Enabling User Agency in Scalable Content Recommendations with Large Language Models

Yucheng Li
Amazon AGI
London, United Kingdom
yucheni@amazon.co.uk

Gerrit J.J. Van den Burg
Amazon AGI
London, United Kingdom
gvdburg@amazon.co.uk

Wei Liu
Amazon AGI
London, United Kingdom
weliuz@amazon.co.uk

Zhunxuan Wang
Amazon AGI
London, United Kingdom
wzhunxua@amazon.co.uk

Abhishek Tripathi
Amazon AGI
Cambridge, United Kingdom
dtriabhi@amazon.co.uk

Murat Sensoy
Amazon AGI
London, United Kingdom
msensoy@amazon.co.uk

ABSTRACT

Existing content recommender systems usually depend on centrally stored interaction histories, creating vendor lock-in and disadvantaging newer providers who lack sufficient user data. They also limit users' ability to understand, control, or edit how their preferences are represented, since profiles are learned as opaque latent vectors within provider-controlled models. We propose a user-centric alternative in which personal agents construct interpretable, editable preference profiles in natural language. Each profile item is associated with a learnable weight indicating its importance, and profiles are learned locally under full user control, laying the groundwork for high-quality personalization across multiple content providers. Recommendations are generated by matching content with weighted profile embeddings in a shared embedding space that is fine-tuned once using only content data and subsequently used by both content providers and personal agents. This design shifts profile ownership to users while maintaining the efficiency of existing recommender systems, as online recommendation reduces to approximate nearest-neighbor search. It further lowers the barrier for new providers, who only need to embed their content into the shared space — personalization naturally emerges from user-side profile embeddings optimized by personal agents to retrieve the most relevant content. Experiments on the MIND and Goodreads datasets show that our system outperforms strong baselines while providing transparency and editability — reimagining personalization as a process owned and controlled by the user.

KEYWORDS

Agents; Interpretable Recommendations; Large Language Models

ACM Reference Format:
Yucheng Li, Gerrit J.J. Van den Burg, Wei Liu, Zhunxuan Wang, Abhishek Tripathi, and Murat Sensoy. 2026. Enabling User Agency in Scalable Content Recommendations with Large Language Models. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 10 pages. https://doi.org/10.65109/CBUQ6936

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/CBUQ6936

1 INTRODUCTION

Online digital platforms (content providers) serve billions of users across vast content collections, making personalized recommendations essential for navigation and discovery [3, 9, 45]. Yet current systems exhibit fundamental limitations that undermine user privacy, autonomy, and agency. Centralized architectures require users to surrender detailed behavioral data to content providers, creating privacy concerns and vendor lock-in [14, 33]. Opaque latent representations prevent users from understanding or correcting how their preferences are modeled [13, 36, 43]. This centralization further entrenches market incumbents, who leverage vast proprietary datasets to train powerful embedding models, while new providers face insurmountable cold-start barriers.

Recent research has begun to address these shortcomings by leveraging large language models (LLMs) to make recommender systems more interpretable and transparent [24]. LLMs offer a promising path toward representing user preferences in natural language — a form users can understand, inspect, and even edit directly. However, existing approaches remain partial solutions: they improve interpretability but do not fully resolve challenges around data privacy, scalability, or accessibility for new providers.

These observations motivate a fundamental question: *How can we design recommendation systems that give users control over their profiles, broaden provider accessibility, and remain scalable?* We argue that achieving this requires satisfying three critical desiderata:

D1. User Interpretability and Control: Users should understand how their preferences are represented and be able to edit these representations directly in natural language while maintaining full ownership of their profiles.

D2. Scalable Retrieval: The system must efficiently handle millions of items using standard approximate nearest-neighbor (ANN) search, maintaining the performance characteristics of existing recommender systems.

D3. Cross-Provider Portability: User profiles should be designed to work seamlessly across multiple providers, enabling users to receive high-quality personalized recommendations without vendor lock-in or requiring providers to collect extensive user data.

Meeting all three desiderata simultaneously exposes a fundamental tension: while LLMs excel at generating rich, human-readable preference descriptions, they are computationally infeasible for

scoring millions of items at query time. Conversely, embedding-based retrieval methods scale efficiently but sacrifice interpretability and user control over preference representations.

We resolve this tension through a novel personal agent architecture that simultaneously satisfies all three desiderata at different levels. Our key insight is that natural language user profiles need not be static summaries—they can be iteratively optimized as computational objects while remaining interpretable and editable. By decoupling profile learning (which happens locally using LLMs) from content retrieval (which uses efficient embeddings), we achieve the interpretability benefits of language models without sacrificing the scalability of vector-based search. Our architecture realizes these desiderata through three key components:

(1) **Personal Agents** (*D1*): Each user maintains a local agent that learns interpretable natural language profiles from interaction history and optimizes them for recommendation quality. Users can inspect and edit these profiles directly, maintaining full ownership and control over their preference representations.
(2) **Shared Embedding Space** (*D2*): A centrally maintained embedding model maps user interests to content descriptions in natural language, enabling efficient similarity search. This space is trained on synthetic content-interest pairs generated by LLMs from content samples without user behavioral data.
(3) **Content Retrieval API** (*D3*): Providers embed their catalogs into the shared embedding space and expose recommendation endpoints. These endpoints use ANN search to return relevant content when queried with profile embeddings. This enables cross-provider portability, allowing new providers to deliver personalized results immediately by focusing solely on embedding their content, without collecting behavioral data or building dedicated recommendation models.

Together, these components enable interpretable and user-controlled recommendations without sacrificing scalability. User preferences become transparent and directly editable, profiles are portable across providers by design, so both new and established providers can deliver competitive recommendations using pre-optimized profile embeddings. Crucially, the embedding-aligned optimization of natural-language profiles allows our approach to achieve the recommendation quality of centralized systems while preserving user control and cross-provider portability in principle. Experimental validation on MIND [37] and Goodreads [30] demonstrates that our approach outperforms strong baselines [13, 16, 36], while providing transparent, interpretable, and editable preference profiles. Our results point toward a new generation of user-centric recommendation systems, where intelligent agents give users direct control over their preferences and enable opportunities for seamless portability across providers without compromising scalability or performance.

