Reproducing Transparent and Scrutable Recommendations: Exploring Open-Weight Models via Natural-Language User Profiles

Noah Mamie
University of Zurich
Department of Informatics
nmamie@ifi.uzh.ch

Laurin van den Bergh
University of Zurich
Department of Informatics
bergh@ifi.uzh.ch

Abstract

In this reproducibility study, we investigate the transparency and scrutability of recommender systems enhanced by incorporating generated natural-language user profiles that represent user preferences. The original paper explores the synthesis of user profiles from raw user-generated review text across domains such as movies and accommodations (Amazon Movies & TV, TripAdvisor). Crucially, these natural-language user profiles enable direct user interaction and intervention, allowing users to customize recommendations by correcting misattributed preferences or addressing cold-start settings. We successfully reproduce the core findings of the original study. Additionally, we extend the evaluation by conducting systematic context ablation experiments, multi-seed stability across five distinct random seeds to establish statistical reliability, and a mechanistic interpretability analysis using the `nnsight` framework to probe internal model representations under counterfactual profile perturbations. Our findings verify the original paper's claim that User Profile Recommendation (UPR) achieves competitive performance under its test-set reranking protocol and makes recommendations more transparent. Perturbing the natural-language profiles does change predictions, but it shifts predicted ratings uniformly across genres with no detectable genre-selective effect, leaving rankings unchanged even under direct activation steering. We trace this back to the rating-regression objective rather than the profile interface, with ranking-objective models clearly exceeding in this task.

1 Introduction

Recommender systems have increasingly transitioned from the classical content-based and collaborative filtering paradigms to Large Language Model (LLM)-driven architectures (Deldjoo et al., 2024). While classical approaches represent users as dense, uninterpretable latent vectors, recent work by Ramos et al. (2024) introduces User Profile Recommendation (UPR). UPR synthesizes a user's past review history into a natural-language summary profile. This textual representation allows users to inspect, scrutinize, and directly modify their own profiles to customize future recommendations.

In this work, we present a comprehensive reproduction and extension of the UPR methodology. We evaluate the framework on the Amazon Movies & TV (He and McAuley, 2016) and TripAdvisor[1] (Li et al., 2021) datasets, investigating both the recommendation accuracy and the semantic properties of the generated user profiles.[2]

Our primary contributions are as follows:

Exact Reproduction & Codebase Fixes: We reproduce the original paper's primary experimental results, resolving several issues and inconsistencies in the original repository.

End-to-End Reproducible Pipeline: We provide a fully reproducible codebase locked with `uv` dependency management, standardized CLI interfaces, and hardware-agnostic execution (`cuda`, `mps`, `cpu`).

Protocol Alignment & Comparison: We evaluate non-LLM baselines in Cornac (Salah et al., 2020) under standardized evaluation protocols matching the LLM evaluation script, presenting side-by-side benchmark tables comparing original published numbers with our reproduction.

Context Ablations & Multi-Seed Stability: We conduct extensive context ablation experiments (comparing user profiles, raw

[1] https://www.tripadvisor.com

[2] We release the full code for optimal reproducibility. We contacted the original authors to make them aware of this reproducibility study and to list points where the repository accompanying the original paper can be improved. They offered to consider our pull request to their repository.

Proceedings of the 9th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP, pages 1273–1284
October 29, 2026 ©2026 Association for Computational Linguistics

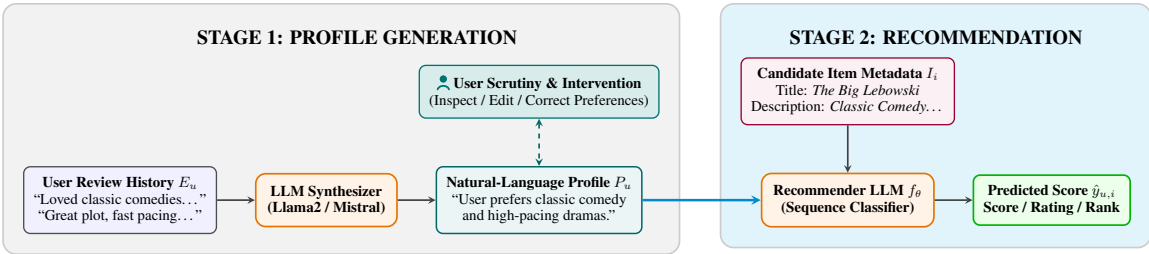

Figure 1: The UPR architecture pipeline by Ramos et al. (2024), including our modifications to test different context ablations. User review histories E_u are synthesized by an LLM into human-readable natural-language profiles P_u. Users can inspect and edit their profiles (scrutability/intervention). For user u and item i, a fine-tuned sequence classification model f_θ predicts recommendation scores $\hat{y}_{u,i}$ given the profile P_u and candidate item metadata I_i.

review history, and item-review history) and evaluate model variance across five distinct random initialization seeds.

Mechanistic Interpretability & Scrutability: We leverage the `nnsight` interpretability library to probe internal model activations, map profile embeddings via UMAP, score cluster key terms via gradient attributions, and track latent representation trajectories under counterfactual user profile perturbations.

