From Volume to Value:

Preference-Aligned Memory Construction for On-Device RAG

Changmin Lee, Jaemin Kim, Taesik Gong
{lchm1106, jm611, taesik.gong}@unist.ac.kr

ICML International Conference On Machine Learning

UNIST | U*AI Ubiquitous Artificial Intelligence Laboratory

Scan Here!

Project Page

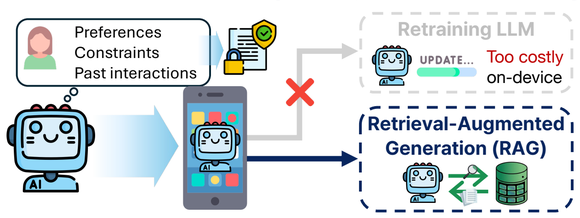
Key Takeaway

For **on-device personalized RAG**, the key is not to store more, **but to store what matters.**

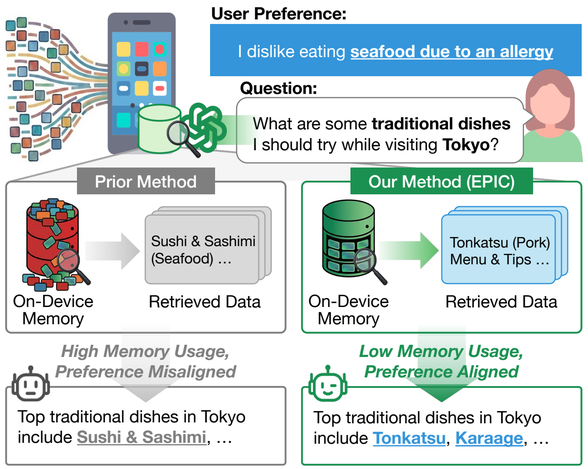
How can we build practical personal agents that run entirely on-device?

Motivation

Personalization needs private, on-device memory

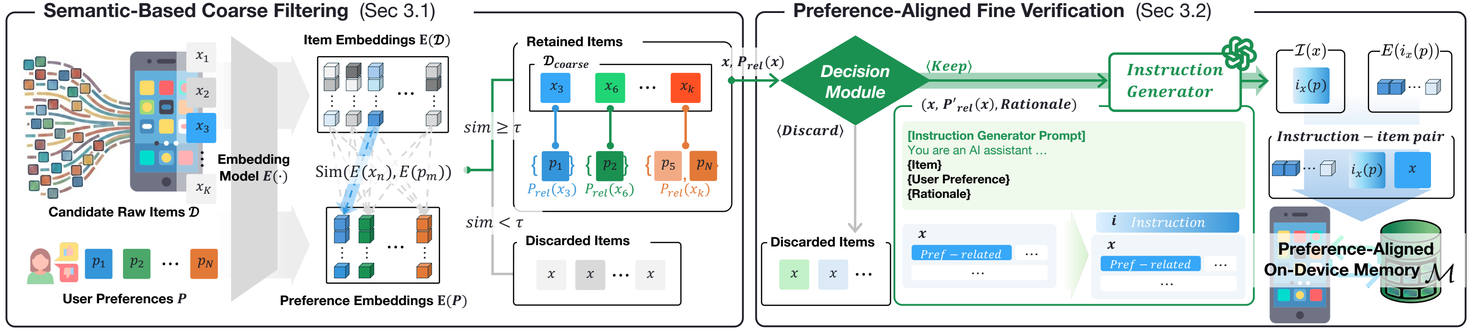


From "how to use" to "what to store"