The remainder of this paper is organized as follows. Section 2 presents our system architecture, detailing the design of the shared embedding space, personal agents, and the recommendation interface. Section 3 describes the experimental setup, datasets, and evaluation metrics, and reports our empirical results, including recommendation performance, ablation studies, and profile editability analysis. Section 4 reviews related work and contrasts it with our contributions. Finally, Section 5 concludes the paper and outlines directions for future research.

2 SYSTEM ARCHITECTURE

Our system implements interpretable recommendations through three interconnected components that work together to satisfy the four desiderata outlined in the introduction. As illustrated in Figure 1, the architecture comprises: (1) a shared content-interest embedding space that enables efficient similarity computation, (2) personal agents that learn and optimize interpretable user profiles locally, and (3) a content retrieval API that allows providers to serve personalized recommendations without accessing raw user data. We introduce each component progressively, building from the foundational embedding space to the user-side agents and finally to the provider-side retrieval interface.

2.1 Shared Content–Interest Embedding Space

The foundation of our architecture is a centrally maintained embedding space defined in $\mathbb{R}^n$, which maps natural language interest descriptions to content representations. This shared latent space addresses a critical challenge: *while personal agents must represent user preferences in interpretable natural language, content retrieval requires efficient vector-based similarity computation over millions of items.* The embedding space bridges these requirements by enabling agents to translate human-readable profiles into dense embeddings suitable for approximate nearest-neighbor search.

Training this embedding model presents a bootstrapping challenge: we require content–interest pairs to learn the mapping, but such pairs are typically unavailable in real-world data. Moreover, even when implicit signals exist, accessing them could violate user privacy. In practice, content providers may observe which users engage with specific content, but not the explicit natural-language motivations behind those interactions. We address this by adopting a self-supervised approach based solely on publicly available content. For each content item in the training corpus, we prompt an LLM to generate plausible user interests that might motivate engagement with that item. This produces synthetic training pairs (c_i, d_i) of content c_i and a corresponding interest description d_i.

Table 1 illustrates this process with examples from the MIND dataset. For instance, given a news article about Rihanna's social media post, the LLM generates the interest "Pop music and celebrity news." These synthetic pairs enable us to fine-tune an embedding model $m_\theta(\cdot)$ using a contrastive learning objective, where semantically related content and interests are embedded nearby, while unrelated pairs are pushed apart. The training objective is:

$$\mathcal{L} = -\frac{1}{|\mathcal{D}|} \sum_{(c_i,d_i)\in\mathcal{D}} \log \frac{\exp(\text{sim}(m_\theta(c_i), m_\theta(d_i)))}{\sum_{k=1}^{N} \exp(\text{sim}(m_\theta(c_i), m_\theta(d_k)))} \quad (1)$$

where $\exp(\cdot)$ denotes the exponential function, $m_\theta(c_i)$ and $m_\theta(d_i)$ are the content and interest embeddings in $\mathbb{R}^n$, $\mathcal{D}$ is the training dataset, $\text{sim}(\cdot,\cdot)$ is the cosine similarity, and N is the batch size. Each content–interest pair (c_i, d_i) serves as a positive example, while other interests d_k $(k \neq i)$ in the batch serve as negative samples.

This objective shapes the embedding space such that user interests expressed in natural language align closely with relevant

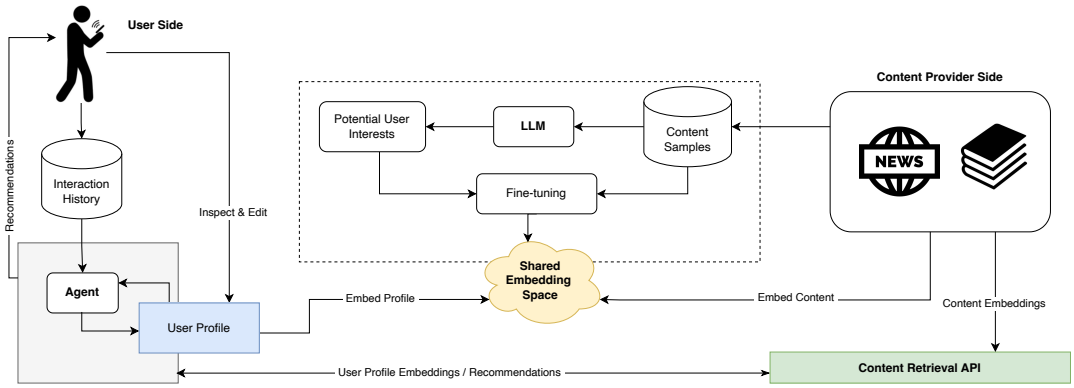


Figure 1: High-level overview of the proposed system, showing (i) personal agents learning user profiles locally, (ii) a shared embedding space trained on content samples using self-supervision, and (iii) a content retrieval API that allows providers to return recommendations based on user profile embeddings in the shared embedding space.

Content	Interests
[music/musicnews]: Is Rihanna Taking a Break? Read Her Cryptic Post About Her 'Overwhelming' Year - RiRi wrote "BRB"...	Pop music and celebrity news
[news/newspolitics]: Major donor who played both sides of the aisle charged with campaign violations and fraud - A...	Government lobbying activities
[finance/finance-real-estate]: Homes in these 25 waterfront cities are a total steal - Plenty of affordable waterfront...	Waterfront real estate

Table 1: Examples of synthetic *(Content, Interest)* pairs generated for embedding model training. Each content item may appear in multiple pairs corresponding to its different interests.

content, enabling efficient approximate nearest-neighbor retrieval while providing meaningful similarity scores for profile optimization. The resulting model thus serves a dual role: it allows scalable content retrieval and provides the reward signal necessary for personal agents to optimize user profiles in natural language.

2.2 Personal Agent

Personal agents operate locally on user devices (or environments dedicated to their users), learning and optimizing interpretable natural language profiles that represent user preferences. Each agent maintains a structured profile consisting of specific user interests (e.g., "investigative journalism about corporate governance," "indie folk music with acoustic arrangements") along with learned weights that indicate the relative importance of each interest for user. The agent's goal is to discover the profile that maximizes recommendation quality while remaining human-readable and editable.