2 Background & Methodology

2.1 Problem Formulation

Let $\mathcal{U}$ denote the set of users and $\mathcal{I}$ denote the set of items. Each user $u \in \mathcal{U}$ has an associated interaction history $E_u = \{e_{u,1}, e_{u,2}, \ldots, e_{u,|E_u|}\}$, where $e_{u,i}$ represents a written review for an item i previously consumed by u, and is often considered an explanation of a rating $y_{u,i}$. The UPR approach consists of two primary stages depicted in Figure 1:

1. **Profile Generation:** An LLM (e.g., Llama 2-7B or Mistral-7B) synthesizes the review history E_u into a concise, natural-language user profile P_u:

$$P_u = \text{LLM_Gen}(E_u) \tag{1}$$

2. **Recommendation / Rating Prediction:** A sequence classification model f_θ is fine-tuned to predict item relevance $\hat{y}_{u,i}$ given the user profile P_u and the item metadata I_i:

$$\hat{y}_{u,i} = f_\theta(\text{Prompt}(P_u, I_i)) \tag{2}$$

2.2 User Scrutability & Interventions

Unlike black-box latent vector embeddings $\mathbf{h}_u \in \mathbb{R}^d$, the natural-language profile P_u is human-readable. Users can edit P_u directly by removing incorrectly inferred preferences (e.g., removing a dislike for a specific subgenre) or inserting new preference constraints (e.g., adding an interest in horror movies).

3 Scope of Reproduction & Extensions

3.1 Scope of Reproduction

Our reproduction focuses on validating the core claim of Ramos et al. (2024): that natural-language user profiles can match or exceed the predictive performance of raw review histories while maintaining human interpretability and scrutability. We reproduce fine-tuning of the recommender LLM f_θ and baseline evaluations on the Amazon Movies & TV and TripAdvisor datasets using open-weight Llama2-7B (Touvron et al., 2023) and Mistral-7B (Jiang et al., 2023) profile generations. We made sure to reuse the profiles generated by Ramos et al. (2024), to ensure an adequate and fair comparison.

3.2 Extensions Beyond the Original Paper

We extend the original study in three key directions:

1. **Context Ablation Study:** We systematically evaluate model performance under three user representations (`user profile`, `review history`, and `item-review history`) paired with two item metadata representations (`item title` and `item title and description`).

2. **Multi-Seed Robustness:** To ensure that reported gains are not artifacts of favorable random initializations, we conduct 5-seed stability trials on the Amazon dataset.

3. **Mechanistic Interpretability & Scrutability:** We leverage the `nnsight` interpretability library (NDIF Team, 2024) to probe internal model activations, map profile embeddings

via UMAP, score cluster key terms via gradient attributions, and track latent representation trajectories under counterfactual user profile perturbations.

4 Experimental Setup

4.1 Datasets & Preprocessing

We utilize the preprocessed Amazon Movies & TV dataset and the TripAdvisor dataset, both formatted into the standard `train.jsonl`, `validation.jsonl`, and `test.jsonl` splits. Comprehensive details on data splits, prompt templates, fine-tuning configurations, hardware, and evaluation protocols are provided in Appendix A.

4.2 Baseline Models & Protocol Alignment

To ensure fair comparison, we evaluate several standard recommendation baselines using Cornac (Salah et al., 2020): `MostPop`, `UserKNN`, `ItemKNN`, `MF`, and `NeuMF`.

We aligned the baseline evaluation script (`rec_baselines.py`) with the LLM evaluation protocol (`evaluate.py`). Exact metric formulas and candidate pool definitions are detailed in Appendix A.

5 Results & Analysis

5.1 Baseline Comparison & UPR Reproduction

Table 1 presents a direct side-by-side comparison between the original published numbers from Ramos et al. (2024) in Part (a) and our reproduced results in Part (b). Comparing Part (a) and Part (b) demonstrates strong qualitative alignment: classical matrix factorization (`MF`) remains the strongest rating accuracy baseline across both published and reproduced benchmarks. Nevertheless, UPR models achieve highly competitive ranking metrics on TripAdvisor (nDCG@10: 0.961, MAP: 0.939) while providing transparent natural-language profiles. We highlight several key methodological nuances regarding baseline reproduction in Part (b):

Non-reproducible Baselines As indicated in gray (†) in Table 1, `PETER+` and `PEPLER-MLP` could not be reproduced because the official open-source repository contains no implementation code or execution scripts for these architectures. We assume that the metrics reported in the original paper were imported directly from prior literature.

NeuMF Model Mismatch Cornac's `NeuMF` is the implicit-feedback variant: rating values never enter its objective, and its scores correlate with held-out ratings at r=0.017, so no training length or rescaling reaches 0.943. The published figure reflects a rating-prediction NeuMF; we therefore report only ranking metrics numerically and the rest as `n/a`.

Single UPR Variant Execution Our objective is to validate the reproducibility of the core UPR methodology for recommending items from `User Profile` → `Title`), rather than to assess profiles (the performance of various LLMs for user profile generation. Therefore, we focus Part (b) on reproducing the primary UPR configuration based on the original Llama2-generated user profiles.

As shown in our subsequent context ablation study (Table 2), the UPR methodology is not only reproducible but also further improves when incorporating richer item metadata (`Title + Description`) or context history, achieving up to 0.895 MAP and 0.934 nDCG@10.