Our Method: EPIC: Efficient Preference-aligned Index Construction

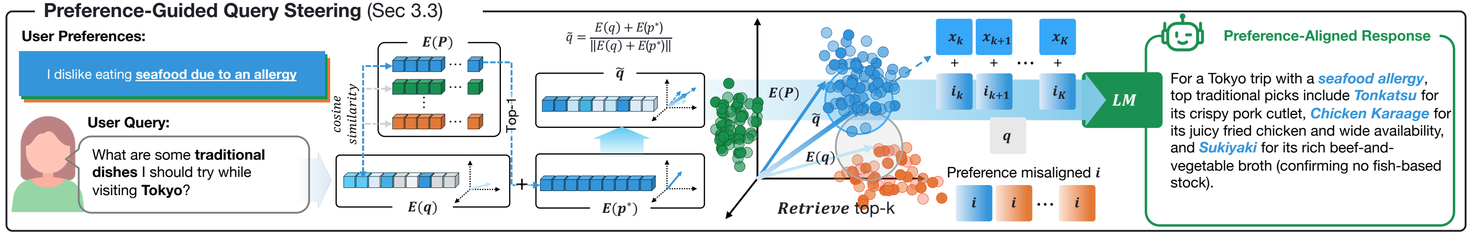
EPIC works in three steps



Retain Preference-related candidates

Verify and generate preference-guided instructions

Preference-Guided Query Steering (Sec 3.3)



Steer retrieval toward the relevant preference

Benchmarks

Diverse Corpora

■ ours ■ existing

1. Static Knowledge	PrefWiki (Recommendation)	
Wikipedia	PrefRQ (Debate)	
2. Web Footprint	PrefELI5 (Explanation)	
Common Crawl		
3. Conversation Logs	PrefEval[1] (conversation)	
LMSYS-Chat		

Per-Persona Indexing

Preference: "I avoid spicy food"
Question: "What food should I try when traveling to Seoul?"

$X\ N =$ *persona*

Each persona contains multiple preference-question pairs designed to expose preference violations.

Evaluation

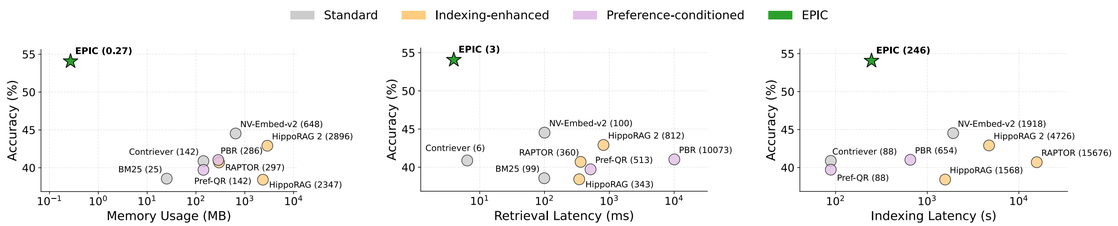
Generated response → LLM judge → 4 error types

Error-free response rate = preference-following accuracy

Reference [1] "Do LLMs Recognize Your Preferences? Evaluating Personalized Preference Following in LLMs". ICLR 2025.

Experiment Results

Server-Side Results



(a) On-disk memory usage. (b) End-to-end retrieval latency. (c) End-to-end indexing latency.

2,404x Less Memory

33.33x Faster Retrieval

7.80x Faster Indexing

On-Device Results

Component	Jetson Orin Nano	MacBook Pro M4	Galaxy Z Flip 6
	Retrieval Latency (ms)		
Embedding query	29.03	3.36	3.85
Preference-Guided Query Steering (Sec. 3.3)	0.18	0.03	0.55
FAISS retrieval	0.14	1.84	0.81
Total Retrieval Latency	**29.35**	**5.23**	**5.21**
	Indexing Latency (ms)		
Embedding items	29.99	4.31	4.01
Semantic-Based Coarse Filtering (Sec. 3.1)	0.02	0.01	0.10
Preference-Aligned Fine Verification (Sec. 3.2)	70.73	47.09	245.65
Embedding instruction	0.02	0.01	0.00
Build FAISS	0.00	0.00	0.00
Total Indexing Latency	**102.67**	**51.42**	**249.76**