2.2.1 Profile-Based Content Ranking. Given a user profile with natural language interests $d_1, d_2, \ldots, d_m$, the agent must efficiently rank candidate content items by relevance. This requires aggregating similarity scores across multiple interests while accounting for their varying importance to the user. A naive approach might weight interests directly by their historical click frequencies. However, such a strategy ignores valuable negative evidence — content that was shown but not clicked — and thus biases the profile toward categories with large numbers of impressions, even when their relative engagement is low. To address these limitations, we learn optimal interest weights directly from historical user engagement signals. For a content item c and the profile with normalized interest embeddings $\boldsymbol{I} = [n_\theta(d_1), \ldots, n_\theta(d_m)] \in \mathbb{R}^{n \times m}$ and the interest weights $\boldsymbol{w} = [w_1, \ldots, w_m] \in \mathbb{R}^{1 \times m}$ as learnable parameters, we compute the relevance score as:

$$\text{score}(c) = \boldsymbol{w}(\boldsymbol{I}^T \cdot n_\theta(c))$$

where $n_\theta(\cdot)$ is calculated by normalizing $m_\theta(\cdot)$ and $\boldsymbol{I}^T \cdot n_\theta(c)$ yields the vector of cosine similarities between content and each interest. This formulation is mathematically equivalent to computing cosine similarity between the content embedding and a weighted combination of interest embeddings when weights are normalised, enabling efficient vector-based search.

We learn interest weights by solving a least squares problem using historical user engagement. Given N items with embeddings $C = [n_\theta(c_1), \ldots, n_\theta(c_N)] \in \mathbb{R}^{n \times N}$ and binary engagement labels $\boldsymbol{b} \in \{0, 1\}^N$, we construct the feature matrix $\boldsymbol{A} = \boldsymbol{C}^T \boldsymbol{I} \in \mathbb{R}^{N \times m}$ where each row represents how well each interest matches a historical item. The optimal weights solve:

$$\hat{\boldsymbol{w}} = (\boldsymbol{A}^T \boldsymbol{A})^{-1} \boldsymbol{A}^T \boldsymbol{b}$$

followed by normalization: $w_k = \hat{w}_k / \sum_j \hat{w}_j$, which effectively captures the relative importance of interests while maintaining the mathematical properties needed for cosine similarity-based scoring.

2.2.2 Profile Learning and Optimization. The core innovation of our approach lies in treating natural language user profiles not as static summaries, but as optimizable objects that can be iteratively

Algorithm 1 Initial Profile Generation

Input: User's historical interactions H, a Large Language Model (LLM)
Output: Initial user profile P
$\mathbf{I} = [\ ]$
for each interaction $h \in H$ **do**
  I_h = GeneratePotentialInterests(LLM, h)
  $\mathbf{I}.extend(I_h)$
end for
clusters = HierarchicalClustering($\mathbf{I}$)
$P = [\ ]$
for each cluster c in clusters **do**
  d_c = GenerateInterestDescription(LLM, c)
  $P.add(d_c)$
end for
Return P

refined to maximize recommendation quality. While previous work generates coarse-grained user descriptions [2, 31, 44], we maintain structured profiles of specific interests that can be individually weighted, modified, and optimized.

Profile learning consists of two phases: initial construction from historical interactions, followed by iterative refinement guided by recommendation performance feedback. This two-stage approach ensures that profiles begin with reasonable coverage of user interests before optimization fine-tunes them for maximum recommendation effectiveness.

Initial Profile Construction. Direct summarization of user interaction histories often produces generic, oversimplified profiles that miss nuanced preferences. Instead, we adopt a bottom-up approach that first identifies specific interests from individual interactions, then consolidates them into coherent themes.

For each item in a user's interaction history, we prompt an LLM to extract potential underlying interests that would motivate engagement with that content. We then embed these interest descriptions using the trained model for shared content-interest embedding space and apply hierarchical clustering to group semantically related interests. Finally, for each cluster, we use an LLM to generate a unified interest description that captures the common theme while remaining specific and actionable.

Algorithm 1 formalizes this process. The result is a structured profile where each interest includes both a natural language description and the set of historical interactions that support it, as illustrated in Figure 2. To mitigate user cold-start when no interaction history is available, the agent can initialize a profile from brief natural-language inputs (e.g., "I enjoy sci-fi") or lightweight preference queries, thereby enabling immediate content personalization.

Iterative Profile Refinement. Initial profiles provide a reasonable starting point, but they may miss subtle preferences or fail to optimally balance different interests. More fundamentally, the goal is not just to summarize past behavior, but to discover the profile that yields the best future recommendations. We formulate this as a search problem: given a space of possible natural language profiles and a reward function based on recommendation quality, find the profile that maximizes expected user satisfaction.

The refinement process operates on a held-out validation dataset $D = (D_c, D_l)$ containing content items and binary engagement labels from user's interaction history. We define three key components that work together to guide profile improvement:

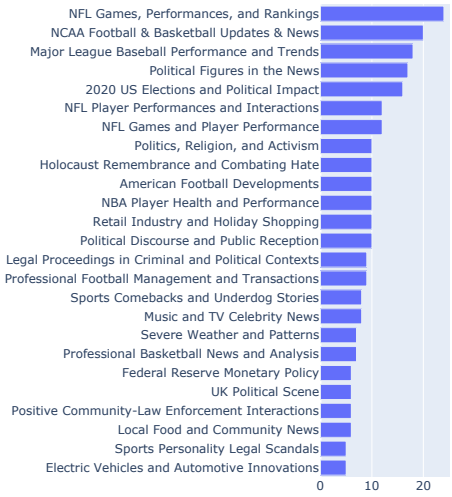


Figure 2: An example of the initial profile generation with number of supporting evidence for each interest.

Algorithm 2 Iterative Profile Refinement

Input: Initial profile P_0, user's data set $D = (D_c, D_l)$, Temperature T, Max iterations N
Output: Best weighted profile: (P^*, w^*)
$w_0 \leftarrow$ InterestWeights(P_0, D)
$o_0 \leftarrow$ PersonalizedRanking(P_0, w_0, D_c)
$F_0 \leftarrow$ BuildFeedback(o_0, D)
$(P^*, w^*), r^* \leftarrow (P_0, w_0)$, Reward($o_0, D_l$)
Initialize node set $S \leftarrow \{(P^*, w^*, r^*, F_0)\}$
for $i = 1$ to N **do**
  $(P, w, r, F) \sim P(s \in S) = \frac{\exp(r_S/T)}{\sum_{s' \in S} \exp(r_{s'}/T)}$
  $P_i \leftarrow$ UpdateProfile(LLM, P, Sample(F))
  $w_i \leftarrow$ InterestWeights(P_i, D)
  $o_i \leftarrow$ PersonalizedRanking(P_i, w_i, D_c)
  $r_i \leftarrow$ Reward(o_i, D_l)
  $F_i \leftarrow$ BuildFeedback(o_i, D)
  Add (P_i, w_i, r_i, F_i) to S
  if $r_i > r^*$ **then**
    $P^*, w^*, r^* \leftarrow P_i, w_i, r_i$
  end if
end for
Return (P^*, w^*)