5.2 Context Ablation

Table 2 presents the ablation results across all input I_i and output context configurations on the Amazon Movies & TV dataset. The ablation results confirm the core hypothesis of Ramos et al. (2024). Concise natural-language user profiles achieve rating prediction errors (MAE 0.784, RMSE 1.063) and ranking performance (MAP 0.895, nDCG@10 0.929) that are highly competitive with full raw review histories (MAE 0.763, nDCG@10 0.926), while using substantially shorter prompt contexts and offering full human readability.

5.3 Multi-Seed Stability

To verify that evaluation significance is not impacted by random initialization or seed variance, we evaluated model performance across five distinct random seeds (seeds 37–41) alongside the reproduction baseline (Seed 42). Table 3 summarizes the results on the Amazon Movies & TV dataset.

The metrics display tight variance ($\sigma_{\text{MAP}} = 0.0009$, $\sigma_{\text{nDCG}} = 0.0007$), demonstrating that performance gains are statistically robust and independent of random seed choices (see Appendix A for detailed seed setup).

5.4 Mechanistic Interpretability

To investigate how the fine-tuned recommender internalizes textual user profiles and responds to

Table 1: Recommendation performance on Amazon Movies & TV and TripAdvisor: Side-by-side comparison between original published numbers (Ramos et al., 2024) in Part (a) and our reproduced results in Part (b). Best value per section per dataset is in **bold**, and second-best is underlined. Grayed-out entries (†) in Part (a) denote baselines whose source code was omitted from the official repository and could not be reproduced.

	Amazon Movies & TV				TripAdvisor			
Model	**RMSE ↓**	**MAE ↓**	**nDCG@10 ↑**	**MAP ↑**	**RMSE ↓**	**MAE ↓**	**nDCG@10 ↑**	**MAP ↑**
(a) Original Paper Reported Results (Ramos et al., 2024)								
MostPop	1.505	0.962	0.896	0.815	1.277	0.902	0.932	0.857
UserKNN	0.960	0.707	0.935	0.864	0.836	0.634	0.952	0.883
Item-KNN	1.045	0.790	0.897	0.823	0.890	0.683	0.953	0.852
MF	**0.925**	0.686	**0.941**	**0.870**	**0.786**	**0.599**	**0.960**	**0.891**
NeuMF	0.943	0.694	0.936	0.866	0.819	0.634	0.955	0.886
PETER+†	**0.924**	0.685	**0.941**	**0.870**	0.803	0.621	0.958	0.889
PEPLER-MLP†	0.925	**0.672**	**0.941**	0.869	0.793	0.606	0.959	0.889
UPR (Llama2)	0.944	0.678	0.938	0.866	0.804	0.616	0.955	0.885
UPR (Mistral)	0.941	0.679	0.940	**0.870**	0.804	0.610	0.952	0.887
(b) Our Reproduced Results								
MostPop	1.505	0.962	0.898	0.797	1.277	0.902	0.930	0.833
UserKNN	0.960	0.707	0.942	0.918	0.836	0.634	0.958	0.934
ItemKNN	1.045	0.790	0.906	0.874	0.890	0.683	0.934	0.901
MF	**0.925**	**0.686**	**0.948**	**0.924**	**0.786**	**0.599**	**0.964**	**0.942**
NeuMF	n/a	n/a	0.903	0.867	n/a	n/a	0.937	0.905
UPR (Original)	0.952	0.687	0.945	0.920	0.809	0.618	0.961	0.939

Table 2: Context Ablation Results on Amazon Movies & TV. Original configuration in *italics*, best in **bold**, second-best underlined.

User (P_u)	**Item (I_i)**	**RMSE ↓**	**MAE ↓**	**nDCG ↑**	**MAP ↑**
Profile	*Title*	*1.062*	*0.788*	*0.928*	*0.894*
Profile	Title+Desc	1.063	0.784	0.929	0.895
Reviews	Title	**1.043**	0.776	0.928	0.894
Reviews	Title+Desc	1.055	**0.763**	0.926	0.891
Item-Rev	Title	1.105	0.864	0.934	**0.902**
Item-Rev	Title+Desc	1.108	0.879	**0.934**	**0.902**

Table 3: Multi-seed evaluation results on Amazon Movies & TV across seeds 37–41 vs baseline (Seed 42, in *italics*). Best in **bold**, second-best underlined.

Seed	**RMSE ↓**	**MAE ↓**	**nDCG@10 ↑**	**MAP ↑**
37	0.948	0.690	0.945	0.921
38	0.949	**0.686**	**0.946**	0.921
39	**0.947**	0.689	0.945	0.920
40	0.960	0.705	0.944	0.919
41	0.949	0.690	**0.946**	**0.921**
42	*0.952*	*0.687*	*0.945*	*0.920*
$\mu \pm \sigma$	0.951±.005	0.691±.007	0.945±.001	0.920±.001

user interventions, we probe the final normalized layer of the transformer backbone using `nnsight`, extracting mean-pooled hidden activation vectors $\mathbf{h}_u \in \mathbb{R}^d$ for every user profile prompt (Appendix B). This captures the model's representation state immediately before classification.

Table 4: Counterfactual profile perturbation states used in the trajectory analysis. Steering keywords added at each transition are in **bold**.

