, a unified long-horizon personalization evaluation protocol should be established. The advantages of existing long-horizon personalization datasets should be integrated to form a relatively accepted benchmark for the field [56]. Evaluation indicators should not calculate only preference-matching accuracy within a single session. They should include cross-session consistency, the ability to follow preference evolution, implicit-preference recall, the incidence of memory errors, and performance changes at different interaction lengths [5]. A unified evaluation protocol would enable horizontal comparison among different algorithms and accelerate iteration across the field [6].

Fifth, a mechanism for verifying and clearing memory errors should be established. User-preference records stored in the memory database should be checked and compared regularly. The latest user interactions should be used to determine whether old preference records still conform to the user's current actual habits and to identify incorrect memories and outdated historical preferences [45]. Confidence weights should be assigned to memory records. Retrieval weights should be gradually reduced for preference records whose confidence continues to decline, and confirmed incorrect memory entries should be removed. This would prevent repeated propagation and accumulation of incorrect memories and reduce preference drift during long-term interaction [60].

Sixth, long-term personalization should be extended to multimodal settings. Most current related research is limited to text-only dialogue, whereas future intelligent agents will process large amounts of multimodal user-interaction data, including images and audio. Many personal preferences are reflected in image and audio content. Extracting user preferences from multimodal interaction streams, incorporating multimodal preference information into memory systems, and constructing long-term personalization systems in multimodal environments are important future directions [7]. At present, relatively few studies address multimodal long-term personalization, leaving substantial room for exploration [21].

7. Conclusion

Long-term personalization is an important direction for moving LLMs toward personal applications. Personalization and long-context management are not two independent technical modules but have a close coupling relationship. The long-context memory system provides the underlying cross-session information carrier for personalization, while user preferences determine the criteria for writing, retrieving, compressing, and forgetting within the memory system. If the two modules are optimized separately and only one-sided technical indicators are considered, three inherent conflicts arise: capacity versus accuracy, general rules versus individual characteristics, and computation versus stability.

Current research in this field still faces practical challenges, including difficulty adapting to the dynamic evolution of user preferences, failed recall of implicit preferences, memory errors and preference drift, fragmented evaluation systems, and difficulty balancing engineering implementation costs. Future work should adopt a system-design approach based on coupled collaboration, develop cognitively inspired memory architectures, preference-aware context compression algorithms, multimode hybrid indexing and retrieval, and mechanisms for verifying and clearing memory errors, establish a unified long-horizon personalization evaluation system, and gradually extend research to multimodal personalization scenarios. Subsequent research should move beyond optimizing only the personalization algorithm or only the memory module and should use the coupling relationship between the two as the core starting point for system design, thereby further advancing long-term personalized LLM-agent technology toward mature implementation.

References

[1] Zhao M, Xu Y, Chen Q, et al. Toward Personalized LLM-Powered Agents: Foundations, Evaluation, and Future Directions[J]. arXiv preprint arXiv:2602.22680, 2026.

© 2026 The Author(s). Authors retain copyright in their articles.

[2] Liu X, Chen Y. Personalization of Large Language Models: A Survey[J]. arXiv preprint arXiv:2411.0027, 2024

[3] Li Y, Wang Z. A Survey of Personalized Large Language Models: Progress and Future Directions[J]. arXiv preprint arXiv:2502.11528, 2025

[4] Luo J, Zhang Y. Large Language Model Agent: A Survey on Methodology, Applications and Challenges[J]. arXiv preprint arXiv:2503.21460, 2025

[5] Xu, P., Singh, A., & Choi, Y. (2025). LaMP-2: Benchmarking Long-Term Personalization for Large Language Models [Unverified preprint metadata; source manuscript lists arXiv:2501.08741].

[6] Kumar I, Viswanathan S, Yerra S, et al. Know It, Act on It: Investigating Memory Utilization in LLM Personalization[J]. arXiv preprint arXiv:2607.29433, 2026

[7] Nie, C., Fu, C., Zhang, Y., et al. (2026). PersonaVLM: Long-Term Personalized Multimodal LLMs [Unverified preprint metadata; source manuscript lists arXiv:2604.13074].

[8] Chhikara, P., Khant, D., Aryan, S., Singh, T., & Yadav, D. (2025). Mem0: Building Production-Ready AI Agents with Scalable Long-Term Memory [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2504.19413

[9] Huang, J., & Li, Z. (2024). A Survey on Long-Context Large Language Models [Unverified preprint metadata; source manuscript lists arXiv:2406.09128].

[10] Gao, F., & Yang, Z. (2025). Long-Context, Less Focus: Understanding Information Dilution in LLMs [Unverified preprint metadata; source manuscript lists arXiv:2502.13572].

[11] Maharana, A., Lee, D. H., Tulyakov, S., Bansal, M., Barbieri, F., & Fang, Y. (2024). Evaluating Very Long-Term Conversational Memory of LLM Agents [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2402.17753

[12] Sun, T., & Zhang, H. (2024). Context Pruning Methods for Large Language Models: A Survey [Unverified preprint metadata; source manuscript lists arXiv:2409.08391].