(1) **Reward Function:** We evaluate profile quality using standard ranking metrics (AUROC, nDCG@5) computed by applying the current profile to *rank all items* in D_c and comparing against ground truth labels in D_l.
(2) **Performance Analysis:** For each profile evaluation, we categorize predictions into true/false positives and negatives using a 0.5 threshold on normalized scores. We also compute interest-level contributions to understand which aspects of the profile drive successful or unsuccessful recommendations.
(3) **LLM-Guided Refinement:** We prompt an LLM with the current profile and structured feedback about its strengths and weaknesses, asking it to generate an improved version. The LLM receives examples of correctly and incorrectly ranked items, along with explanations of how different interests contributed to each prediction.

Algorithm 2 formalizes our iterative refinement process. We maintain a set S of candidate profiles along with their performance

scores and diagnostic feedback. At each iteration, we sample a profile from S using a temperature-controlled distribution that balances exploration of diverse profiles with exploitation of high-performing ones.

The sampling probability for profile p with reward r_p is proportional to $\exp(r_p/T)$, where temperature T controls the exploration-exploitation tradeoff. For the sampled profile, we provide the LLM with structured feedback including specific examples where the profile succeeded or failed to predict user preferences. The LLM generates a refined profile, which we evaluate and add to S. This process continues for a fixed number of iterations, ultimately selecting the profile with highest validation performance.

Our experiments show that uniform exploration ($T = \infty$) consistently outperforms exploitation-focused strategies, suggesting that the ability to escape local optima is essential for discovering effective profiles in the complex space of natural language descriptions.

2.3 Content Retrieval API

The final component of our architecture allows content providers to serve personalized recommendations without accessing users' raw behavioral data. Providers participate by embedding their content catalogs in the shared embedding space and exposing standardized API endpoints that accept weighted profile embeddings from personal agents and return ranked recommendations using approximate nearest-neighbor search.

When a provider joins the system, they encode their entire catalog using the shared embedding model, creating a searchable index of item embeddings. Personal agents then convert their optimized natural language profiles into embeddings and query these endpoints, receiving ranked recommendations without revealing the underlying interaction history or profile structure. Agents can also apply filtering thresholds to discard items with low similarity scores, giving users fine-grained control over recommendation quality.

This API-based design provides several advantages: it enables any provider to participate without large user datasets or complex infrastructure and supports cross-provider recommendation portability, allowing the same profile to generate relevant recommendations across multiple content providers.

3 EXPERIMENTS

Datasets and Evaluation Protocol. We evaluate our approach on two real-world datasets from distinct content domains:

(1) **Microsoft News Dataset (MIND)** [37], which consists of news articles aggregated by MSN News from major news providers including CNN, Reuters, BBC, and Fox News.
(2) **Goodreads** book recommendation dataset [30], which is gathered from the Goodreads platform.

This selection allows us to assess the generalizability of our method across different types of content, providers, and user behavior.

Our evaluation protocol differs from traditional recommendation systems due to our decentralized architecture. While baseline methods train centralized models on large datasets with thousands of users to learn population-level patterns, our personal agents learn individual user profiles using only each user's own interaction history. This limited access to data presents a disadvantage in terms of discovering broader behavioral patterns. Due to computational constraints, we randomly sample 300 users for each experiment. Although this number is small relative to the full user base, we repeat the experiments five times with different user selections and report the average performance across these runs. All models, including our approach, are evaluated on the same sampled users in each run, ensuring a fair and consistent comparison. Table 2 shows dataset statistics for each experiment.

Table 2: Dataset statistics for MIND and Goodreads for each experiment. †Complete training set is only used for baseline methods; our method uses training data for users in test set.

	MIND		Goodreads	
Dataset	Train†	Test	Train†	Test
# content	65,238	3,631	16,833	2,980
tokens/title	13.56	10.56	6.10	8.10
# users	94,057	300	23,089	300
content/user	14.98	15.81	7.81	9.21
# pos	347,727	430	273,888	1,102
# neg	8,236,715	3201	485,233	1,860

Implementation Details. Our implementation uses open-source models to ensure reproducibility and practical deployment feasibility. Profile learning employs Mixtral-8x7B Instruct [12] for both initial profile generation and iterative refinement. The shared embedding space uses mxbai-embed-large [15] as the base model, fine-tuned using SentenceBERT [25] with contrastive learning.

For embedding model training, we use batch size 128 with gradient accumulation and multiple negatives ranking loss to effectively leverage in-batch negative examples. Profile optimization uses an 80-20 train-validation split with a maximum of 10 refinement iterations per user. We set temperature $T = \infty$ for uniform exploration based on our ablation studies showing superior performance compared to exploitation-focused strategies.

Baseline Methods. We compare against representative approaches from three recommendation paradigms:

(1) **NRMS** [36]: A neural news recommender using multi-head self-attention in a two-tower architecture, representing the standard deep learning approach to recommendation.
(2) **MINER** [16]: An explicit interest modeling system that learns multiple interest vectors per user through poly-attention mechanisms. This baseline is particularly relevant as it also attempts to capture diverse user preferences, though through latent rather than interpretable representations. (MIND dataset only, due to named entity requirements.)
(3) **GLIMPSE** [13]: A leading LLM-based recommender system for pointwise learning-to-rank, which set the state of the art in language-model-based recommendation upon its publication at EMNLP 2024 and remains a strong baseline today.

This selection enables comprehensive evaluation against both established neural methods and cutting-edge LLM approaches while providing direct comparison with systems sharing similar goals (interest modeling) or technical foundations.

Table 3: Performance comparison on MIND and Goodreads datasets (metrics shown as percentages). Our personal agent approach outperforms all baselines.