State	**Natural-Language Profile Text**	**Target**
1 (baseline)	*"I like romantic comedies with authentic love stories and lighthearted romance."*	Romance
2 (weak)	*"I like romantic comedies with **lots of action scenes and fast-paced thrillers**."*	Action/Romance
3 (strong)	*"My favorite genre is **crime and mystery thrillers with suspenseful plots, detective investigations, and dark puzzles**."*	Crime

5.4.1 Representational structure

Figure 2 shows that natural-language profiles form well-separated groups in the model's 2D UMAP activation space, and tracks the representation under the three edits of Table 4. Re-embedding each edited profile yields smooth, continuous, and directional shifts: textual interventions do measurably change the model's internal user representation.

These groups do not align with user tastes. Labeling 1500 users by the dominant genre of their *actual* review history (romance 795, crime 530, action 175; retained only when at least three reviewed items carry a genre and one genre holds

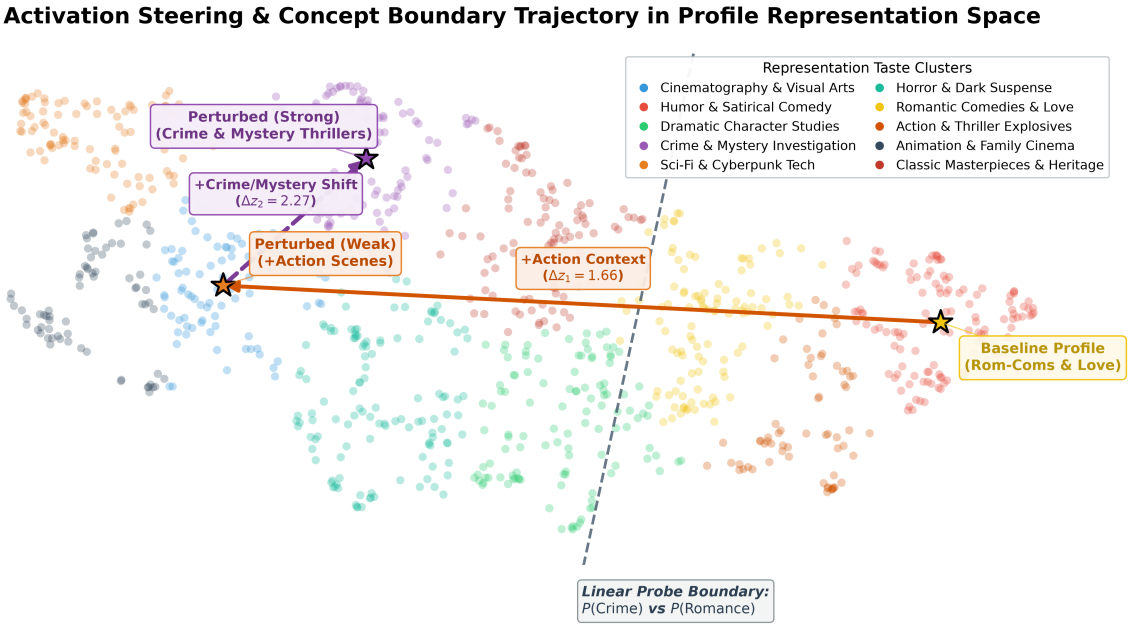

Figure 2: 2D UMAP visualization of final layer normalization activations extracted via `nnsight`. Background point clouds are population-wide groupings of profile representations, color-coded and labeled by their most salient profile terms. Trajectory vectors show the three counterfactual edits of Table 4, with a fitted linear probe boundary.

the majority) and comparing against K-means over the same representations gives chance agreement: adjusted Rand index 0.0008, normalized mutual information 0.0038, with every cluster reproducing the population base rate (46–59% romance). A linear probe for the same label reaches 0.449 accuracy against a 0.530 majority-class baseline, above a shuffled-label control (0.406) but too weak to be useful. The clusters, therefore, organize profiles lexically, and the archetype labels describe profile language rather than validated taste segments.

5.4.2 Do profile edits change predictions?

Movement in representation space need not change recommendations. Table 5 decomposes predicted-rating variance over a profile $\times$ item grid. The profile is not inert, explaining 47.6–48.5% of the variance, comparable to item identity, but it enters as a *user-level rating offset* rather than a key matched against item content: the interaction term accounts for only 4.1% with item titles alone and 16.1% once descriptions are supplied. This bounds what any content-conditioned steering could explain, and all analyses below use the title-and-description configuration, since a title-only model never observes the genre evidence such an effect requires.

Textual edits. We build three genre pools (romance, action, crime) of N=110 catalog items each by weak supervision over item descriptions and score all 330 candidates under each state. Editing the profile shifts predicted ratings substantially (+0.195 at State 2, +0.070 at State 3, both $p < 10^{-9}$), but identically across the three pools (one-way ANOVA, p=0.79 and p=0.60; Table 6). The residual differences point the predicted way: under the crime-targeting edit, crime items gain most (+0.086 against +0.059 for romance) and are the only pool to rise in rank (−3.9 of 330 positions, while action items fall by 3.9). No contrast is significant (one-sided p=0.15 for $\Delta\hat{y}$, p=0.31 for ΔRank), the crime share of the top-50 moves only from 28% to 30%, and a borderline result at N=40 did not survive tripling N.