[13] Zhang, S., & Liu, H. (2024). Dynamic User Profiling for Continual LLM Personalization [Unverified preprint metadata; source manuscript lists arXiv:2408.15217].

[14] Chen, L., & Zhou, H. (2024). Lightweight Personalization for LLMs: Prompt, LoRA and RAG Comparison [Unverified preprint metadata; source manuscript lists arXiv:2410.02744].

[15] Wu, X., & Zhou, R. (2025). Dynamic User Profile Extraction from Multi-Session Dialogues [Unverified preprint metadata; source manuscript lists arXiv:2502.09597].

[16] Peng, B., & Lin, Y. (2025). A Review of Long-Context Techniques for Large-Language-Model-Based Agents [Unverified preprint metadata; source manuscript lists arXiv:2504.02381].

[17] Han, Z., & Wang, J. (2025). Memory Mechanisms for LLM Agents: Techniques and Evaluations [Unverified preprint metadata; source manuscript lists arXiv:2507.02203].

[18] Wu, Z., & Zhang, X. (2024). Survey on Retrieval-Enhanced Memory for Conversational Agents [Unverified preprint metadata; source manuscript lists arXiv:2412.04254].

[19] Ding, S., & Qian, Y. (2025). Risk of Erasing User Preference by Generic Context Pruning [Unverified preprint metadata; source manuscript lists arXiv:2512.02748].

[20] Liu H, Wang J. Preference-Aware Memory Update for Long-Term LLM Agents[J]. arXiv preprint arXiv:2510.09720, 2025

[21] Zhang Y, Wang H. From Storage to Experience: A Survey on the Evolution of LLM Agent Memory Mechanisms[J]. arXiv preprint arXiv:2605.06716, 2026

[22] Jiang, T., & Li, S. (2024). Challenges in Real-World LLM Personalization [Unverified preprint metadata; source manuscript lists arXiv:2405.13219].

© 2026 The Author(s). Authors retain copyright in their articles.

[23] Madaan, A., Tandon, N., Gupta, P., et al. (2022). MemPrompt: Memory-Assisted Prompting for Large Language Models [Unverified preprint metadata; source manuscript lists arXiv:2210.11797].

[24] Salemi, A., Kim, S., & Schölkopf, B. (2023). Prompt Compression for Large Language Models [Unverified preprint metadata; source manuscript lists arXiv:2305.14821].

[25] Shi, W., & Liu, J. (2025). Token Budget Management for Personalized System Prompts [Unverified preprint metadata; source manuscript lists arXiv:2503.12774].

[26] Lin, Y., & Zhang, Q. (2024). User-Profile Summarization for LLM Personalization [Unverified preprint metadata; source manuscript lists arXiv:2412.08152].

[27] Zhou, K., & Wu, L. (2025). Abstractive Compression of User Interaction History for Personalized Agents [Unverified preprint metadata; source manuscript lists arXiv:2507.11364].

[28] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., & Chen, W. (2021). LoRA: Low-Rank Adaptation of Large Language Models [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2106.09685

[29] Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., de Laroussilhe, Q., Gesmundo, A., Attariyan, M., & Gelly, S. (2019). Parameter-Efficient Transfer Learning for NLP [Preprint]. arXiv. https://doi.org/10.48550/arXiv.1902.00751

[30] Li, X. L., & Liang, P. (2021). Prefix-Tuning: Optimizing Continuous Prompts for Generation [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2101.00190

[31] Zhang, R., & Liu, Y. (2025). Overfitting Analysis for User-Specific LoRA Personalization [Unverified preprint metadata; source manuscript lists arXiv:2505.02873].

[32] Wang, Z., & Chen, M. (2024). Data Scarcity in Individual User LLM Personalization [Unverified preprint metadata; source manuscript lists arXiv:2407.16291].

[33] Lee, J., & Park, S. (2025). Incremental LoRA for Continual User Preference Update [Unverified preprint metadata; source manuscript lists arXiv:2508.15623].

[34] Zhao, H., & Li, W. (2024). Catastrophic Forgetting in Continual LLM Personalization [Unverified preprint metadata; source manuscript lists arXiv:2411.08476].

[35]Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., & Lowe, R. (2022). Training language models to follow instructions with human feedback [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2203.02155

[36] Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D., & Finn, C. (2023). Direct Preference Optimization: Your Language Model is Secretly a Reward Model [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2305.18290

[37] Xie, Z., Wu, J., Shen, Y., Xia, Y., Li, X., Chang, A., Rossi, R., Kumar, S., Majumder, B. P., Shang, J., Ammanabrolu, P., & McAuley, J. (2025). A Survey on Personalized and Pluralistic Preference Alignment in Large Language Models [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2504.07070

[38] Liu, Y., & Yang, B. (2024). User-Specific DPO for Personalized Large Language Models [Unverified preprint metadata; source manuscript lists arXiv:2406.17819].