	MIND				Goodreads			
	AUC	MRR	nDCG@5	nDCG@10	AUC	MRR	nDCG@5	nDCG@10
NRMS	67.83	29.31	33.05	39.64	60.29	61.41	76.65	76.61
MINER	51.51	23.75	20.24	26.93	–	–	–	–
GLIMPSE	71.66	36.00	41.29	47.36	66.56	68.04	83.54	84.03
Personal Agent ($T = 0$)	71.40	39.97	37.77	43.16	68.86	83.29	86.37	87.14
Personal Agent ($T = \infty$)	**76.93**	**47.55**	**44.49**	**49.54**	**69.56**	**83.58**	**86.60**	**87.30**
Personal Agent ($T = 1$)	73.55	43.93	40.27	46.29	69.43	**83.58**	86.58	87.28

3.1 Main Results

Table 3 presents our main experimental results, demonstrating that personal agents achieve superior performance across all evaluation metrics on both datasets. On MIND, our approach with optimal exploration ($T = \infty$) achieves 76.93% AUROC, substantially outperforming the strongest baseline GLIMPSE (71.66%). Similarly, on Goodreads, we achieve 69.56% AUROC compared to GLIMPSE's 66.56%, while maintaining strong performance across ranking-focused metrics like nDCG@5 and nDCG@10.

Figures 3a and 3b illustrate the iterative refinement of user profiles, showing consistent improvements across successive optimization steps. Our approach outperforms traditional baselines within just two iterations and surpasses GLIMPSE between iterations 4 and 8, demonstrating the effectiveness of our optimization strategy.

Although AUROC was used as the optimization criterion, consistent gains are observed across all ranking metrics. On both datasets, personal agents achieve rapid initial improvements—especially in the first iteration—followed by steady performance gains. On the MIND dataset, our method exceeds the NRMS baseline within two iterations on both AUROC and NDCG@5, and surpasses GLIMPSE by the 4th iteration on AUROC and 8th iteration on NDCG@5. On Goodreads, it outperforms all baselines within the first one to two iterations. While the rate of improvement slows over time, performance continues to rise without any regression or plateau. These results confirm the effectiveness of our iterative profile refinement in optimizing natural language representations of user interest profiles using personal agents.

While these experiments validate our approach on two distinct domains (news and books), they do not directly measure cross-provider portability. Our evaluation focuses on demonstrating that recommendation quality depends solely on the shared embedding space and locally optimized profiles rather than on provider-specific training data. This design choice implies that a single profile could, in principle, retrieve relevant content from multiple providers as long as they embed their catalogs in the shared space. Empirically validating this portability is an important direction for future work.

3.2 Ablation Study

To assess the contribution of key architectural components to our system's performance, we conduct ablation studies on the MIND dataset, examining: (i) exploration strategies through temperature control, (ii) the impact of fine-tuning shared embedding space, and (iii) the role of learned interest weights in profile optimization.

Figure 4a confirms our hypothesis that uniform exploration ($T = \infty$) outperforms exploitation-focused strategies, enabling agents to escape local optima in the complex space of natural language profiles. Lower temperature values prematurely converge to suboptimal profiles, while uniform sampling maintains the diversity needed for effective optimization.

Figure 4b reveals that both embedding fine-tuning and interest weighting are essential components. Without fine-tuned embeddings, performance stagnates around 55% AUROC and deteriorates over iterations, as misaligned interest-content relationships provide misleading feedback to the LLMs and agents. Similarly, removing learned weights caps performance at 60% AUROC, preventing the system from properly balancing multiple interests within profiles.

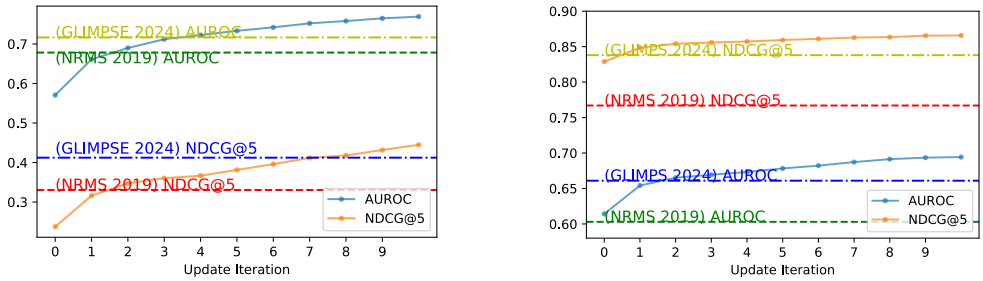
Our complete approach demonstrates the synergy between these components: starting from 58% AUROC, iterative refinement with properly aligned embeddings and learned weights achieves 77% AUROC, validating our architectural design choices.

3.3 Profile Editability Validation

A key advantage of our approach is enabling users to directly edit their preference representations. We validate this through controlled experiments on 100 users for news recommendations, testing both interest insertion and deletion scenarios across five content categories: *Music*, *Sports*, *Technology*, *Politics*, and *Local News*.

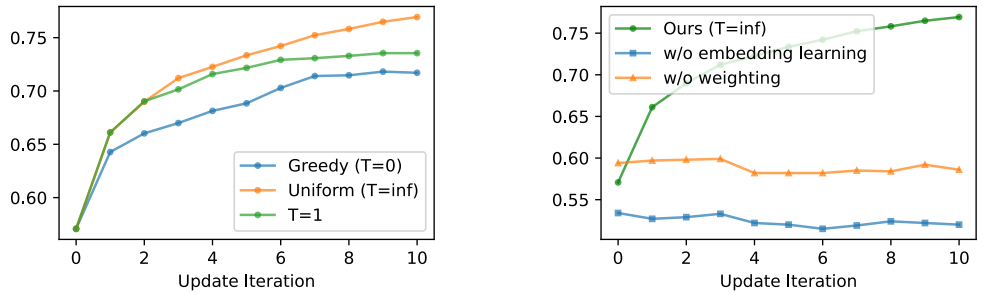
For insertion experiments, we add relevant interests to user profiles (e.g., "breaking news about local government decisions" for Local News) and measure whether subsequent recommendations reflect these changes. Table 4 shows that inserted interests successfully influence recommendations in 88% of cases on average, ranging from 84% (Local News) to 93% (Politics). This demonstrates that users can effectively steer their recommendations by articulating new interests. This capability is particularly important for users with limited or no interaction history, where their agents rely primarily on user-defined interests for bootstrapping user profiles.

Our deletion experiments provide complementary validation. Removing topic-related interests from profiles reduces recommendations for those topics to just 15% on average. In these experiments, deletions are purely mechanical; users remove interests to refine or sanitize their profiles rather than to indicate negative preferences. Consequently, content related to removed interests may still appear



(a) MIND Dataset (b) Goodreads Dataset

Figure 3: Ranking metrics for our approach over iterations on the MIND and Goodreads datasets against baselines.