Table 5: Variance decomposition of predicted ratings $\hat{y}$ over a grid of 25 real profiles $\times$ 24 catalog items. The profile explains roughly half the variance in both configurations, but as a user-level offset: the profile $\times$ item interaction, where content-based personalization must live, accounts for 4.1% and 16.1% respectively.

Input Configuration	Item	Profile	Interaction
Profile + item title	48.3%	47.6%	4.1%
Profile + title & descr.	35.5%	48.5%	16.1%

Direct activation steering. A textual edit reaches behavior in two steps, and the above does not specify which step fails. We therefore intervene on the representation itself, adding $\alpha\mathbf{d}$ to the resid-

Table 6: Behavioral validation of the perturbation states in Table 4. For each state, we score N=110 catalog items per genre pool (330 candidates) with the profile + title & description model, reporting mean predicted rating $\hat{y}$, mean rank in the pooled candidate set (lower is better), and the change relative to State 1.

	State 1 (Baseline)		State 2 (Weak)		State 3 (Strong)	
Item Genre Pool	$\hat{y}$	Rank	$\Delta\hat{y}$	ΔRank	$\Delta\hat{y}$	ΔRank
Romance	4.096	151.8	+0.186	+1.9	+0.059	+0.0
Action	4.008	167.8	+0.195	−1.3	+0.066	+3.9
Crime	3.999	176.9	+0.204	−0.6	**+0.086**	**−3.9**
Main effect (pooled)	—	—	+0.195, $p<10^{-50}$		+0.070, $p<10^{-9}$	
Genre-selectivity of $\Delta\hat{y}$	—	—	F=0.24, p=0.79		F=0.51, p=0.60	
Genre-selectivity of Δ*Rank*	—	—	F=0.08, p=0.92		F=0.45, p=0.64	

Table 7: Percentile at which each ranker places a user's held-out positive item among 99 items never interacted with (200 users, identical candidate sets; 0.5 is chance).

Ranker	Objective	Percentile	Top-1
WMF	ranking	0.806	13.0%
`MostPop`	popularity	0.697	6.0%
MF	rating	0.547	2.0%
UPR (title)	rating	0.542	1.5%
UPR (title+desc)	rating	0.510†	1.5%

†not different from chance (p=0.62); all others $p < 0.05$.

ual stream at the profile token positions, where $\mathbf{d}$ is the crime-minus-romance difference of means. Three checks precede any null: a zero vector and a *constant* vector are both exact no-ops (layer normalization removes the latter, so steering directions must be non-uniform), while a random direction is not. Additive steering of the final representation cannot be selective at all, since the linear head shifts every item by the same $\alpha(\mathbf{w}\cdot\mathbf{d})$; we therefore steer at blocks 2, 6 and 9, early enough to interact with item tokens through attention.

The result dissociates representation from behavior (Table 8 in Appendix B). The probe readout moves as intended, from $P(\text{crime})$=0.064 at α=0 to 1.000 at $\alpha\geq4$ – a complete reversal of apparent genre preference – while crime items' mean rating *falls* slightly (3.999 → 3.984) and their mean rank shifts by at most 0.2 of 330 positions at every layer. The probe reversal is partly guaranteed by construction, since the injected direction is the class-mean difference the probe reads; the behavioral null is the finding.

5.4.3 Does the model personalize at all?

These nulls admit a simpler explanation: there may be no personalization signal to steer. For each of the 200 users, we rank one held-out positive item against 99 items they have not interacted with, using identical never-interacted candidate sets for each ranker. The task is feasible on this data with a ranking-objective model, as Weighted Matrix Factorization (WMF) achieves 0.806. The profile model reaches only 0.542 with titles (p=0.047) and 0.510 with descriptions (p=0.62), below `MostPop` (0.697), which personalizes not at all (Table 7). Its scores are uncorrelated with item popularity (Spearman's $r = -0.03$ to -0.08, $n.s.$), even though the positives are far more popular than the distractors (81.6 versus 41.9 training interactions).

The ordering identifies the cause. While the ranking-objective model WMF reaches 0.806, rating-regression objectives reach 0.51–0.55. Every training pair is an item the user chose and reviewed, so the data holds no mismatched user–item pairs from which a content-matching rule could be learned, and the loss is minimized by a user offset plus an item offset – the structure of Table 5. This is consistent with the metrics in §5: test-set reranking over the Sakai condensed list orders only items the user already selected, for which item-side priors suffice. The limitation is thus a property of the rating-regression objective, inherited from the original formulation, rather than of natural-language profiles as an interface.

5.4.4 Controls for the attribution analysis

Figures 3 and 4 report token-level gradient attribution over profile tokens. This could equally reflect the architecture, the pretrained backbone or the profile corpus, so we repeat the global analysis on the pretrained backbone with a randomly initialized head, on the same architecture without pretrained weights, and on word-shuffled profiles, comparing against corpus-frequency and subword-count baselines. Because gradient magnitudes are not comparable across models, we summarize each

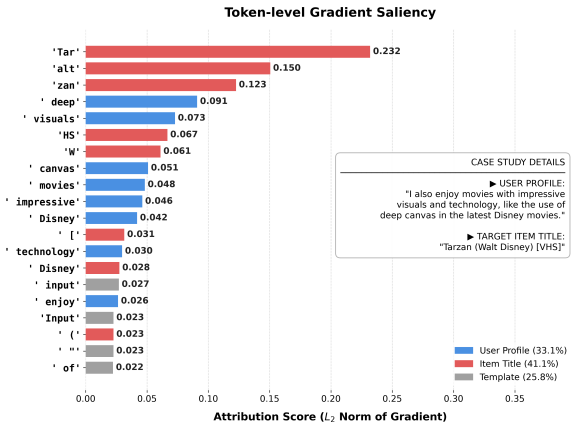

Figure 3: Local feature attribution for a single user profile and target item.