[39] Zhang, H., & Wang, Y. (2025). Meta-Alignment for Fast User Adaptation of LLMs [Unverified preprint metadata; source manuscript lists arXiv:2501.04211].

[40] Tan, S., & Li, Q. (2025). Decay of Preference Alignment Over Extended Dialogue Sessions [Unverified preprint metadata; source manuscript lists arXiv:2509.14726].

[41] Chen, J., & Gong, L. (2025). Inference-Time Personalized Alignment Without Per-User Tuning [Unverified preprint metadata; source manuscript lists arXiv:2506.03948].

[42] Wang, P., & Zeng, H. (2024). Preference-Guided Decoding for Zero-Shot LLM Personalization [Unverified preprint metadata; source manuscript lists arXiv:2410.14635].

© 2026 The Author(s). Authors retain copyright in their articles.

[43] Wang, T., Tao, M., Fang, R., Wang, H., Wang, S., Jiang, Y. E., & Zhou, W. (2024). AI PERSONA: Towards Life-long Personalization of LLMs [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2412.13103

[44] Zheng, Y., & Guo, S. (2026). Evolving User Persona for Long-Term LLM Agents [Unverified preprint metadata; source manuscript lists arXiv:2601.12587].

[45] Yu, L., & Zhang, T. (2025). Detection of Out-of-Date Preferences in Dynamic Profiles [Unverified preprint metadata; source manuscript lists arXiv:2511.05472].

[46] Meng, X., & He, K. (2024). Profile Distortion Caused by Context Window Truncation in User Modeling [Unverified preprint metadata; source manuscript lists arXiv:2408.19762].

[47] Feng, Q., & Sun, B. (2026). Benchmark for Dynamic User Profile Evolution [Unverified preprint metadata; source manuscript lists arXiv:2603.11892].

[48] Beltagy, I., Peters, M. E., & Cohan, A. (2020). Longformer: The Long-Document Transformer [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2004.05150

[49] Zaheer, M., Guruganesh, G., Dubey, A., Ainslie, J., Alberti, C., Ontanon, S., Pham, P., Ravula, A., Wang, Q., Yang, L., & Ahmed, A. (2020). Big Bird: Transformers for Longer Sequences [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2007.14062

[50] Ding, J., & Zhou, W. (2025). Sparse Attention Variants for Extending LLM Context Windows [Unverified preprint metadata; source manuscript lists arXiv:2503.07415].

[51] Chen, Y., & Zhang, L. (2025). Empirical Study of Attention Dilution in Million-Token LLM Contexts [Unverified preprint metadata; source manuscript lists arXiv:2505.08917].

[52] Sui, X., & Liu, Z. (2024). When Bigger Context Window Does Not Mean Better Performance for LLMs [Unverified preprint metadata; source manuscript lists arXiv:2410.17123].

[53] Kim, N., & Cho, J. (2023). Context Pruning by Importance Scoring for Long-Document LLMs [Unverified preprint metadata; source manuscript lists arXiv:2307.06904].

[54] Xu, J., & Yu, H. (2024). Sliding-Window Abstractive Summarization for Long Dialogue History [Unverified preprint metadata; source manuscript lists arXiv:2404.05911].

[55] Yang, M., & Qian, R. (2025). Hybrid Vector-Text Compression for Agent Conversation Context [Unverified preprint metadata; source manuscript lists arXiv:2506.18422].

[56] Wang X, Zhou L. VitaBench 2.0: Evaluating Personalized and Proactive Agents in Long-Term User Interactions[J]. arXiv preprint arXiv:2605.27141, 2026

[57] Ji, S., Li, Y., & Hooi, B. (2026). Memory is Reconstructed, Not Retrieved: Graph Memory for LLM Agents [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2606.06036

[58] Zhang, N., & Yang, X. (2026). HiMem: Hierarchical Long-Term Memory for LLM Long-Horizon Agents [Unverified preprint metadata; source manuscript lists arXiv:2602.10147].

[59] Zhong, W., Guo, L., Gao, Q., Ye, H., & Wang, Y. (2023). MemoryBank: Enhancing Large Language Models with Long-Term Memory [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2305.10250

[60] Wang R, Lu H. AdaMem: Adaptive User-Centric Memory for Long-Horizon Dialogue Agents[J]. arXiv preprint arXiv:2603.16496, 2026

Funding

This research received no external funding.

Conflicts of Interest

The authors declare no conflict of interest.

© 2026 The Author(s). Authors retain copyright in their articles.

Acknowledgment

This paper is an output of the science project.

Copyrights

Copyright for this article is retained by the author (s), with first publication rights granted to the journal. This is an open-access article distributed under the terms and conditions of the Creative Commons Attribution license (http://creativecommons.org/licenses/by/4.0/).

© 2026 The Author(s). Authors retain copyright in their articles.