(a) Temperature (b) Ablation

Figure 4: AUROC for different temperatures (4a) and variants of our approach (4b).

if it overlaps with other retained interests, which is expected and highlights the nuanced nature of real-world recommendations.

In scenarios where users might indicate negative interests, personal agents could, in principle, filter out undesired content post-retrieval. However, this mechanism is neither modeled in our approach nor tested in the current experiments, as we prioritize scalable content retrieval. Future work will investigate strategies to enhance profile expressiveness and editability, including mechanisms for modeling explicit negative preferences, while maintaining the system's scalability.

Table 4: Profile editability results showing the percentage of customers recommended content related to the edited topics after insertions and deletions.

Edited Topic	Insertion	Deletion
Music	87%	18%
Sports	91%	12%
Technology	85%	17%
Politics	93%	10%
Local News	84%	19%
Average	88%	15%

4 RELATED WORK

Our work lies at the intersection of several research areas. In this section, we situate it within existing literature and highlight how our agent-based approach differs from prior work.

Recommendation Systems. Modern recommendation systems have progressed from classical content-based methods [26] and collaborative filtering [27] to sophisticated neural architectures capable of processing diverse content types, including music [3], news [9], videos [45], and books [30]. Contemporary approaches leverage pre-trained language models such as *BERT* and *T5* [13, 43] to encode textual content and user histories into latent representations optimized for recommendation accuracy. However, these centralized approaches suffer from fundamental limitations. They require users to surrender behavioral data to content/service providers, creating vendor lock-in and limiting user agency over their preference representations. The resulting latent representations are opaque, preventing users from understanding or correcting how their preferences are modeled [42]. While recent work adds post-hoc explanations [6] or integrates collaborative signals into LLMs for explanation generation [20], such approaches still lack direct user control, editability, and support for cross-provider portability.

LLMs for Recommendation. The success of large language models has motivated their application to recommendation tasks [18]. Existing approaches broadly fall into four categories: (i) feature engineering [5, 8, 34], where LLMs augment traditional models with synthetic features; (ii) feature encoding [22], where LLMs serve as semantic encoders; (iii) direct scoring, where LLMs rank items through text generation; and (iv) user interaction guidance [7, 32]. Recent work has explored natural language user profiles for transparency [2, 24, 31, 44]. Ramos *et al.* [24] generate interpretable

profiles from reviews and fine-tune LLMs for recommendation, demonstrating that scrutability can be achieved without sacrificing performance. However, these approaches remain centralized, requiring providers to construct and manage user profiles and to rely on computationally expensive LLM inference as part of the recommendation process. Similarly, augmentation methods like KAR [38] and ONCE [19] use LLMs to generate user summaries but encode them into opaque embeddings, forfeiting interpretability. Our work addresses these limitations through a decentralized architecture where personal agents optimize structured profiles locally—maintaining individual interests with learnable weights—and recommendations are generated via efficient embedding-based retrieval rather than continuous LLM inference.

Federated Learning for Recommendations. Federated learning enables collaborative model training without centralizing raw data [21, 39]. Federated recommendation systems [1, 29] train models by collecting intermediate parameters instead of real user data, with recent extensions to LLM-based recommendation [17, 40]. However, these approaches face fundamental challenges: coordinated communication rounds with substantial parameter transmission costs, heterogeneous data distributions complicating convergence, and opaque model representations that users cannot inspect or modify. Our approach departs from this paradigm by decoupling personalization from collaborative training entirely. Instead, the shared embedding space is trained once on public content using LLM-generated synthetic pairs, eliminating the need for behavioral data while maintaining interpretable profiles that users can directly control and edit. This enables cross-provider portability, allowing new providers to deliver high-quality personalization by responding to profile embeddings optimized by personal agents without ever requiring access to behavioral data.

Agent-Based Recommendation Systems. Recent work on agent-based recommendations has explored user-centric explanation strategies [4] and metrics for evaluating explainability [11], focusing on making centralized systems more interpretable through post-hoc explanations. Other approaches investigate tool integration, user behavior simulation, and multi-agent collaboration [10, 23, 28, 35, 41]. Our work differs fundamentally: rather than adding explainability to centralized systems or simulating users, we design autonomous agents that construct and optimize structured natural language profiles under full user control. Each agent treats profile items as optimizable computational objects with learnable importance weights while preserving interpretability and enabling cross-provider portability. Agents interact with providers solely through a shared embedding space trained on content descriptions, enabling scalable retrieval while giving users complete ownership and control over their preference representations.

5 CONCLUSIONS AND FUTURE DIRECTIONS

We introduced a personal agent architecture that reimagines recommendation systems by satisfying three critical desiderata: user interpretability and control, cross-provider portability, and scalable retrieval. Through LLM-guided optimization of natural language profiles, our approach demonstrates that users can maintain full ownership and control over their preference representations while achieving superior recommendation quality and system scalability.

Our architecture resolves the fundamental tension between interpretability and scalability that has limited previous approaches. By decoupling profile learning (which leverages LLMs locally) from content retrieval (which uses efficient embeddings), we achieve the interpretability benefits of language models without the computational overhead of scoring millions of items. The resulting system enables user-controlled recommendation where users own and optimize profiles locally while accessing personalized content from multiple providers through a shared embedding space.

Experimental validation on MIND and Goodreads demonstrates that our approach achieves superior performance across all metrics while enabling direct profile editability (88% success rate for interest insertion). The iterative refinement process consistently improves recommendation quality, surpassing established baselines within $4 - 8$ iterations. Ablation studies confirm that uniform exploration strategies and properly aligned embeddings are crucial for effective profile optimization.

This work opens several promising research directions. First, we plan to explore richer editability mechanisms, including negative interests — explicit representations of content users wish to avoid. When users manually edit profiles, agents could automatically search the natural language space to better align edits with the shared embedding space, balancing user intent with retrieval optimization. Second, our content-based approach could be extended with collaborative filtering elements, enabling agents to leverage social signals or community preferences to (i) further improve recommendation quality while preserving transparency and interpretability and (ii) discover new interests through similar users. Third, an important direction is to empirically validate and further strengthen cross-provider portability. Although our architecture is designed to support portability across providers through a shared embedding space, real-world deployments may involve provider-specific or proprietary embedding models. In such cases, personal agents could optimize distinct profiles locally for each provider's embedding space, ensuring that high-quality personalization remains possible even when providers adopt different embedding representations for content–interest mappings. Finally, additional directions include extending our approach beyond text to multimedia content such as images, audio, and video, broadening its applicability to new domains and recommendation scenarios.