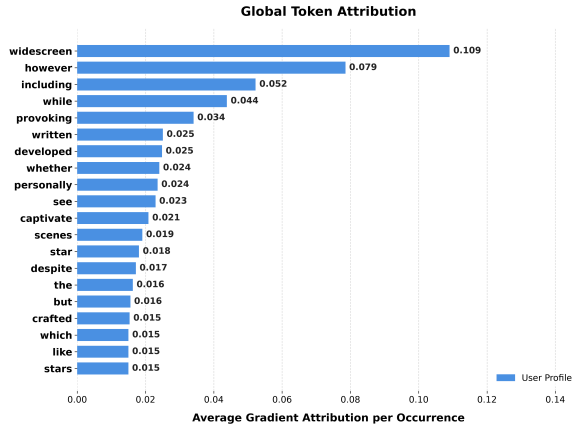

Figure 4: Global profile attributions over the Amazon user population.

condition by a scale-invariant statistic: mean attribution on a fixed a priori genre lexicon divided by mean attribution on all other profile words.

The controls (Table 9 in Appendix B) do not support reading these figures as evidence that fine-tuning concentrates attribution on preference features. Genre words receive roughly *half* the attribution of other profile words (ratio 0.546, p=0.002) and none appear in the top 20, which is led by `widescreen` alongside discourse and style terms (`however`, `while`, `levity`, `cinematography`). Eleven of those 20 recur under random initialization, and attribution correlates at ρ=0.807 with the fine-tuned model on word-shuffled profiles, so the pattern follows which words are present rather than their arrangement or meaning. Attribution is also negatively correlated with corpus frequency ($\rho = -0.320$, p=0.027), tracking word rarity, while subword count is not a significant driver (ρ=0.212, $n.s.$). We therefore treat these figures as descriptive of the saliency method and do not claim they isolate features acquired through fine-tuning.

Profile edits thus move the model's representation but not its recommendations, and scrutability in this formulation gives a user control over how generously they are modeled rather than what they are recommended. Training the same interface on a ranking objective with negative sampling is the change most likely to make the profile-edit behavior meaningful.

6 Discussion on Reproducibility

What was easy? The conceptual framework of UPR is intuitive, and the availability of preprocessed interaction datasets and initial source code provided a valuable starting point to reproduce the original paper's findings.

What was hard? Replicating the results required modifying the codebase to fix several issues: **Syntax & Variable Errors:** The original codebase contained a few broken variable references and undeclared identifiers. **Missing Environment Definitions:** No `requirements.txt` or environment locks were supplied. We fixed this by introducing a clean `uv` environment specification. **Evaluation Split Issue:** The original code validated the model using the test split instead of the validation split. Correcting this to ensure the test split is used only during testing yielded metric values closely aligned with the reported numbers. **Baseline Protocol Realignment:** Original baseline scripts evaluated models under mismatched candidate protocols. Standardizing all baselines in `rec_baselines.py` resolved this discrepancy.

7 Conclusion

This paper reproduces transparent and scorable recommendations using the UPR method developed by Ramos et al. (2024) and extends it to new context configurations, random seeds for robustness checks, and mechanistic interpretability for probing the model activation space with counterfactuals. Our reproduction study confirms that the natural-language user profiles generated by UPR are both useful for the final recommendation task and increase the transparency and scrutability of the recommendation model. This enables users to better understand the recommendations provided to them and modify the suggested user profile, steering predictions towards a state that better reflects reality.

Limitations

Our reproduction study focuses on the Amazon Movies & TV and TripAdvisor datasets, and evaluates profiles synthesized via the Llama2-7B model. While our mechanistic interpretability analysis confirms structured feature representations in latent space, exploring profile stability under varied prompt templates and open LLM generation settings remains an important area for future research. Furthermore, our steering validation experiments reveal an important issue with the regression objective of the loss function, as it inherently diminishes the model's ability to rank items. A natural next step is to explore a ranking objective to assess whether the user-profile bottleneck allows meaningful steering of user predictions based on their interests.

References

Yashar Deldjoo, Zhankui He, Julian McAuley, Anton Korikov, Scott Sanner, Arnau Ramisa, René Vidal, Maheswaran Sathiamoorthy, Atoosa Kasirzadeh, and Silvia Milano. 2024. A Review of Modern Recommender Systems Using Generative Models (GenRecSys). In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6448–6458, Barcelona Spain. ACM.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*, pages 507–517.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Lei Li, Yongfeng Zhang, and Li Chen. 2021. Personalized transformer for explainable recommendation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4947–4957.

NDIF Team. 2024. Nnsight: Mechanistic interpretability at scale. https://nnsight.net.

Jerome Ramos, Hossein A. Rahmani, Xi Wang, Xiao Fu, and Aldo Lipani. 2024. Transparent and scrutable recommendations using natural language user profiles. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13971–13984, Bangkok, Thailand. Association for Computational Linguistics.

Aghiles Salah, Quoc-Tuan Tran, and Hady W Lauw. 2020. Cornac: A comparative framework for multimodal recommendation. *Journal of Machine Learning Research*, 21(95):1–5.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

A Detailed Experimental Setup & Hyperparameters

To enable exact end-to-end reproduction, this section details the complete hardware, software, fine-tuning configuration, prompt formatting, model architectures, compute resources, and evaluation protocols used across all experiments.

A.1 Software Environment & Compute Resources

All models were fine-tuned and evaluated using PyTorch 2.2+ and HuggingFace `transformers` 4.38+. Environment dependencies are strictly locked using `uv` via `pyproject.toml` and `uv.lock`.

Compute Resources & Training Efficiency: Training a single UPR model configuration in its original baseline setup (`User Profile` $\rightarrow$ `Item Title`) on either the Amazon Movies & TV or TripAdvisor dataset requires approximately **1 hour** of wall-clock training time when executed in parallel across **8 $\times$ NVIDIA RTX 4090 (24GB)** GPUs (using PyTorch DataParallel across device indices `CUDA_VISIBLE_DEVICES=0,1,2,3,4,5,6,7`). Computations were also validated for hardware-agnostic execution on `mps` (Apple Silicon) and `cpu`.

The full experiment scales proportionally based on the number of evaluated configurations:

- **Context Ablation Study:** Evaluating all 6 combinations of input contexts (`user profile`, `review history`, `item-review history`) and output contexts (`item title`, `item title and description`) requires $6\times$ single-run time (≈ 6 wall-clock hours on $8\times$ RTX 4090 GPUs).
- **Multi-Seed Stability Study:** Training across 5 distinct random initialization seeds (seeds 37–41) requires $5\times$ single-run time (≈ 5 wall-clock hours on $8\times$ RTX 4090 GPUs).

A.2 Fine-Tuning Configuration & Hyperparameters

In the codebase (`train.py`), recommendation scoring is framed as sequence classification fine-tuning. The model (f_θ) is instantiated via HuggingFace's `AutoModelForSequenceClassification` initialized from pretrained causal LM backbones, defaulting to `gpt2` (124M parameters) or open-weight instruction models (`meta-llama/Llama-2-7b-chat-hf` and `mistralai/Mistral-7B-Instruct-v0.1`), with a single scalar output regression head (`num_labels=1`).

- **Model Initialization & Tokenizer Setup:** Loaded via HuggingFace's `AutoTokenizer`. Tokenizer padding is configured to right-side (`padding_side="right"`) with sequence truncation enabled (`truncation=True`, `padding="max_length"`), capped at a maximum sequence length of 300 tokens (`max_length=300`). The padding token is explicitly mapped to the end-of-sequence token (`pad_token = eos_token`), and model config is aligned via `model.config.pad_token_id = model.config.eos_token_id`.
- **Target Normalization & Inverse Transformation:** Ground-truth user rating scores $y \in [1, 5]$ are linearly normalized to $[0, 1]$ during dataset tokenization via $y_{\text{scaled}} = (y - 1)/4$. During evaluation, model scalar logit outputs $\hat{s} \in [0, 1]$ are transformed back to the original rating scale via $\hat{y} = \hat{s} \cdot 4 + 1$.
- **Optimizer, Learning Rate & Precision:** Fine-tuning uses the AdamW optimizer with a linear learning rate decay schedule (`lr_scheduler_type="linear"`). The initial learning rate is set to $\eta = 3 \times 10^{-4}$ (0.0003). Computation runs in full precision (FP32 / FP16 depending on CUDA hardware capability).
- **Batching, Epochs & Checkpoint Strategy:** Effective per-device batch size is set to 64 (`per_device_train_batch_size=64`, `per_device_eval_batch_size=64`). Fine-tuning runs for up to 5 epochs with epoch-wise evaluation and checkpointing (`save_strategy="epoch"`, `eval_strategy="epoch"`). Only the single best checkpoint is retained (`save_total_limit=1`, `load_best_model_at_end=True`). Early stopping is monitored via `EarlyStoppingCallback` with patience of 3 evaluation epochs (`patience=3`, `threshold=0.0`). Unused dataset columns are stripped during batch collation (`remove_unused_columns=True`).
- **Random Seeds:** Deterministic random initializations are enforced via

`set_random_seeds(seed)` across PyTorch, NumPy, and Python `random`. Baseline reproduction uses seed 42, while multi-seed robustness tests evaluate seeds 37, 38, 39, 40, and 41.