Our iterative profile refinement process naturally updates profiles over time as user preferences evolve. The ideal refinement frequency may vary across users. In our experiments, randomly sampled users exhibit widely varying interaction-history lengths, yet the method performs robustly across this variability, suggesting stability under different update granularities. Future work may explore periodic updates (e.g., weekly or monthly), threshold-based updates (e.g., after N new interactions), or on-demand refinement based on user volatility. Furthermore, the editable nature of profiles allows users to introduce entirely new interests at any time, enabling personalization to evolve dynamically under user direction.

Overall, we demonstrate the feasibility of user-centric recommendation systems that prioritize transparency and user control, without compromising performance or scalability, and are designed for cross-provider portability.

REFERENCES

[1] Zareen Alamgir, Farwa K Khan, and Saira Karim. 2022. Federated recommenders: methods, challenges and future. *Cluster Computing* 25, 6 (2022), 4075–4096.

[2] Seunghwan Bang and Hwanjun Song. 2025. LLM-based User Profile Management for Recommender System. *arXiv preprint arXiv:2502.14541* (2025).

[3] Jiajun Bu, Shulong Tan, Chun Chen, Can Wang, Hao Wu, Lijun Zhang, and Xiaofei He. 2010. Music recommendation by unified hypergraph: combining social media information and music content. *ACM Multimedia* (2010), 391–400. https://doi.org/10.1145/1873951.1874005

[4] Berk Buzcu, Emre Kuru, and Reyhan Aydogan. 2024. User-centric explanation strategies for interactive recommenders. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*. 2174–2176.

[5] Zhixuan Chu, Yan Wang, Qing Cui, Longfei Li, Wenqing Chen, Sheng Li, Zhan Qin, and Kui Ren. 2024. LLM-Guided Multi-View Hypergraph Learning for Human-Centric Explainable Recommendation. *arXiv preprint* (2024). https://doi.org/10.48550/arXiv.2401.08217

[6] Zhaocheng Du, Chuhan Wu, Qinglin Jia, Jieming Zhu, and Xu Chen. 2024. A Tutorial on Feature Interpretation in Recommender Systems. *ACM Conference on Recommender Systems* (2024). https://doi.org/10.1145/3640457.3687094

[7] Wenqi Fan, Zihuai Zhao, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Jiliang Tang, and Qing Li. 2023. Recommender Systems in the Era of Large Language Models (LLMs). *IEEE Transactions on Knowledge and Data Engineering* (2023). https://doi.org/10.1109/TKDE.2024.3392335

[8] Zhaolin Gao, Joyce Zhou, Yijia Dai, and Thorsten Joachims. 2024. End-to-end Training for Recommendation with Language-based User Profiles. *arXiv preprint* (2024). https://doi.org/10.48550/arXiv.2410.18870

[9] Florent Garcin, Christos Dimitrakakis, and Boi Faltings. 2013. Personalized news recommendation with context trees. *RecSys* (2013), 105–112. https://doi.org/10.1145/2507157.2507166

[10] Xu Huang, Jianxun Lian, Yuxuan Lei, Jing Yao, Defu Lian, and Xing Xie. 2025. Recommender AI agent: Integrating large language models for interactive recommendations. *ACM Transactions on Information Systems* 43, 4 (2025), 1–33.

[11] Joris Hulstijn, Igor Tchappi, Amro Najjar, and Reyhan Aydoğan. 2023. Metrics for evaluating explainable recommender systems. In *International Workshop on Explainable, Transparent Autonomous Agents and Multi-Agent Systems*. Springer, 212–230.

[12] Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, A. Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, L'elio Renard Lavaud, Lucile Saulnier, M. Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of Experts. *arXiv preprint* (2024). https://doi.org/10.48550/arXiv.2401.04088

[13] Nithish Kannen, Yao Ma, Gerrit J.J. van den Burg, and J. Faddoul. 2024. Efficient Pointwise-Pairwise Learning-to-Rank for News Recommendation. *Conference on Empirical Methods in Natural Language Processing* (2024). https://doi.org/10.18653/v1/2024.findings-emnlp.723

[14] Thomas Kramer. 2006. The Effect of Measurement Task Transparency on Preference Construction and Evaluations of Personalized Recommendations. (2006). https://doi.org/10.1509/jmkr.44.2.224

[15] Sean Lee, Aamir Shakir, Darius Koenig, and Julius Lipp. 2024. *Open Source Strikes Bread - New Fluffy Embeddings Model.* https://www.mixedbread.ai/blog/mxbai-embed-large-v1

[16] Jian Li, Jieming Zhu, Qiwei Bi, Guohao Cai, Lifeng Shang, Zhenhua Dong, Xin Jiang, and Qun Liu. 2022. MINER: Multi-Interest Matching Network for News Recommendation. *ACL* (2022), 343–352. https://doi.org/10.18653/V1/2022.FINDINGS-ACL.29

[17] Zhiwei Li, Guodong Long, Chunxu Zhang, Honglei Zhang, Jing Jiang, and Chengqi Zhang. 2024. Navigating the future of federated recommendation systems with foundation models. *arXiv preprint arXiv:2406.00004* (2024).

[18] Jianghao Lin, Xinyi Dai, Yunjia Xi, Weiwen Liu, Bo Chen, Xiangyang Li, Chenxu Zhu, Huifeng Guo, Yong Yu, Ruiming Tang, and Weinan Zhang. 2023. How Can Recommender Systems Benefit from Large Language Models: A Survey. *ACM Transactions on Information Systems* (2023). https://doi.org/10.48550/arXiv.2306.05817

[19] Qijiong Liu, Nuo Chen, Tetsuya Sakai, and Xiao-Ming Wu. 2023. ONCE: Boosting Content-based Recommendation with Both Open- and Closed-source Large Language Models. *Web Search and Data Mining* (2023). https://doi.org/10.1145/3616855.3635845

[20] Qiyao Ma, Xubin Ren, and Chao Huang. 2024. XRec: Large Language Models for Explainable Recommendation. *Conference on Empirical Methods in Natural Language Processing* (2024). https://doi.org/10.48550/arXiv.2406.02377

[21] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.

[22] Emiliano Penaloza, Olivier Gouvert, Haolun Wu, and Laurent Charlin. 2024. TEARS: Textual Representations for Scrutable Recommendations. *arXiv preprint* (2024). https://doi.org/10.48550/arXiv.2410.19302

[23] Ivens Da Silva Portugal, Paulo Alencar, and Donald Cowan. 2024. An agentic AI-based multi-agent framework for recommender systems. In *2024 IEEE International Conference on Big Data (BigData)*. IEEE, 5375–5382.

[24] Jerome Ramos, Hossen A. Rahmani, Xi Wang, Xiao Fu, and Aldo Lipani. 2024. Transparent and Scrutable Recommendations Using Natural Language User Profiles. *Annual Meeting of the Association for Computational Linguistics* (2024). https://doi.org/10.18653/v1/2024.acl-long.753

[25] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. https://arxiv.org/abs/1908.10084

[26] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2021. Recommender systems: Techniques, applications, and challenges. *Recommender systems handbook* (2021), 1–35.

[27] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. 285–295.

[28] Yunxiao Shi, Wujiang Xu, Zhang Zeqi, Xing Zi, Qiang Wu, and Min Xu. 2025. PersonaX: A Recommendation Agent-Oriented User Modeling Framework for Long Behavior Sequence. In *Findings of the Association for Computational Linguistics: ACL 2025*, Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (Eds.). Association for Computational Linguistics, Vienna, Austria, 5764–5787. https://doi.org/10.18653/v1/2025.findings-acl.300

[29] Zehua Sun, Yonghui Xu, Yong Liu, Wei He, Lanju Kong, Fangzhao Wu, Yali Jiang, and Lizhen Cui. 2024. A survey on federated recommendation systems. *IEEE Transactions on Neural Networks and Learning Systems* 36, 1 (2024), 6–20.

[30] Mengting Wan and Julian J. McAuley. 2018. Item recommendation on monotonic behavior chains. *RecSys* (2018), 86–94. https://doi.org/10.1145/3240323.3240369

[31] Lu Wang, Di Zhang, Fangkai Yang, Pu Zhao, Jianfeng Liu, Yuefeng Zhan, Hao Sun, Qingwei Lin, Weiwei Deng, Dongmei Zhang, et al. 2025. LettinGo: Explore User Profile Generation for Recommendation System. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*. 2985–2995.

[32] Qi Wang, Jindong Li, Shiqi Wang, Qianli Xing, Runliang Niu, He Kong, Rui Li, Guodong Long, Yi Chang, and Chengqi Zhang. 2024. Towards Next-Generation LLM-based Recommender Systems: A Survey and Beyond. *arXiv preprint* (2024). https://doi.org/10.48550/arXiv.2410.19744

[33] Weiquan Wang and May D. Wang. 2019. Effects of Sponsorship Disclosure on Perceived Integrity of Biased Recommendation Agents: Psychological Contract Violation and Knowledge-Based Trust Perspectives. *Information systems research* (2019). https://doi.org/10.1287/ISRE.2018.0811

[34] Yan Wang, Zhixuan Chu, Ouyang Xin, Simeng Wang, Hongyan Hao, Yue Shen, Jinjie Gu, Siqiao Xue, James Y. Zhang, Qing Cui, Longfei Li, Jun Zhou, and Sheng Li. 2024. LLMRG: Improving Recommendations through Large Language Model Reasoning Graphs. *AAAI Conference on Artificial Intelligence* (2024). https://doi.org/10.1609/aaai.v38i17.29887

[35] Yancheng Wang, Ziyan Jiang, Zheng Chen, Fan Yang, Yingxue Zhou, Eunah Cho, Xing Fan, Yanbin Lu, Xiaojiang Huang, and Yingzhen Yang. 2024. RecMind: Large language model powered agent for recommendation. (2024), 4351–4364.

[36] Chuhan Wu, Fangzhao Wu, Suyu Ge, Tao Qi, Yongfeng Huang, and Xing Xie. 2019. Neural News Recommendation with Multi-Head Self-Attention. *Conference on Empirical Methods in Natural Language Processing* (2019). https://doi.org/10.18653/v1/D19-1671

[37] Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, and Ming Zhou. 2020. MIND: A Large-scale Dataset for News Recommendation. *ACL* (2020), 3597–3606. https://doi.org/10.18653/V1/2020.ACL-MAIN.331

[38] Yunjia Xi, Weiwen Liu, Jianghao Lin, Xiaoling Cai, Hong Zhu, Jieming Zhu, Bo Chen, Ruiming Tang, Weinan Zhang, and Yong Yu. 2024. Towards Open-World Recommendation with Knowledge Augmentation from Large Language Models. In *Proceedings of the 18th ACM Conference on Recommender Systems* (Bari, Italy) *(RecSys '24)*. Association for Computing Machinery, New York, NY, USA, 12–22. https://doi.org/10.1145/3640457.3688104

[39] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)* 10, 2 (2019), 1–19.

[40] Rui Ye, Wenhao Wang, Jingyi Chai, Dihan Li, Zexi Li, Yinda Xu, Yaxin Du, Yanfeng Wang, and Siheng Chen. 2024. OpenFedLLM: Training large language models on decentralized private data via federated learning. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 6488–6499.

[41] Junjie Zhang, Yupeng Hou, Ruobing Xie, Wenqi Sun, Julian McAuley, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2024. Agentcf: Collaborative learning with autonomous language agents for recommender systems. In *Proceedings of the ACM Web Conference 2024*. 3679–3689.

[42] Yongfeng Zhang and Xu Chen. 2018. Explainable Recommendation: A Survey and New Perspectives. *Foundations and Trends in Information Retrieval* (2018).

https://doi.org/10.1561/1500000066
[43] Zizhuo Zhang and Bang Wang. 2023. Prompt Learning for News Recommendation. *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (2023). https://doi.org/10.1145/3539618.3591752
[44] Joyce Zhou, Yijia Dai, and Thorsten Joachims. 2024. Language-based user profiles for recommendation. *arXiv preprint arXiv:2402.15623* (2024).
[45] Renjie Zhou, Samamon Khemmarat, and Lixin Gao. 2010. The impact of YouTube recommendation system on video views. *Internet Measurement Conference* (2010), 404–410. https://doi.org/10.1145/1879141.1879193