A.3 Evaluation Protocol & Metric Definitions

Evaluation is conducted via `evaluate.py` using ground-truth test splits for both Amazon and TripAdvisor Datasets. We used the original split sizes from Ramos et al. (2024) to ensure a fair comparison:

- **Test Candidate Pool:** For each user u, candidate items evaluated during ranking are restricted to the test interaction items in `test.jsonl`, aligning metric computation directly with standard test-set reranking protocols.
- **Calculated Metrics:** All metrics are implemented using the Cornac framework (Salah et al., 2020).
  1. **RMSE (Root Mean Squared Error):**
  $$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{u,i}(\hat{y}_{u,i} - y_{u,i})^2} \quad (3)$$
  2. **MAE (Mean Absolute Error):**
  $$\text{MAE} = \frac{1}{N}\sum_{u,i}|\hat{y}_{u,i} - y_{u,i}| \quad (4)$$
  3. **nDCG@10 (Normalized Discounted Cumulative Gain):** Measures ranking quality at cutoff $k = 10$, normalized by the Ideal DCG (IDCG):
  $$\text{DCG@k}(u) = \sum_{j=1}^{k} \frac{2^{y_{u,\pi(j)}} - 1}{\log_2(j+1)}, \quad (5)$$
  $$\text{nDCG@k} = \frac{1}{|U|}\sum_{u \in U} \frac{\text{DCG@k}(u)}{\text{IDCG@k}(u)}, \quad (6)$$
  where $\pi(j)$ represents the item ranked at position j for user u.
  4. **MAP (Mean Average Precision):** Computes the mean of Average Precision (AP) across all users U:
  $$\text{AP}(u) = \frac{1}{|R_u|}\sum_{j=1}^{N_u} P@j(u) \cdot \mathbb{I}(y_{u,\pi(j)} > 0), \quad (7)$$
  $$\text{MAP} = \frac{1}{|U|}\sum_{u \in U} \text{AP}(u), \quad (8)$$
  where R_u is the set of relevant items for user u, $P@j(u)$ is Precision at rank j, and $\mathbb{I}(\cdot)$ is an indicator function for relevance.
- **Output Aggregation:** Raw per-item prediction scores are saved, and aggregated dataset-wide metrics are written to disk.

A.4 Prompt Construction & Context Formats

The model is fed structured natural-language prompts formatted as follows:

Prompt Template Format: Input context and candidate item metadata are formatted as:

```
Input Context: {input_context}
Based on the input context, from a
scale of 1 to 5 (1 being lowest and
5 being highest), I would give
"{output_context}" a rating of
```

Where the input and output contexts vary across our ablation configurations:

1. **Input Context (`context_in`):**
   - `user profile`: The LLM-synthesized natural-language profile summary P_u.
   - `review history`: Concatenation of up to 5 recent item reviews written by user u excluding target item i.
   - `item-review history`: Concatenation of up to 5 recent reviews written by other users for candidate item i.
2. **Output Context (`context_out`):**
   - `item title`: The candidate item title string I_i.
   - `item title and description`: Candidate item title paired with text description ($I_i + D_i$).

B Mechanistic Interpretability Setup with `nnsight`

To probe the internal hidden representations of fine-tuned recommendation LMs, we leverage the `nnsight` interpretability library:

- **Probed Layer:** We trace hidden activations from the final layer normalization layer (`model.transformer.ln_f`).

Table 8: Activation steering at block 6. Adding $\alpha\mathbf{d}$ to the residual stream at the profile token positions, with $\mathbf{d}$ the crime-minus-romance difference of means, drives the probe readout across its full range while leaving ratings and rankings effectively unchanged. Blocks 2 and 9 behave identically. Rank is over the 330-item set of Table 6.

	Probe readout		**Mean $\hat{y}$**		**Crime**
α	P(crime)	P(romance)	Crime	Romance	**rank**
0	0.064	0.934	3.999	4.096	176.9
1	0.547	0.451	3.997	4.093	176.8
2	0.952	0.047	3.995	4.091	176.8
4	1.000	0.000	3.991	4.088	176.8
8	1.000	0.000	3.984	4.082	176.8

- **Pooling Method:** Mean-pooling across sequence tokens weighted by the input attention mask:

$$\mathbf{h}_u = \frac{\sum_{t=1}^{T} m_t \cdot \mathbf{h}_{u,t}}{\sum_{t=1}^{T} m_t} \tag{9}$$

  where $m_t \in \{0, 1\}$ is the attention mask at token position t.

- **UMAP & Clustering:** High-dimensional representations $\mathbf{h}_u \in \mathbb{R}^d$ are projected into 2D UMAP space using cosine distance (`n_neighbors=10`, `min_dist=0.05`, `random_state=42`). K-Means clustering ($K = 10$) identifies preference archetype clusters.

- **Counterfactual Steering & Profile Texts:** Three profile states (Baseline Romance, Weak Action, Strong Dark Crime; exact text in Table 4) are embedded and transformed through the fitted UMAP reducer to trace displacement vectors Δd across a fitted linear probe decision boundary P(Romance) vs. P(Crime).

The following tables further support the analysis in §5.4:

Table 9: Controls for the global attribution analysis. The genre ratio is mean attribution on a fixed genre lexicon over mean attribution on all other profile words (1.0 = no distinction). Agreement is the Spearman correlation of per-word attribution with the fine-tuned condition. Genre words are *under*-attributed, and the pattern largely survives removing fine-tuning, removing pretraining, and shuffling word order.

Condition	**Genre ratio**	**Agreement ρ**	**Top-20 overlap**
Fine-tuned (reported)	0.546	—	—
Pretrained backbone only	0.381	$+0.452$	$11/20$
Random initialization	1.169	$+0.229$	$11/20$
Fine-tuned, shuffled words	1.121	$+0.807$	$10/20$

vs corpus frequency: $\rho = -0.320$ (p=0.027); vs subword count: ρ=0.212 ($n.s.$).